Understanding Data Center Traffic Characteristics

Theophilus Benson[⋆], Ashok Anand[⋆], Aditya Akella[⋆] and Ming Zhang[†]
[⋆]UW-Madison, [†]Microsoft Research

ABSTRACT

As data centers become more and more central in Internet communications, both research and operations communities have begun to explore how to better design and manage them. In this paper, we present a preliminary empirical study of end-to-end traffic patterns in data center networks that can inform and help evaluate research and operational approaches. We analyze SNMP logs collected at 19 data centers to examine temporal and spatial variations in link loads and losses. We find that while links in the core are heavily utilized the ones closer to the edge observe a greater degree of loss. We then study packet traces collected at a small number of switches in one data center and find evidence of ON-OFF traffic behavior. Finally, we develop a framework that derives ON-OFF traffic parameters for data center traffic sources that best explain the SNMP data collected for the data center. We show that the framework can be used to evaluate data center traffic engineering approaches. We are also applying the framework to design network-level traffic generators for data centers.

Categories and Subject Descriptors

D.4.8 [**Performance**]: Measurements, Modeling and prediction

General Terms

Measurement

Keywords

Data center traffic, traffic modeling

1. INTRODUCTION

In recent years, data centers or large clusters of servers have been increasingly employed in university, enterprise and consumer settings to run a variety of applications. These range from Internet facing applications such as web services, instant messaging and gaming, to computation intensive applications such as indexing Web content, data analysis and scientific computing.

While there is growing interest both in the research and in the operations communities on data center network design and management techniques for optimally supporting various applications, to the best of our knowledge, very little is known about traffic characteristics within data center networks. For instance, how do traffic volumes and loss rates vary with time and with the location of a link in a multi-tier data center? What are the underlying arrival processes and interarrival times that define data center traffic? How bursty is the traffic?

Lack of this information impedes both research and operations. For instance, research proposals for switching fabrics and load balancing schemes are evaluated using synthetic workloads such as mixtures of long and short-lived TCP flows [4, 2]. It is unclear how these proposals perform relative to each other and to current schemes in face of bursty traffic. Similarly, anecdotal evidence suggests that operators largely reuse or tweak traffic engineering mechanisms designed for enterprises or wide-area networks to manage limited data center network resources. However, this approach is suitable only if link-level traffic in data centers has similar properties as those in enterprises or wide-area networks.

Today, link-level statistics can be readily obtained via SNMP polls. Thus, it becomes immediately possible to study coarse grained characteristics such as average traffic volumes, loss rates etc. While interesting, this characterization is of limited value. In particular, SNMP data is too coarse-grained to study fine-grained traffic properties such as interarrival times and burstiness that have a significant impact on the effectiveness of switching mechanisms and traffic engineering schemes. Of course, it is possible to obtain fine-grained data today by enabling packet logging and Netflow universally. However, given the size of modern data centers, instrumenting all the links and switches to collect this information is simply infeasible.

In this paper, we present a preliminary investigation of traffic characteristics in data centers, leading up to a new framework that bridges the gap between the need for fine-grained information about data center traffic and the availability of just coarse-grained data.

We start our study by analyzing the SNMP data from 19 production data centers to conduct a macroscopic study of temporal and spatial variations in traffic volumes and loss rates in data center switching infrastructures. We find that the average link loads are high in the core of data center switching architectures and they fall progressively as one moves toward the edge. In contrast, average link losses are higher at the edge and lowest at the core (Section 4).

We then analyze finer-grained packet traces collected from a small number of edge switches in a data center to do a microscopic study of traffic behavior in data centers. We find that traffic at the data center edge can be characterized by ON-OFF patterns where the ON and OFF periods, and packet interarrival times with an ON period follow lognormal distributions (Section 4).

Finally, we develop a strawman framework for reverse engineering fine-grained characteristics for data center traffic using coarse grained information (i.e. SNMP counters), coupled with high-level inferences drawn from finer-grained information collected from a small fraction of network links (i.e. the fact that traffic is ON-OFF in nature). This framework is particularly useful to simulate the prevalent traffic characteristics in a data center even though it is not possible to collect fine-grained data everywhere. Our framework takes as input the coarse-grained loss rate and traffic volume distribution at a network link, and outputs parameters for the traffic arrival process on the link that best explains the coarse-grained information. This framework combines ideas from search space exploration with statistical testing to quickly examine the space of possible ON-OFF traffic source patterns in order to obtain ones that best explain the coarse-grained information.

Our framework has two applications: (1) It can be used to optimize traffic engineering and load balancing mechanisms. To illustrate this, we apply the framework to SNMP data from a data center network. We leverage the insights derived from the framework to infer that packet losses on links close to the edge of the data center occur in bursts. We further note that the bursts are too short-lived for traditional traffic engineering approaches to react to them. Thus, a fine-grained traffic engineering approach may have to be adopted by the data center in question. (2) It can be used to design workload generators for data centers. We are currently developing such a data center network traffic generator based on our empirical insights.

2. RELATED WORK

Traffic analysis in ISP backbones faces similar challenges to data centers in terms of the significant investment in infrastructure required to capture fine-grained information. As with data centers today, ISP backbones collect coarse grained information using SNMP. In the ISP case, this granularity proves sufficient for most management tasks such as traffic engineering [3]. However, as prior work [2] has shown, such coarse grained information is insufficient for traffic engineering needs in data centers. We develop an algorithm for deriving fine grained information from the coarse grained SNMP data, which can then be used in traffic engineering.

Numerous studies [5, 7] have been performed on modeling wide-area and ethernet traffic. Such models have informed various approaches for traffic engineering, anomaly detection, provisioning and synthetic workload generation. However, no similar models exist for the nature of traffic in data centers. In that respect, we take the first steps toward an extensive study and modeling of the traffic behavior in data centers.

Finally, recent research on data centers utilized toy traffic models or WAN-based models for evaluation (e.g., constant traffic in [2]). Our observations can be used to create more realistic workloads to evaluate current and future proposals for data center design and management.

3. DATA SETS

In this section, we describe the datasets used in our study. We collected two sets of measurement data. The first data set comprised of SNMP data extracted from 19 corporate and enterprise data centers hosting either intranet and extranet server farms or internet server farms. These data centers support a wide range of applications such as search, video streaming, instant messaging, map-reduce, and web applications. SNMP data provides aggregate traffic statistics of every network device (switch or router) in five minute increments and is typically collected because of the infeasible storage overhead required to continually collected detailed packet traces from all devices. The second data set is comprised of packet traces from five switches in one of the data centers.

Table 1 summarizes device information about the first data set and illustrates the size of each data center and the fractions of devices belonging to each layer. All these data centers follow a tiered architecture, in which network devices are organized into two or three layers. The highest and lowest layers are called the *core* and the *edge* layers. Between the two layers, there may be an *aggregation* layer when the number of devices is large. The first eight data centers have two layers due to their limited size. The remaining eleven data centers have all the three layers, with the number of devices ranging from tens of devices to several hundred of devices. Note that due to security concerns, we do not include absolute values.

Tens of Gigabytes of SNMP data was collected from all of the devices in the 19 data centers over a 10-day period. The SNMP data is collected and stored by a measurement server which polls each device using the RRDTool [6]. The RRDTool is configured to poll a device every 5 minutes and record 7 data-points for an active interface. These data-points include: inoctect (the number of bytes received), outoctect (the number of bytes sent), indiscards (the number of input packets discarded), inerrors (the number of input error packets), outerrors (the number of output error packets), and outdiscards(the number of output packets discarded). The network devices currently run version 2 of the MIB protocol [8], wherein packets can be discarded if an interface runs out of resources, such as memory for queuing. Packet discards usually indicate congestion.

The second dataset contains packet traces from five switches in one of data centers included in the first dataset. The traces were collected by attaching a dedicated packet sniffer to a SPAN port on each of the switches. The sniffer ran WinDump, which is able to record packets at a granularity of 10 milliseconds. The packet traces were collected during a 15-day period and provide fine-grained traffic information such as packet inter-arrival times.

4. EMPIRICAL STUDY

In this section, we identify the most fundamental properties of data center traffic. We first examine the SNMP data to study the link utilization and packet loss of core, edge, and aggregation devices. We then characterize the tempo-

Data-Center Name	Fraction Core Devices	Frac Aggr Devices	Frac Edge Devices
DC1	0.000	0.000	1.000
DC2	0.667	0.000	0.333
DC3	0.500	0.000	0.500
DC4	0.500	0.000	0.500
DC5	0.500	0.000	0.500
DC6	0.222	0.000	0.778
DC7	0.200	0.000	0.800
DC8	0.200	0.000	0.800
DC9	0.000	0.077	0.923
DC10	0.000	0.043	0.957
DC11	0.038	0.026	0.936
DC12	0.024	0.072	0.904
DC13	0.010	0.168	0.822
DC14	0.031	0.018	0.951
DC15	0.013	0.013	0.973
DC16	0.005	0.089	0.906
DC17	0.016	0.073	0.910
DC18	0.007	0.075	0.918
DC19	0.005	0.026	0.969

Table 1: We present information about the devices in the 19 Data Centers studied. For each Data Center, we present the total number of devices and the fraction of devices in each layer.

	Core Links	Aggr Links	Edge Links
% Used	58.88%	73.7%	57.52%
% links with Loss	3.78%	2.78%	1.6%

Table 2: This table presents statistics for the interfaces polled for SNMP data. The information is broken down according to the layer that the interfaces belong to. For each layer, we present the percent of interfaces that were utilized and the percent of interfaces that experienced losses.

ral patterns of data center traffic using the packet traces. The observations we make will help to build a realistic traffic model which can be used to evaluate various schemes for traffic engineering and data center switching.

4.1 Data Center Traffic:Macroscopic View

The devices in the data center are organized into multiple layers. Devices in different layers have different physical capabilities and path diversity. By characterizing the link utilization and packet loss of each layer, we aim to determine how the different layers will benefit from traffic engineering.

Table 2 provides a breakdown of the links across all the data centers. Roughly 60% of the core links and the edge links are actively being used. Figure 2 shows the CDF of the 95th percentile utilization of those used links (where the 95th percentile is computed over all the 5 minute intervals where the link was utilized). We find that the utilization is significantly higher in the core than in the aggregation and edge layers. This is expected since a small number of core links multiplex traffic arising from a large collection of edge links (Table 2). Link utilization is 4X lower in the aggregation layer than in the core layer, with the 95th percentile utilization not exceeding 10% for any link. Again this is expected because in our data center topologies there are nearly four times as many aggregation links as core links. Moreover, link capacities of aggregation and core links are the same in

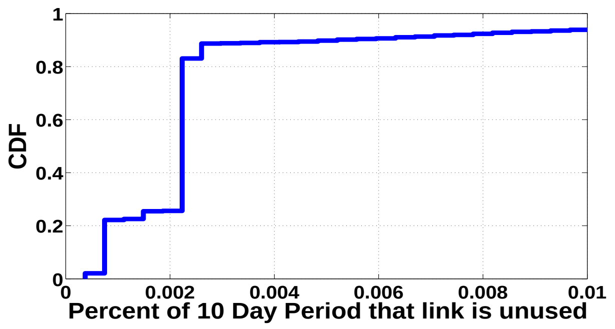

Figure 1: A CDF of the percent of times a link is unused during the 10 day interval

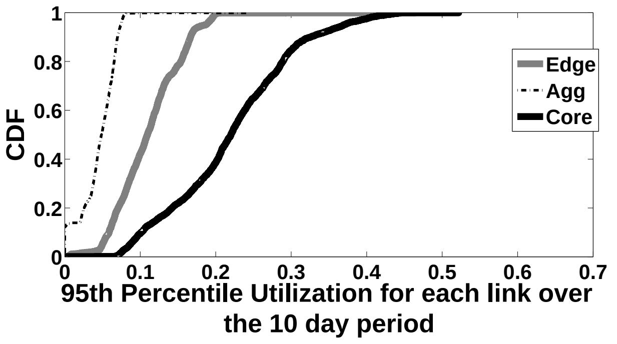

Figure 2: A CDF of the 95th link utilization at the various layers in the Data Centers Studied

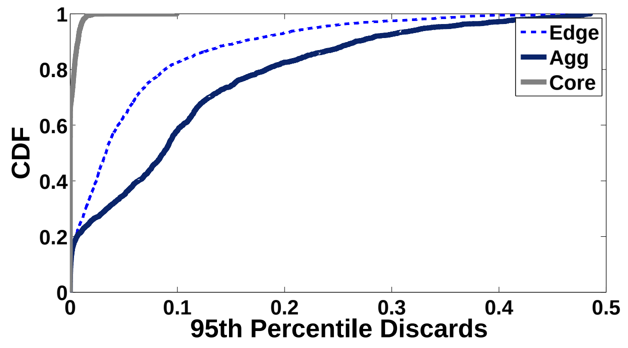

Figure 3: CDF of 95th percentile "scaled" loss rates of links at the various layers in all data center.

many cases. Edge links have slightly higher utilization than aggregate links because of their lower capacities (100Mbps at the edge vs 1Gbps or 10Gbps in the aggregation).

Figure 3 illustrates the CDF of 95th percentile "scaled" packet loss rates on the core, edge, and aggregation links. To compute actual link loss rates, we need the number of bytes discarded and the total number of input bytes. In contrast, SNMP counters only provide the number of packets discarded and the input bytes. Thus, we compute a "scaled" loss rate by converting discarded packets into bytes after scaling by an average packet size of 850B. We derived the

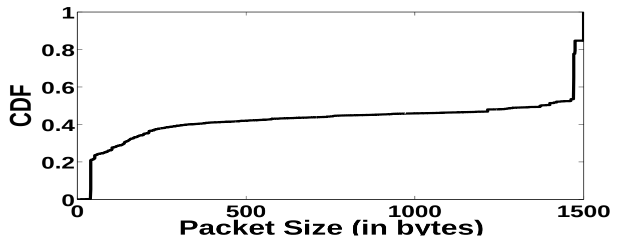

Figure 4: A CDF of the size of packets observed in DC 10

scaling factor of 850B from the observations made about the size of packets observed in DC 10. In Figure 4, we present a CDF of the packets sizes in DC 10. From Figure 4 we observe a bimodal distribution with peaks around 40B and 1470B. We observe an average packet size of 850B, we use this average packet size as the scaling factor. The bimodal distribution is likely due to the usage of layer 2 ethernet within the data centers, we similar distributions in other data centers.

The real loss rates are likely to be different; however, comparison of loss rate distributions across the three layers is likely to be the same for real and scaled loss rates. We note from Figure 3 that all layers experience certain level of losses. Surprisingly, in spite of the higher utilization in the core, core links observe the least loss rates, while links near the edges of the datacenter observe the greatest degree of losses. This suggests traffic is more bursty on aggregation and edge links than on the core links. Another important observation is that a small fraction of the links experience much bigger losses than the rest of the links. Thus, it is possible to route traffic on alternate paths to avoid most of the losses.

Given the large number of unused links (40% are never used), an ideal traffic engineering scheme would split traffic across the over-utilized and the under-utilized links. While many existing traffic engineering schemes can perform this type of load balancing, they require a relatively stable traffic matrix. Digging deeper, we examine the link idleness in one of the data centers, DC 17, and observe that although a large number of links are unused, the exact set of links that are unused constantly changed. In Figure 1 we present a CDF of the fraction of the 10 day period that each unused link is found to be idle. From Figure 1, we observe that 80% of the unused links are idle for 0.002% of the 10 days or 30 minutes. We can infer from this that although significant amounts of links are idle, the set of links that are idle in any given 5 minute interval is constantly changing. As we will show next, the traffic in the data center can be quite bursty, which accounts for the unpredictability of idle links and makes existing traffic engineering schemes less applicable.

4.2 Data Center Traffic: Microscopic View

The SNMP data only provides a coarse-grained view on the data center traffic because each data-point represents an aggregate value in a 5-minute interval. To understand lower-level details such as the burstiness of traffic and arrival patterns, more detailed information is necessary. For example, although aggregate traffic rate may be below the link capacity, a momentary traffic burst can lead to short-lived congestion in the network. To understand the properties

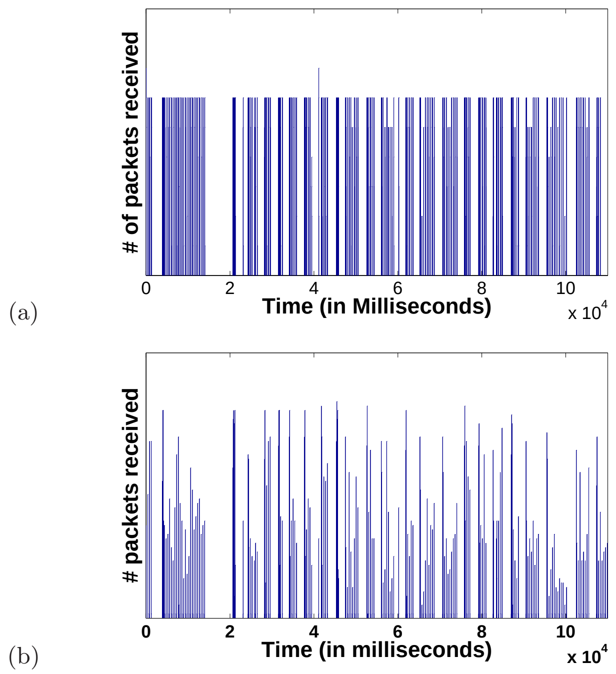

Figure 5: Pictorial proof of ON/OFF characteristics: Time series of Data Center traffic (number of packets per time) binned by three different time scales. Traffic in figure (a) is binned by 15ms and by 100ms in (b).

of such a traffic burst, more detailed information is needed. We obtained packet traces from five edge switches in one of the data centers and use this data to derive temporal traffic patterns. DC 10, the data center studied in this section, is a 2-tier corporate datacenter containing intranet server farms. DC 10 hosts several line-of-business applications (e.g. web services).

Our observations are limited by the vantage points we have, i.e., the five edge switches. As follow-up work, we plan to investigate whether these observations hold for traffic in the other parts of the network and in other data centers. The traces were collected the week of December 18, 2008. To reduce storage overhead, only the timestamp and TCP/IP header fields of each packet was stored. Figure 5 shows the time-series of the number of packets received during each short time interval at one of the switches. Clearly, the packet arrivals exhibit an ON/OFF pattern, irrespective of the granularity of the time interval. We observed similar traffic patterns at the remaining four switches as well.

Based on this observation, we use a packet inter-arrival time threshold to identify the ON/OFF periods in the traces. Let $arrival_{95}$ be the 95th percentile value in the inter-arrival time distribution at a particular switch. We define a $period_{on}$ as a longest continual period during which all the packet inter-arrival times are smaller than $arrival_{95}$. Accordingly, a $period_{off}$ is a period between two on periods. To characterize this ON/OFF traffic pattern, we focus on three aspects: i) the durations of the ON periods; ii) the durations of the OFF periods; and iii) the the packet inter-arrival times within ON periods.

Figure 6 illustrates the distribution of inter-arrival times

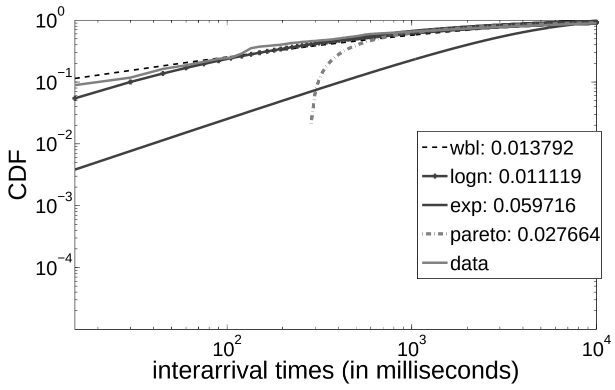

Figure 6: CDF of the distribution of the arrival times of packets at one of the switches in DC 10. The figure contains best fit curve for lognormal,weibul, pareto, and exponential as well as the least mean errors for each curve. We notice that the lognormal fit produces the least error

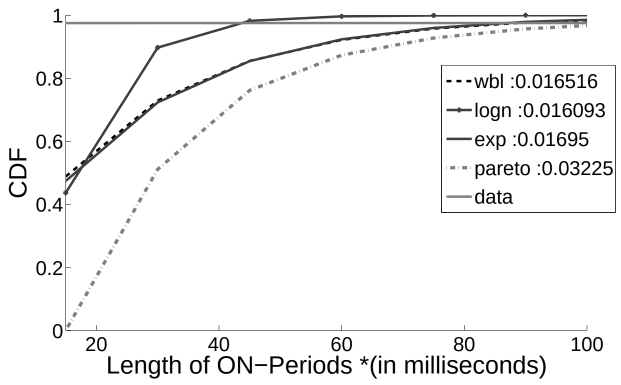

Figure 8: CDF of the distribution of the ON-Period lengths at one of the switches in DC 10. The figure contains best fit curve for lognormal,weibul, pareto, and exponential as well as the least mean errors for each curve. We notice that the lognormal fit produces the least error

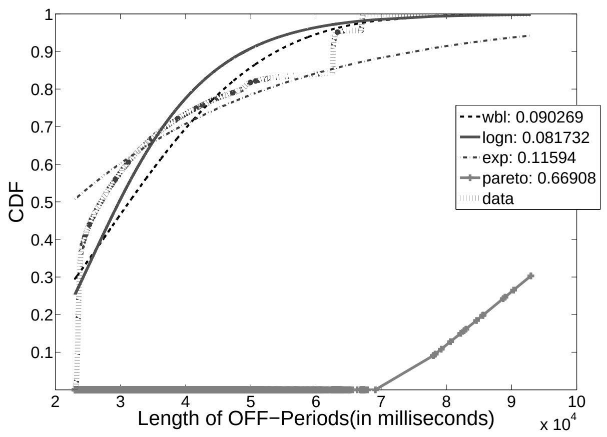

Figure 7: CDF of the distribution of OFF-Period lengths at one of the switches in DC 10. The figure contains best fit curve for lognormal,weibul, pareto, and exponential as well as the least mean errors for each curve. We notice that the lognormal fit produces the least error

within ON periods at one of the switches. We bin the inter-arrival times according to the clock granularity of 10 us. Clearly, the distribution has a positive skew and a long tail. Using Matlab, we attempt to fit the observed distribution with a number of well-known curves, such as lognormal, exponential, and Pareto and found that the lognormal curve produces the best fit with the least mean error. Figure 8 shows the distribution of the durations of ON periods. Similar to the inter-arrival time distribution, this ON period distribution also exhibits a positive skew and fits well with a lognormal curve. The same observation can be applied to the OFF period distribution as well, as shown in Figure 7.

To summarize, we find that traffic at the five edge switches exhibits an ON/OFF pattern. The durations of the ON/OFF periods and the packet inter-arrival times within ON periods all follow some lognormal distributions. In the next section, we will present our techniques for finding the appropriate lognormal random processes that can generate traffic under certain volume and loss rate conditions.

5. GENERATING FINE-GRAINED OBSERVATIONS FROM COARSE-GRAINED DATA

It is difficult for operators to instrument packet sniffing and netflow on all the devices in a data center. However, access to fine-grained data provides insight into traffic characteristics such as time-scales of congestion which can then be used to inform traffic engineering and switching fabric design. In this section, we present a framework for generating fine-grained traffic data from readily available coarse-grained data such as SNMP polls. This framework utilizes the observations made earlier in Section 4 that the arrival processes at the edge switches can be explained by 3 lognormal distributions. The framework aims to discover a set of parameters for these distributions such that the traffic data generated by the arrival process explains the coarse-grained characteristics captured by SNMP.

5.1 Parameter Discovery Algorithm

Finding the appropriate parameters for an arrival process that matches the SNMP data is a non-trivial task because it requires searching through a multi-dimensional space, where each dimension in the space represents possible values of the parameters for one of the three distributions. Since each distribution is described by 2 parameters (the mean and the standard deviation), this results in a 6-dimensional search space.

Exploring all points, or system parameters, in the search space will result in accurate results but is clearly infeasible. In what follows, we describe a new search space exploration algorithm that is both fast and reasonably accurate. Our algorithm is iterative in nature and is similar to simulated annealing — in each iteration the algorithm explores the search space by examining the neighbors of a candidate point and moving in the direction of the neighbor with the highest "score". Here, "score" is a measure of how well the parameters corresponding to the point in question describe the coarse-grained distributions.

There are four challenges in developing such an algorithm; 1) developing an accurate scoring function for each point, 2) determining a set of terminating conditions 3) defining a heuristic to avoid getting stuck in local maxima and selecting an appropriate starting point and 4) defining the neighbors of a point and selecting the next move .

Our framework takes as input the distribution of SNMP-derived volumes ($volume_{SNMP}$), and loss rates ($lossrate_{SNMP}$) for a given link at the edge of the data center. The approach returns as output, the parameters for the 3 distributions (on_{times}, off_{times}, $arrival_{times}$) that provide fine-grained descriptions of the traffic on the edge link.

To create a distribution of volume and loss, we aggregate several hours worth of data and assume that the target distributions remain relatively constant during this period.

5.1.1 Scoring Function

An effective scoring function, for deciding the utility of a point and the appropriate direction for the search to proceed in, is not obvious. To score the parameters at a point, we utilize two techniques: first, we use a heuristic algorithm to approximate the distributions of loss and volume that the parameters corresponding to the point in question generate; we refer to these as $volume_{generated}$ and $lossrate_{generated}$. Second, we employ a statistical test to score the parameters based on the similarity of the generated distributions to the input distributions $volume_{SNMP}$ and $lossrate_{SNMP}$.

Obtaining $volume_{generated}$ and $lossrate_{generated}$. We use a simple heuristic approach to obtain the loss rate and volume distributions generated by the traffic parameters ($\mu_{on}, \sigma_{on}, \mu_{off}, \sigma_{off}, \mu_{arrival}, \sigma_{arrival}$) corresponding to a given point in the search space. Our heuristic relies on the following subroutine to derive a single sample for the $volume_{generated}$ and $lossrate_{generated}$ distributions:

```
//0. calculate the mean on and off period lengths
0. $mean_{on} = \exp(\mu_{on}) + \sigma_{on}$
0. $mean_{off} = \exp(\mu_{off}) + \sigma_{off}$

// 1. determine the total on-time in a 5 minute interval
1. $total_{on} = 300 * (mean_{on}/(mean_{off} + mean_{on}))$

// 2. calculate the average number of on-periods
2. $NumOnPeriods = total_{on}/mean_{on}$.

// 3. calculate the maximum number of bytes
// that can be sent during the on period
3. $link_{capacity} = links_{speed} * mean_{on}/8$.

// 4. determine how much bytes can be absorbed by buffering
//during the off period
4. $buf_{capacity} = min(bitsofbuffering, links_{speed} * mean_{off})/8$

// iterate over on-period to calculate net volume and loss rate
// observed over the 5 minute interval
5. for $i = 0$ to $NumOnPeriods$
    a. $a_i \in A\{interarrival\ time\ distribution\}$
    b. $vol_{on} = (mean_{on}/a_i) * pktSize$
    c. $vol_{total} += min(vol_{on}, link_{capacity} + buf_{capacity})$
    d. $loss_{total} += max(vol_{on} - link_{capacity} - buf_{capacity}, 0)$
```

The above subroutine determines the loss and volume during a 5-minute interval as the sum of loss and volume in each individual on-period. Line 0 calculates the average length of an on-period and an off-period. The volume in an on-period is the sum of the bytes in the packets received, where packets are spaced based on the inter-arrival time distribution (calculated in Line 5.c). The loss in that ON-period is the number of bytes received minus the bytes successfully transmitted during the on period and the number of bytes buffered. We assume an average packet size of 1KB. In Line 5.b, the interarrival time distribution is used in the generation of a_i – each time a new value, a_i, is drawn from the distribution. The distribution A in Line 5.b is a lognormal distribution with the following parameters, $\mu_{arrival}, \sigma_{arrival}$).

We run the above subroutine several times to obtain multiple samples for vol_{total} and $loss_{total}$. From these samples, we derive the distributions $volume_{generated}$ and $loss\text{-}rate_{generated}$.

Statistical Test. We use the Wilcoxon similarity test [9] to compare the distribution of computed volumes $volum\text{-}e_{generated}$ (loss rates) against the distribution of empirical volumes $volume_{SNMP}$ (loss rates). The Wilcoxon test is a non-parametric test to check whether two different distributions are equally distributed around a mean – the test returns the probability that this check holds. The Wilcoxon test is used because unlike the popular t- or chi-test, the wilcoxon does not make any assumptions about the distribution of the underlying input data-sets. The usage of a distribution free statistical test allows for the underlying distribution for any of the 3 parameters to change.

We compute the score of a point as the minimum of the two Wilcoxon scores – the confidence measure for similarity – for volume distribution and loss distribution. We use the minimum for comparison instead of other functions such as average, because this forces the search algorithm to favor points with high scores for both distributions.

5.1.2 Starting Point Selection And Local Optimum Avoidance

To avoid getting stuck in a local optimum, we run our search a predefined number of times, N_{sub} and vary the starting points for each search. Our framework then compares the scores returned by each of the search and chooses the parameters for the search with the best score. Two key challenges are determining the number of searches to perform and determining the start point for each search.

To solve both challenges, we partition the search space into N_{sub} regions and initiate an independent search at a randomly chosen point in each sub-region. Our algorithm performs a parallel search through each of the N_{sub} regions and returns the parameters for the regions with the highest score. The choice of N_{sub} depends on the trade-off between the time consumed by the search space exploration algorithm (which deteriorates with N_{sub}) and the accuracy of the outcome (which is good for high N_{sub}). In our evaluation, we find that $N_{sub} = 64$ offers a good trade-off between speed and accuracy.

5.1.3 Neighbor Selection and Choosing The Next Move

Each point is described as a coordinate in the 6 dimension space. The natural neighbor for such a point, are the points closest to it in the coordinate space. For each point we evaluate 12 neighbors using the heuristic defined in 5.1.1.

These 12 neighbors represent adjacent points along each of the 6 dimensions.

Once all 12 neighbors are evaluated, the algorithm chooses the neighbor with the best score or randomly selects a neighbor if all neighbors have identical scores.

5.1.4 *Terminating Condition*

A search within a sub-space terminates for one of two reasons: a point with a good score ($> 90\%$) is discovered in a search sub-space, or searching in this portion of the space is futile. We deem a search sub-space futile if after 1000 iterations, the search fails to discover a suitable score ($> 10\%$). When searches in all sub-spaces terminate, the top score discovered across all sub-spaces and the parameters corresponding to the point in the space that yielded the top score are both returned.

5.2 Validation by simulation

We claim that the framework introduced in the previous section discovers parameters for an arrival process that approximates traffic at the edge switches. To verify this claim, we implement the arrival process in NS2 [1] and validate the results against data from a randomly chosen data center, DC #17. Verification of the framework consists of three steps; (a) using the framework to generate parameters that match the SNMP data for an edge switch (b) running the NS2 simulator with the parameters as input, and (c) performing a statistical test to compare the original data to the results of the simulation.

We model a edge link in NS2 as a link with the following properties: 5 MB of input buffering, FIFO queuing discipline, and a propagation delay of 548 nanoseconds. We chose these parameters based on conversations with operators of the data centers studied.

We evaluate the quality of a match by running the Wilcoxon test to compare data observed from the switch with the data generated by the simulation. To get the distributions of volume and loss rate from the simulator, we run the simulator several times to get a statistically significant number of datapoints for each distribution. In these experiments, we chose to collect 100 data points. We consider a match successful if the Wilcoxon test passes with over 90% confidence.

In Figure 9, we provide a CDF of the confidence returned by the Wilcoxon test for the validation runs on over 200 edge links in the DC #17. Our algorithm can find appropriate arrival processes for over 70% of the devices with at least a 90% confidence according to the Wilcoxon test.

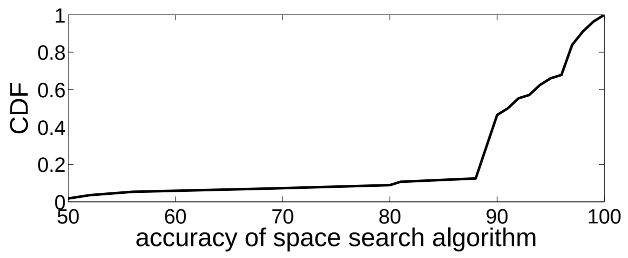

Figure 9: CDF of the validation score for each edge link simulated. Each score is a minimum of the Wilcoxon test on the volume distributions and the Wilcoxon test on the loss distributions for the switch

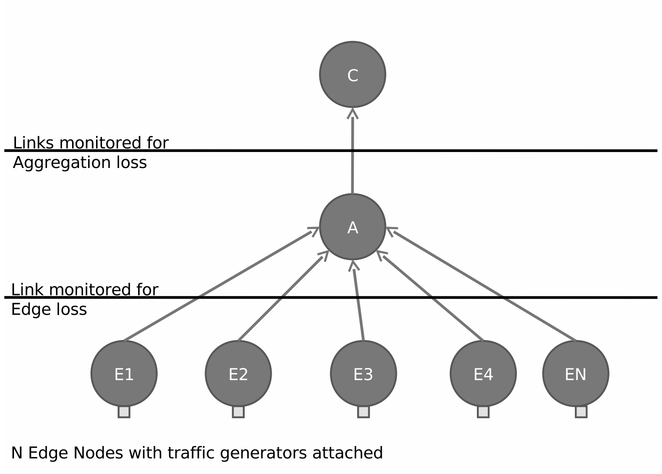
C
Links monitored for Aggregation loss
A
Link monitored for Edge loss
E1
E2
E3
E4
EN
N Edge Nodes with traffic generators attached

Figure 10: Topology simulated in deriving loss patterns.

6. APPLICATIONS

In the previous section, we presented an approach that derives fine-grained network-wide traffic characteristics that best describe coarse-grained network-wide performance data. The traffic characteristics are modeled upon inferences drawn from a limited amount of low-level information collected at a few instrumented locations. The framework we devised has a few important practical applications aside from supporting our empirical observations about data center traffic patterns.

6.1 Informing data center traffic engineering

Current approaches to traffic engineering in data-centers borrow heavily from well established traffic engineering techniques in Wide Area Networks. For example, edge devices are equipped with ECMP and MPLS both of which employ static, slowly-adapting techniques to map flows to routes. Our framework can be used to examine the effectiveness of such techniques to a given data center network. In particular, an operator could use SNMP data collected in his network to derive the best-fitting parameters for application traffic flows. If the derived app-level traffic is not too bursty and hence, reasonably predictable, then slowly-adapting techniques such as the current ones in use may be sufficient. If on the other hand the traffic is too bursty, then our approach can indicate the granularity at which re-routing decisions must be made in the network.

In what follows, we use inferences regarding traffic sources in one of the data centers to examine the loss rate patterns on network links in the data center. We use the ns2 setup used for the validation step earlier, however instead of focusing on the traffic out of each edge switch in isolation, we allow data from several edge switches to interact at a switch in the aggregation layer.

We use the simulation to study time-scales of losses at the aggregation switch. To do this, we define the notion of "micro-burst losses". Each microburst is a group of consecutive packets with the following three properties: (1) The group starts and ends with a packet drop (2) There are fewer than X consecutive packets that are received successfully in the group, for some constant X. Thus, the instantaneous packet drop rate within a micro-burst is $\geq 2/(2 + X)$. And (3) each microburst should have a certain minimum number of packets in total – we arbitrarily set this to 100.

X	Min loss rate during microburst	Fractions of drops outside microbursts
5	30%	10%
10	16%	3%
15	12%	1.6%

Table 3: Existence of burst losses.

We first study how often losses happen in such bursts, or equivalently, how many losses are isolated or not part of a microburst. The results are shown in Table 3. We find only a small fraction of losses do not belong to any microburst. This indicates that, more often that not, when losses happen at the edge or aggregation links, they happen in bursts.

We now study the length of microbursts, defined as the difference in the timestamps of the packet losses at either boundary. In Figure 11 we show a distribution of the lengths of the microburst losses at the aggregation switch for $X = 5, 10, 15$. We see that an overwhelming fraction of microbursts last less than 10s. Dynamic load balancing and rerouting could help avoid losses due to microbursts. However, these initial results indicate that, to be effective, they must take rerouting/load balancing decisions at the granularity of once every few seconds.

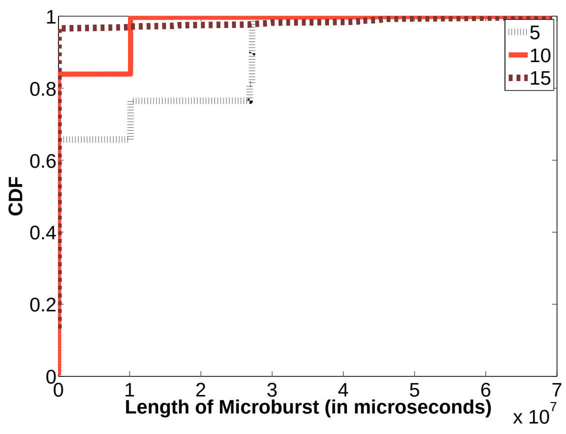

Figure 11: A CDF of the length of the microbursts (in microseconds).

6.2 Traffic generators for data centers

Several recent studies have examined how to design scalable data center topologies. To evaluate the proposed topologies, these studies have relied on experiments that use "toy" traffic workloads such as mixtures of long and short-lived flows. It is unclear if these workloads are representative of network traffic patterns in data centers. Other studies have employed application-specific workloads such as map-reduce or scientific computing and used the resulting traffic traces for evaluating data center network designs and management approaches. However, network traffic in data centers is composed of traffic from a variety of other applications with diverse send/receive patterns.

Our empirical observations and the framework we presented in the previous section could help in the design of realistic network traffic generators for data centers. As part of current work, we are developing such a traffic generator. Our generator has several tuneable parameters such as the size of the data center, the number of servers and the interconnection topology. All servers generate on-off traffic where the parameters of the traffic are drawn at random from the traffic models that we have derived from the 19 data centers examined in this study. Rather than requiring users to simulate popular data center applications like Map-Reduce and scientific computing, our simulator directly generates representative network traffic that is a result of multiple data center applications running together.

7. CONCLUSION AND FUTURE WORK

In this paper, we studied traffic patterns in data center networks at both macroscopic and microscopic scales. Using network-wide SNMP data collected at 19 data centers, we found that links in the core of data centers are more heavily utilized on average, but those closer to the edge observe higher losses on average. Using a limited set of packet traces collected at a handful of datacenter switches we found preliminary evidence of ON-OFF traffic patterns.

A key contribution of our work is a new framework for deriving likely candidates for parameters that define the sending behaviors of data center traffic sources. Interestingly, our approach relies on network-wide coarse-grained data that is easily available and combines it with high-level information derived from fine-grained data that is collected at a small number of instrumented locations. This general framework can be used to examine how to design traffic engineering mechanisms that ideally suit the prevailing traffic patterns in a data center.

As part of future work, we plan to develop new fine-grained traffic engineering mechanisms that are more responsive for data center traffic than traditional wide-area approaches. We are also developing a traffic workload generator for datacenter traffic that is motivated by our empirical observations.

8. ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their valuable feedback. This work was supported in part by an NSF CAREER Award (CNS-0746531) and an NSF NeTS FIND Award (CNS-0626889).

9. REFERENCES

[1] The network simulator - ns-2. http://www.isi.edu/nsnam/ns/. *http://www.isi.edu/nsnam/ns/*.
[2] M. Al-Fares, A. Loukissas, and A. Vahdat. A scalable, commodity data center network architecture. In *SIGCOMM*, pages 63–74, 2008.
[3] N. Feamster, J. Borkenhagen, and J. Rexford. Guidelines for interdomain traffic engineering. *SIGCOMM Comput. Commun. Rev.*, 33(5), 2003.
[4] A. Greenberg, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta. Towards a next generation data center architecture: scalability and commoditization. In *PRESTO '08: Proceedings of the ACM workshop on Programmable routers for extensible services of tomorrow*, pages 57–62, New York, NY, USA, 2008. ACM.
[5] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson. On the self-similar nature of ethernet traffic (extended version). *IEEE/ACM Trans. Netw.*, 2(1), 1994.
[6] T. Oetiker. Round robin database tool. http://oss.oetiker.ch/rrdtool/.
[7] V. Paxson and S. Floyd. Wide area traffic: the failure of poisson modeling. *IEEE/ACM Trans. Netw.*, 3(3):226–244, 1995.
[8] C. S. version 2 MIBs. http://www.cisco.com/public/mibs/.
[9] F. Wilcoxon. Biometrics bulletin. 1:80–83, 1945.