Advanced Topics in Communication Networks

Internet Routing and Forwarding

Laurent Vanbever

nsg.ee.ethz.ch

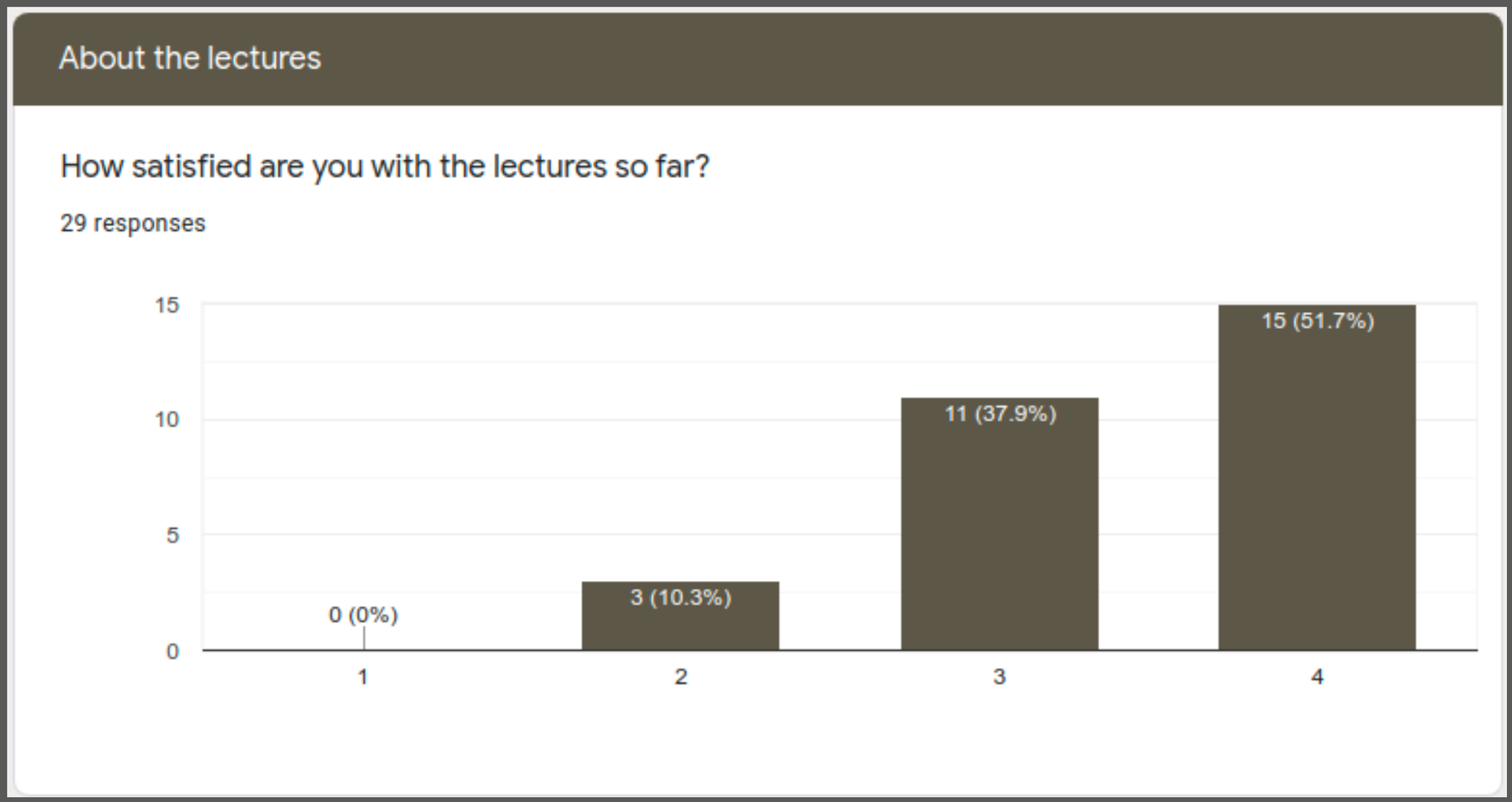
20 Oct 2020

Lecture starts at **14:15**

Subsets of the materials inspired and/or coming from Olivier Bonaventure

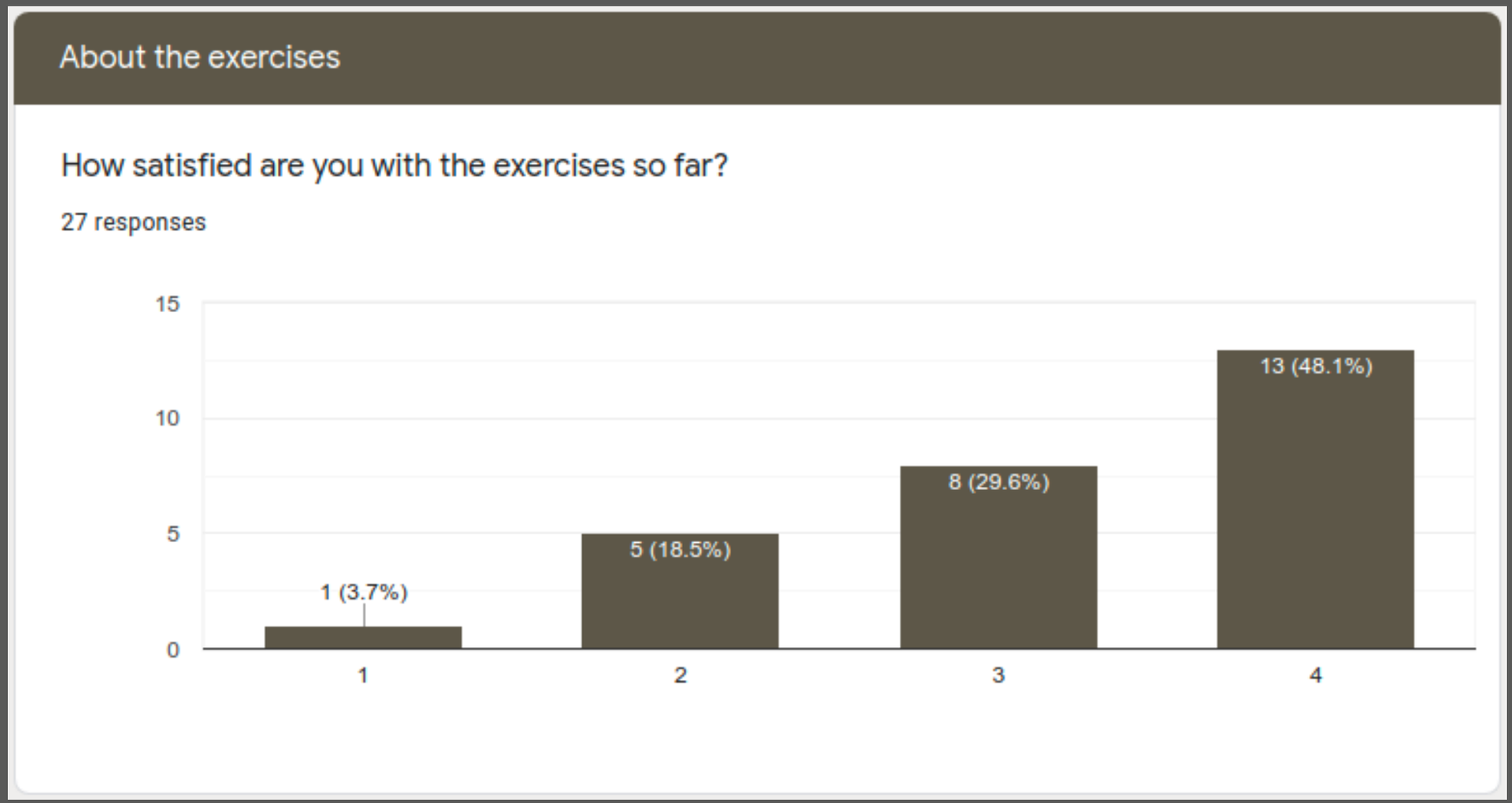
Looks like you appreciate the lecture & exercises :-)!

Thanks *a lot* for your feedback!



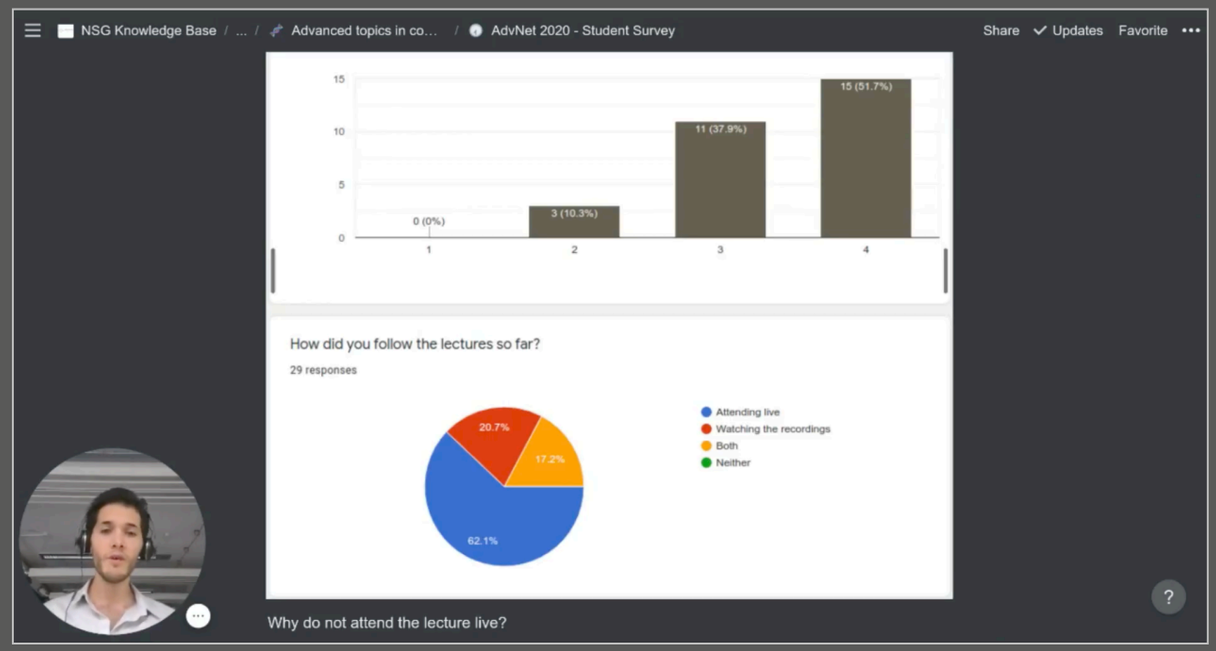
Looks like you appreciate the lecture & exercises :-)!

Thanks *a lot* for your feedback!



We'll do our best to take into account your comments and suggestions going forward — **it was not in vain**

Romain's analysis of your feedback on Moodle



Last week on

Advanced Topics in Communication Networks

Advanced Load Balancing

How can we do better than ECMP?

(using smarter data planes)

MPLS-based Traffic Engineering

RSVP-TE

(at long last)

Advanced Load Balancing

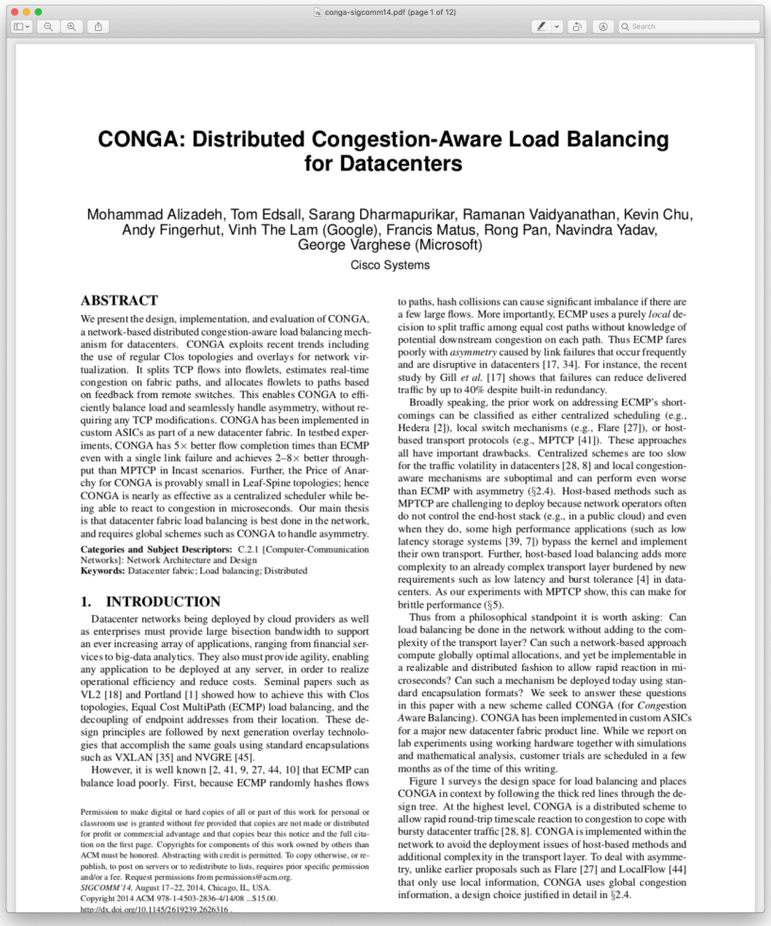
MPLS-based Traffic Engineering

How can we do better than ECMP?

(using smarter data planes)

How can we fight ECMP main two shortcomings: hash collisions and asymmetry

use network-wide feedback



conga-sigcomm14.pdf (page 1 of 12)

CONGA: Distributed Congestion-Aware Load Balancing for Datacenters

Mohammad Alizadeh, Tom Edsall, Sarang Dharmapurikar, Ramanan Vaidyanathan, Kevin Chu, Andy Fingerhut, Vinh The Lam (Google), Francis Matus, Rong Pan, Navindra Yadav, George Varghese (Microsoft)
Cisco Systems

ABSTRACT

We present the design, implementation, and evaluation of CONGA, a network-based distributed congestion-aware load balancing mechanism for datacenters. CONGA exploits recent trends including the use of regular Clos topologies and overlays for network virtualization. It splits TCP flows into flowlets, estimates real-time congestion on fabric paths, and allocates flowlets to paths based on feedback from remote switches. This enables CONGA to efficiently balance load and seamlessly handle asymmetry, without requiring any TCP modifications. CONGA has been implemented in custom ASICs as part of a new datacenter fabric. In testbed experiments, CONGA has 5× better flow completion times than ECMP even with a single link failure and achieves 2–8× better throughput than MPTCP in Incast scenarios. Further, the Price of Anarchy for CONGA is provably small in Leaf-Spine topologies; hence CONGA is nearly as effective as a centralized scheduler while being able to react to congestion in microseconds. Our main thesis is that datacenter fabric load balancing is best done in the network, and requires global schemes such as CONGA to handle asymmetry.

Categories and Subject Descriptors: C.2.1 [Computer-Communication Networks]: Network Architecture and Design
Keywords: Datacenter fabric; Load balancing; Distributed

1. INTRODUCTION

Datacenter networks being deployed by cloud providers as well as enterprises must provide large bisection bandwidth to support an ever increasing array of applications, ranging from financial services to big-data analytics. They also must provide agility, enabling any application to be deployed at any server, in order to realize operational efficiency and reduce costs. Seminal papers such as VL2 [18] and Portland [1] showed how to achieve this with Clos topologies, Equal Cost MultiPath (ECMP) load balancing, and the decoupling of endpoint addresses from their location. These design principles are followed by next generation overlay technologies that accomplish the same goals using standard encapsulations such as VXLAN [35] and NVGRE [45].

However, it is well known [2, 41, 9, 27, 44, 10] that ECMP can balance load poorly. First, because ECMP randomly hashes flows to paths, hash collisions can cause significant imbalance if there are a few large flows. More importantly, ECMP uses a purely *local* decision to split traffic among equal cost paths without knowledge of potential downstream congestion on each path. Thus ECMP fares poorly with *asymmetry* caused by link failures that occur frequently and are disruptive in datacenters [17, 34]. For instance, the recent study by Gill *et al.* [17] shows that failures can reduce delivered traffic by up to 40% despite built-in redundancy.

Broadly speaking, the prior work on addressing ECMP's shortcomings can be classified as either centralized scheduling (e.g., Hedera [2]), local switch mechanisms (e.g., Flare [27]), or host-based transport protocols (e.g., MPTCP [41]). These approaches all have important drawbacks. Centralized schemes are too slow for the traffic volatility in datacenters [28, 8] and local congestion-aware mechanisms are suboptimal and can perform even worse than ECMP with asymmetry (§2.4). Host-based methods such as MPTCP are challenging to deploy because network operators often do not control the end-host stack (e.g., in a public cloud) and even when they do, some high performance applications (such as low latency storage systems [39, 7]) bypass the kernel and implement their own transport. Further, host-based load balancing adds more complexity to an already complex transport layer burdened by new requirements such as low latency and burst tolerance [4] in datacenters. As our experiments with MPTCP show, this can make for brittle performance (§5).

Thus from a philosophical standpoint it is worth asking: Can load balancing be done in the network without adding to the complexity of the transport layer? Can such a network-based approach compute globally optimal allocations, and yet be implementable in a realizable and distributed fashion to allow rapid reaction in microseconds? Can such a mechanism be deployed today using standard encapsulation formats? We seek to answer these questions in this paper with a new scheme called CONGA (for *Conges*tion *A*ware Balancing). CONGA has been implemented in custom ASICs for a major new datacenter fabric product line. While we report on lab experiments using working hardware together with simulations and mathematical analysis, customer trials are scheduled in a few months as of the time of this writing.

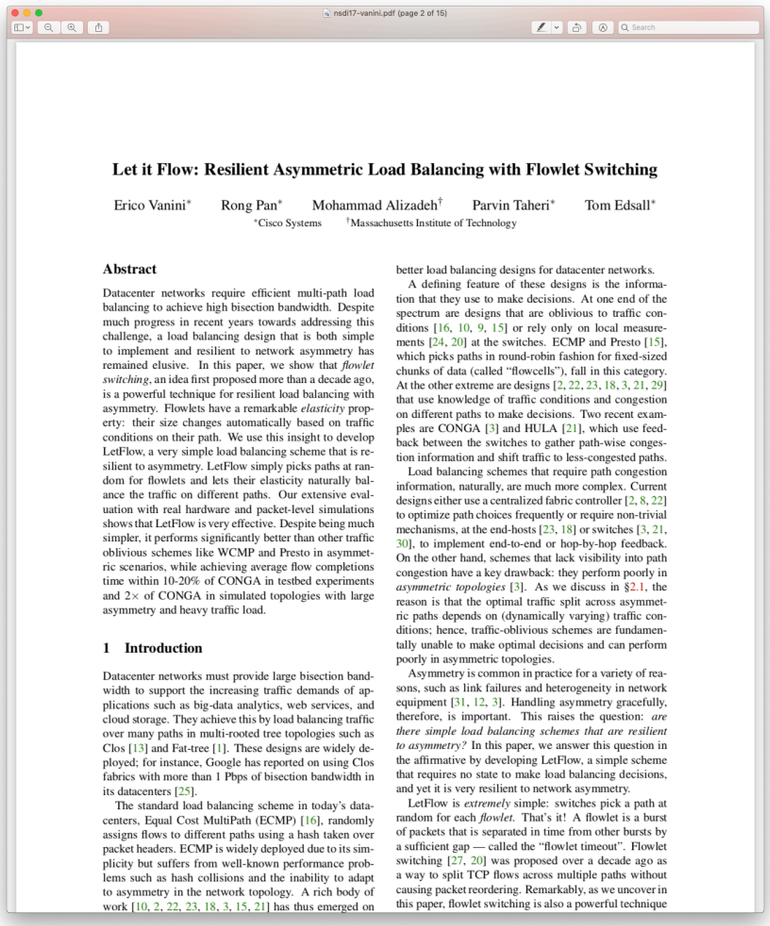
Figure 1 surveys the design space for load balancing and places CONGA in context by following the thick red lines through the design tree. At the highest level, CONGA is a distributed scheme to allow rapid round-trip timescale reaction to congestion to cope with bursty datacenter traffic [28, 8]. CONGA is implemented within the network to avoid the deployment issues of host-based methods and additional complexity in the transport layer. To deal with asymmetry, unlike earlier proposals such as Flare [27] and LocalFlow [44] that only use local information, CONGA uses global congestion information, a design choice justified in detail in §2.4.

SIGCOMM'14, August 17–22, 2014, Chicago, IL, USA.

http://dx.doi.org/10.1145/2619239.2626316.

CONGA [SIGCOMM'14]

locally load balance "elastic" entities



nsdi17-vanini.pdf (page 2 of 15)

Let it Flow: Resilient Asymmetric Load Balancing with Flowlet Switching

Erico Vanini* Rong Pan* Mohammad Alizadeh† Parvin Taheri* Tom Edsall*
*Cisco Systems †Massachusetts Institute of Technology

Abstract

Datacenter networks require efficient multi-path load balancing to achieve high bisection bandwidth. Despite much progress in recent years towards addressing this challenge, a load balancing design that is both simple to implement and resilient to network asymmetry has remained elusive. In this paper, we show that *flowlet switching*, an idea first proposed more than a decade ago, is a powerful technique for resilient load balancing with asymmetry. Flowlets have a remarkable *elasticity* property: their size changes automatically based on traffic conditions on their path. We use this insight to develop LetFlow, a very simple load balancing scheme that is resilient to asymmetry. LetFlow simply picks paths at random for flowlets and lets their elasticity naturally balance the traffic on different paths. Our extensive evaluation with real hardware and packet-level simulations shows that LetFlow is very effective. Despite being much simpler, it performs significantly better than other traffic oblivious schemes like WCMP and Presto in asymmetric scenarios, while achieving average flow completions time within 10-20% of CONGA in testbed experiments and 2× of CONGA in simulated topologies with large asymmetry and heavy traffic load.

1 Introduction

Datacenter networks must provide large bisection bandwidth to support the increasing traffic demands of applications such as big-data analytics, web services, and cloud storage. They achieve this by load balancing traffic over many paths in multi-rooted tree topologies such as Clos [13] and Fat-tree [1]. These designs are widely deployed; for instance, Google has reported on using Clos fabrics with more than 1 Pbps of bisection bandwidth in its datacenters [25].

The standard load balancing scheme in today's datacenters, Equal Cost MultiPath (ECMP) [16], randomly assigns flows to different paths using a hash taken over packet headers. ECMP is widely deployed due to its simplicity but suffers from well-known performance problems such as hash collisions and the inability to adapt to asymmetry in the network topology. A rich body of work [10, 2, 22, 23, 18, 3, 15, 21] has thus emerged on better load balancing designs for datacenter networks.

A defining feature of these designs is the information that they use to make decisions. At one end of the spectrum are designs that are oblivious to traffic conditions [16, 10, 9, 15] or rely only on local measurements [24, 20] at the switches. ECMP and Presto [15], which picks paths in round-robin fashion for fixed-sized chunks of data (called "flowcells"), fall in this category. At the other extreme are designs [2, 22, 23, 18, 3, 21, 29] that use knowledge of traffic conditions and congestion on different paths to make decisions. Two recent examples are CONGA [3] and HULA [21], which use feedback between the switches to gather path-wise congestion information and shift traffic to less-congested paths.

Load balancing schemes that require path congestion information, naturally, are much more complex. Current designs either use a centralized fabric controller [2, 8, 22] to optimize path choices frequently or require non-trivial mechanisms, at the end-hosts [23, 18] or switches [3, 21, 30], to implement end-to-end or hop-by-hop feedback. On the other hand, schemes that lack visibility into path congestion have a key drawback: they perform poorly in *asymmetric topologies* [3]. As we discuss in §2.1, the reason is that the optimal traffic split across asymmetric paths depends on (dynamically varying) traffic conditions; hence, traffic-oblivious schemes are fundamentally unable to make optimal decisions and can perform poorly in asymmetric topologies.

Asymmetry is common in practice for a variety of reasons, such as link failures and heterogeneity in network equipment [31, 12, 3]. Handling asymmetry gracefully, therefore, is important. This raises the question: *are there simple load balancing schemes that are resilient to asymmetry?* In this paper, we answer this question in the affirmative by developing LetFlow, a simple scheme that requires no state to make load balancing decisions, and yet it is very resilient to network asymmetry.

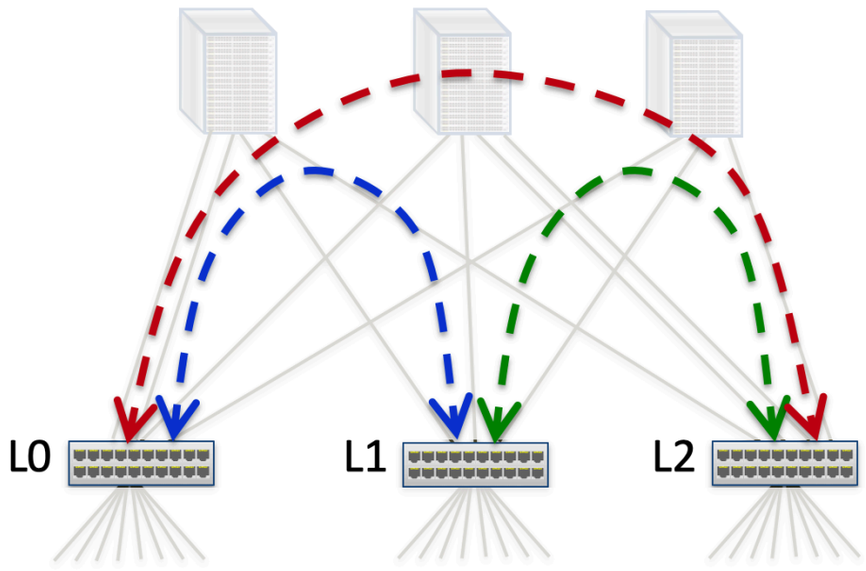
LetFlow is *extremely* simple: switches pick a path at random for each *flowlet*. That's it! A flowlet is a burst of packets that is separated in time from other bursts by a sufficient gap — called the "flowlet timeout". Flowlet switching [27, 20] was proposed over a decade ago as a way to split TCP flows across multiple paths without causing packet reordering. Remarkably, as we uncover in this paper, flowlet switching is also a powerful technique

LetFlow [NSDI'17]

CONGA in 1 Slide

1. Leaf switches (top-of-rack) track congestion to other leaves on different paths in near real-time

1. Use greedy decisions to minimize bottleneck util



Fast feedback loops between leaf switches, directly in dataplane

12

Source: CONGA: Distributed Congestion-Aware Load Balancing for Datacenters, Mohammad Alizadeh et al., 2014

LetFlow

Simple:

Randomly assign Flowlets to available paths

Flowlets:

"Flowlets are bursts from same flow separated by at least Δ"

"the main origin of flowlets is the burstiness of TCP at RTT and sub-RTT scales."

Kandula et al, "Dynamic load balancing without packet reordering", (2007)

Source: Let It Flow Resilient Asymmetric Load Balancing with Flowlet Switching, Vanini et al., 2017

Advanced Load Balancing

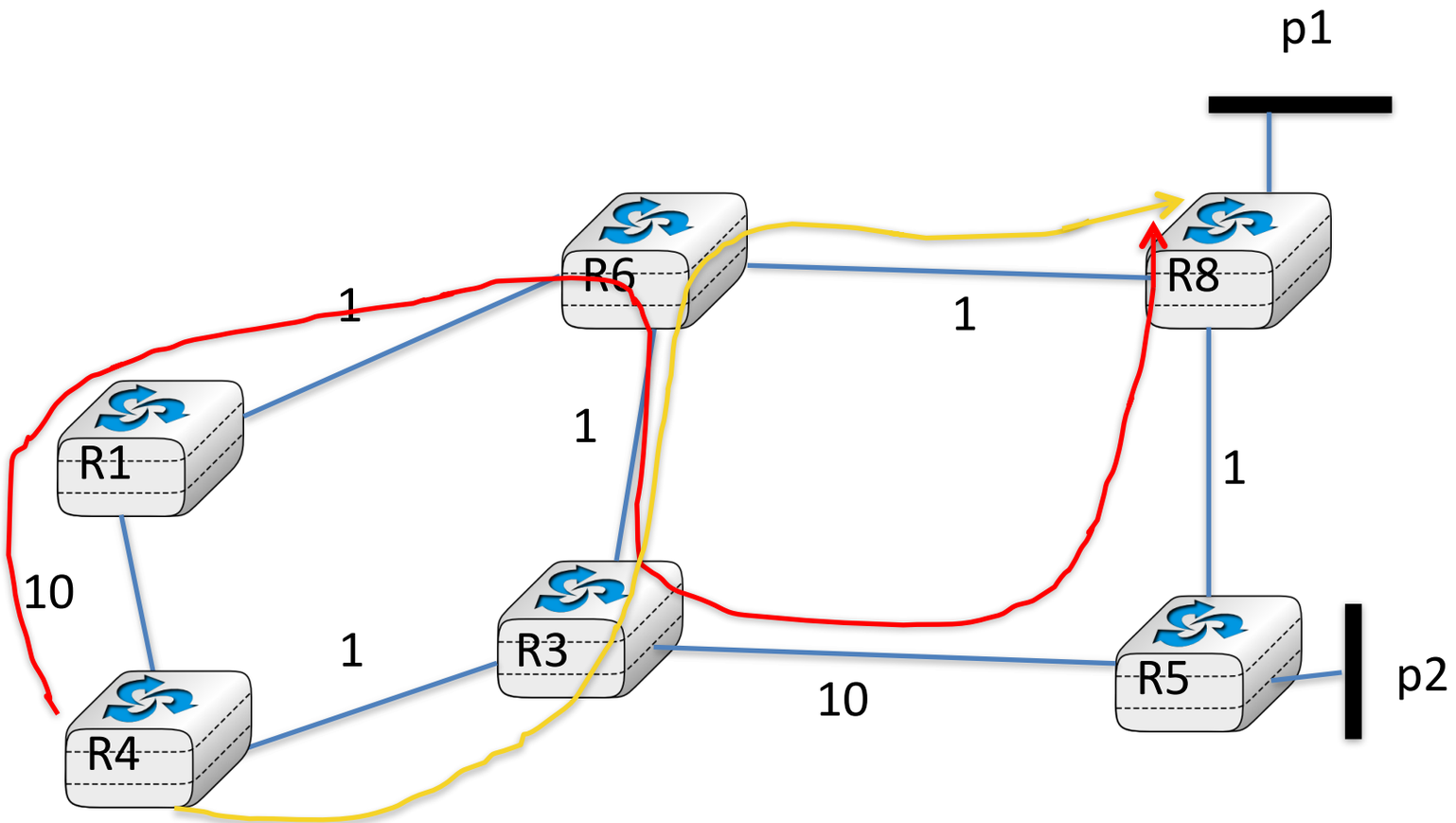
MPLS-based Traffic Engineering

RSVP-TE

(at long last)

MPLS-based Traffic Engineering

- How to create LSPs along a non-shortest path ?



MPLS-based Traffic Engineering

- Two sub-problems

1. How to find a path through the network that meets the requirements for the LSP?
2. How to signal the LSP along this chosen path?

MPLS TE relies on new link attributes propagated by link state routing protocols

- link type and link id
- local and remote IP addresses
- Traffic Engineering metric
  - additional metric to specify the cost of this link (usually delay)
- maximum bandwidth
  - maximum amount of bandwidth usable on this link
- maximum reservable bandwidth
  - maximum amount of bandwidth that can be reserved by LSPs
- unreserved bandwidth
  - amount of bandwidth that is not yet reserved by LSPs
- resource class/color
  - can be used to specify the type of link
    (e.g. expensive link would be colored in red and cheap links in green)

How does an ingress LSR find a compliant path through the network ?

- Path that minimises additive constraint
  - Propagation delay (TE metric)
  - IGP metric

- Dijkstra algorithm
  - Works with only one constraint, in practice either
    - minimize Sum(TE metrics)
    - minimize Sum(IGP metrics)

This week on

Advanced Topics in Communication Networks

MPLS-based Traffic Engineering

RSVP-TE

(at long last)

Quality of Service

How do we manage congestion?

(locally, on a per link basis)

MPLS-based Traffic Engineering

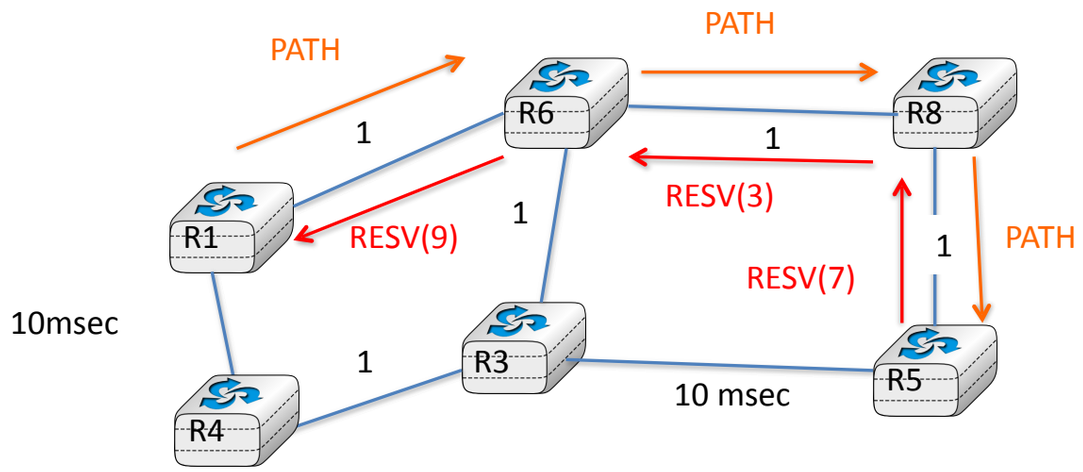
Quality of Service

RSVP-TE
(at long last)

Let's switch to
03_01c_te.pdf, slide 45/65

How to signal TE LSPs ?

- By using RSVP
  - PATH is used to request a label
  - RESV returns the allocated label



MPLS-based Traffic Engineering

Quality of Service

How do we manage congestion?

(locally, on a per link basis)

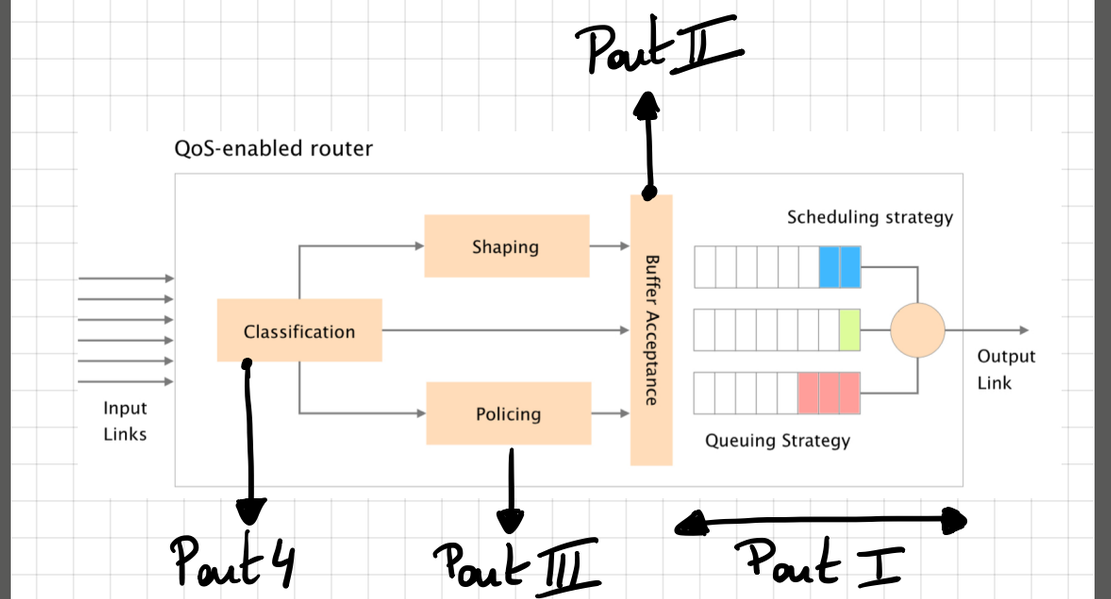
Let's switch to

04_qos_notes.pdf

Advanced Topics in Communication Networks

Lecture 6 : Quality of Service / 20.10.2020

Prof. Laurent VANBEVER - nsg.ee.ethz.ch



Advanced Topics in Communication Networks

Internet Routing and Forwarding

Laurent Vanbever

nsg.ee.ethz.ch

ETH Zürich (D-ITET)

20 Oct 2020