Cost Analysis of Data Centers with High Availability

Sofia Salgueiro
sofia.salgueiro@tecnico.ulisboa.pt
Instituto Superior Técnico, Lisboa, Portugal
Januray 2021

Abstract—**With the growth of cloud computing, data centers are increasingly being used to host applications and data of external companies. As such, data center companies must guarantee that the services hosted in their data centers are available most of the time with minimum latency. Nonetheless, failures in the services offered by data centers lead to the loss of revenue of clients and consequent reimbursements from data center companies. We studied how a data center network with hundreds of thousands of servers can be built to minimize the impact of link failures and, at the same time, its cost. We assumed data centers use folded Clos topologies and implement the Border Gateway Protocol. We identified the characteristics of the different routers used in data center networks and estimated their cost with the main goal of comparing the acquisition cost of routers of different topologies. We analyzed, through mathematical expressions and algorithmic computations, the impact of link failures in the connectivity between pairs of servers and between the servers and the Internet. We studied the cost also considering traffic and the congestion in links in the presence of link failures. To do this, we assumed traffic patterns of three applications: distributed applications with communications only internal to the data center, search engines and streaming services. Finally, we were able to provide answers on how to build the minimum cost network that provides the availability of services offered by data centers.**

1. Introduction

Data centers comprise hundreds of thousands of servers interconnected by thousands of routers in a single network [1]. Although data centers may be used to house resources of the owning company, the data centers we focus on are used to lease resources, in the form of, for example, data storage, virtual machines and full servers, to external clients. This model is known as cloud computing [2] and it is used by private costumers and companies. Therefore, each data center may house a large number of different applications: machine learning models that can take days to train [3], websites of enterprises, online stores, search engines, streaming services, etc...

However, when there are failures in data centers, which may result in delays or unavailability, that is, resources are unreachable, clients that use data centers to host their resources lose revenue. For example, a study [4] found that, for Amazon, an increase in latency of 100 ms resulted in a loss revenue of 1%.

Since delays and unavailability can cause a significant loss in revenue, Service-Level Agreements (SLAs) are signed between data center companies and clients. SLAs have demanding characteristics: for example, Azure guarantees that, when there are two instances of a virtual machine in different data centers, the client can access at least one instance 99.99% of the time in a month. It should be noted that this value of minimum availability translates to a downtime of, at most, 4 minutes. Therefore, when there are failures in data centers, not only do clients lose money but data center companies lose as well, in addition to being detrimental to their reputation and make them lose actual and prospective clients.

In this work, we focus on minimizing the cost of data centers for availabilities usually specified in SLAs. To do this, we assume that the topology is a 3-tier folded Clos network, and that Border Gateway Protocol (BGP) is the routing protocol used. We study two different aspects that have an impact on the availability of a network: link failures and congestion.

2. An overview of data center networks

We start with a brief overview of the most common network topology and routing protocol used in data centers.

2.1. Clos network topology

Data center companies want to minimize the risk of unavailability and one way to do this is to have many redundant paths. So far, Microsoft [5] and Facebook [6] data centers are using extensions of Clos topologies. Clos topologies can be designed to provide a high number of shortest disjoint paths between pairs of servers and between the data center and external endpoints. Additionally, its modular and repetitive structure simplifies the physical implementation of the network.

In this work, we are studying 3-tier folded Clos toplogies, as depicted in figure 1. The T_0s, or Top of Racks (ToRs), due to being placed at the top of rack cabinets, make the bottom-tier routers of this architecture and every one of them connects to every router in the adjacent top-tier, designated T_1s. T_0s and T_1s are grouped in **clusters** and the T_2s provide the connection between clusters. The T_2s are grouped in *spine planes* and the first T_1 of each cluster connects to all the T_2s of the first spine plane, the second T_1 of each cluster connects to all the T_2s of the second spine plane, and so on. This network provides redundancy, i.e. there are multiple paths between every pair of servers. A pair of servers from the same cluster can connect through many T_1s with paths of the form T_0 - T_1 - T_0, and servers of different clusters can connect through many T_1s, and each T_1 has many T_2s to choose from, with paths of the form T_0 - T_1 - T_2 - T_1 - T_0.

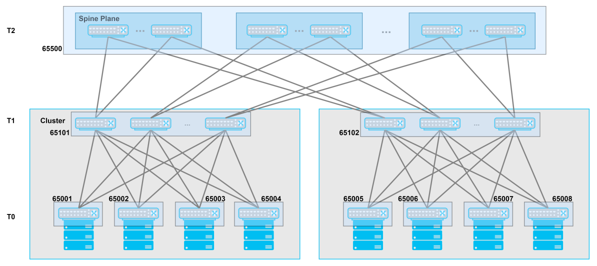

Figure 1. A 3-tier folded Clos network and its assignment of Autonomous System Numbers.

2.2. Border Gateway Protocol (BGP)

Routing protocols are responsible for electing the best routes, given metrics specific to each protocol, and advertise these routes to the other routers in the network. Therefore, all routers automatically learn how to reach each network prefix and, each time the preferred route of a router changes, the other routers are informed of this change. This way, every alteration in the network is propagated to all routers. The routing protocol is also responsible for determining the number of redundant paths and, if possible, to distribute the traffic by the best paths available.

BGP, Border Gateway Protocol, is the routing protocol used in the Internet [7]. It has many stable and robust implementations and, additionally, supports different extensions [8], such as *multi-path* [9], which allows routers to register multiple paths to reach a destination. Additionally, it is implemented in most routers available in the market and there are people specialized in it. These advantages contributed to the first implementation of BGP in the data center by Microsoft [8]. Today, it is also chosen by many other companies that have data centers, such as Facebook [6], [8].

In data centers, BGP configurations should make sure a pair of servers are always connected by the shortest path and, if possible, traffic should be distributed by all the shortest paths available. *Jayaraman et al.* [5] provide, in their work, an example of a BGP adaptation to a folded Clos network of a Microsoft data center. Given the well-defined structure of a folded Clos network, the shortest paths between servers are known and BGP was implemented to take advantage of this, specifically with the following assignment of Autonomous System Numbers (ASNs):

- every T_0 has its own ASN.
- all the T_1s inside a cluster have the same ASN.
- all the T_2s of the data center have the same ASN.

Figure 1 is an example of this numbering scheme for a network with 2 clusters and 4 T_0s per cluster. This numbering scheme implies that, due to the loop detection in BGP through its *AS-PATH* attribute, each T_0 only keeps the shortest paths in both the Routing Information Base (RIB) and the Forwarding Information Base (FIB).

The shortest paths are of length 2 to other T_0s of the same cluster and of length 4 to a T_0 of another cluster. An example of a path of length 2 between two T_0s is 65001 - 65101 - 65002, where 65001 and 65002 identify the first and second T_0s of the first cluster, respectively, and 65101 can be any of the T_1s of the first cluster. An example of a path of length 4 between two T_0s of different clusters is 65001 - 65101 - 65500 - 65102 - 65006, where 65101 is any of the T_1s of the first cluster, 65102 is any of the T_1s of the second cluster and 65500 is any of the T_2s.

By using a BGP extension called *multi-path* [9], [10], it is possible to install many paths of equal cost in the FIB, responsible for the fast lookup of the next-hops of each prefix. In the case of the network in figure 1, the cost of each route is determined by the *AS-PATH* length and, consequently, routes with the same length have the same cost. Therefore, each router is able to store every shortest path available for each prefix. Then, *multi-path* uses Equal-Cost Multi-Path (ECMP) to distribute the traffic by the routes installed in the FIB, in order to achieve load-balancing in terms of traffic flows. This way, although a traffic flow is indivisible, different traffic flows can be distributed among different paths. For example, if there are no link failures, router 65001 can use every T_1 of its cluster to send packets to router 65002. This not only minimizes the load of each link used in the paths but also increases the bandwidth available between the servers. If, however, we want to limit the number of routes for each prefix FIB, in order to diminish the memory needed, it is possible to be configured.

3. Network cost estimation and analysis

In this study, we are only analyzing acquisition cost (Capex) of the network. Thus, we differentiate the acquisition cost of each topology by the cost of the routers. In the case of the operational cost (Opex), we calculate the power utilized by the routers and assume that the cooling equipment needed for the network will use a proportional cost. Thus, even though the cooling power is not fixed, it is sufficient to calculate the power used by the routers to analyze and evaluate the different networks in terms of operational cost.

3.1. Acquisition Cost

Starting with the cost of routers, the ones used in data centers have characteristics such as:

- *Low latency* - latency in routers is the time that it takes for a router to start sending a packet after receiving it. In the case of data center routers, it should be in the order of nanoseconds.
- *High throughput* - data center routers have ports of 10, 40, or even 100 Gbps.
- *High forwarding capacity* - forwarding capacity represents the number of minimum sized Ethernet packets a router can switch per second. In data centers, it can go as high as billion packets per second (Bpps).
- *High number of ports* - in order to support hundreds of thousands of servers, routers used in data centers can have as much as 200 ports.

With these demanding characteristics, routers used in data centers are expensive and building an entire data center requires thousands of them. Some companies, such as Facebook [6], even started building their own routers. In this study, we use the cost of the Juniper routers provided in [11] to approximate the prices.

The cost of routers depends on many factors:

- **Memory size** - the Ternary Content-Addressable Memory (TCAM) is used to store the FIB and Access Control Lists (ACLs) [12], [13], which are rules that tell if packets from specific Internet Protocol (IP) addresses or prefixes should be forwarded or discarded. The RIB, responsible for saving all paths learned, is stored in the Dynamic Random-Access Memory (DRAM). [14], which is hundreds of times cheaper than the TCAM [15].
- **Number of and capacity of each port** - routers have a set of physical ports with a specific maximum capacity. However, it is possible to have different port combinations for the same router.
- **Forwarding capacity**
- **Modularity -** there are two types of routers in terms of structure: fixed routers, with a fixed number of ports, and modular routers, which are composed of a chassis and multiple line cards. The chassis functions as a cabinet that also provides the backplane, the power source and the ventilation. The line cards are inserted in the chassis, resembling drawers, and each one has a set of ports and its own FIB table. The backplane is responsible for the management of the FIB in each line card, so that they all share the same information, and for the switching between line cards.

In order to understand which characteristic has a bigger influence in the cost of these routers, we analyzed the approximated prices of some routers, taken from [11], along with the number of supported FIB (IPv4) entries and forwarding capacity, as provided in datasheets. We concluded that, since routers with different forwarding capacities and number of ports, but the same number of FIB (IPv4) routes have a similar cost, the memory is what more influences the cost.

Routers in the two bottom tiers, T_0s and T_1s, have similar characteristics and are used in clusters, the base unit of folded Clos topologies. Therefore, these routers are not usually used to scale the network and are responsible for switching the traffic of a limited number of servers. Consequently, we are using fixed routers for T_0s and T_1s, i.e. routers with a fixed number of up to 96 ports.

Due to the fact that routers in the two bottom tiers, T_0s and T_1s, have similar characteristics, we set the cost of these routers independently of the number of ports. For port speeds of 10 Gbps, we used the Juniper QFX5200-32C router, configured as a router with 128 ports of 10 Gbps, to set the cost at \$20 000. For T_0 and T_1 routers with 40 Gbps ports, we used the Juniper QFX5210-64C router, with a configuration of 96 ports of 50 Gbps, to set the price of T_0 and T_1 routers with 40 Gbps ports at \$30 000 each.

However, T_2 routers have different and more demanding characteristics when compared to T_0s and T_1s. Firstly, since T_2s connect the data center to the Internet, they need to know routes imported from all over the Internet and, consequently, need bigger FIBs. Additionally, since T_2s connect clusters to each other and the Internet, they have a faster backplane, when compared to T_0s and T_1s, in order to provide a higher forwarding capacity.

Due to these characteristics, for T_2s, we used modular routers, the Juniper chassis QFX10008, that supports 8 line cards, and QFX10000-30C line cards. These line cards have 30 physical ports and two possible configurations that interest us, 96 ports of 10 Gbps or 30 ports of 40 Gbps. Therefore, each T_2 has the fixed cost of the chassis QFX10008, which is, approximately, \$60 000, and the variable cost of the number of line cards used, at a price of \$90 000 per line card.

The prices of routers are summarized in table 1.

TABLE 1. PRICE OF EACH ROUTER.

Throughput	Cost of each router (\$)	
	T_0s and T_1s	T_2s
10 Gbps	20 000	$60\ 000 + 90\ 000 \times \left\lceil \frac{nrofports}{96} \right\rceil$
40 Gbps	30 000	$60\ 000 + 90\ 000 \times \left\lceil \frac{nrofports}{30} \right\rceil$

In graph 2, we present the cost per port of each type of router. By comparing the cost per port, it possible to see that T_0s and T_1 present a lower cost than T_2, due to the different characteristics required in both types of routers. We can conclude, by analyzing the graph that, by setting a fixed cost of T_0s and T_1s, the cost per port is inversely proportional to the number of ports. In the case of T_2s, the cost per port also diminishes with the number of ports. This is because the cost has a fixed part, corresponding to the chassis, and a variable cost corresponding to the number of line cards used, that is, the number of ports. Thus, T_2s with a higher number of ports distribute the cost of the chassis by more ports and, consequently, the cost per port decreases.

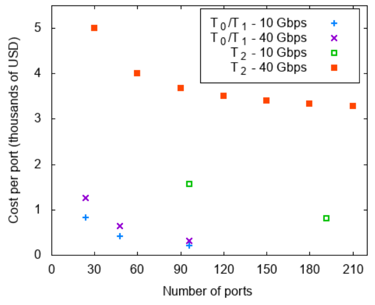

Figure 2. Cost per port of each type of router used in the network.

In order to analyze the cost of networks with a different number of servers, in terms of routers, we fixed a cluster of approximately a thousand servers. We used T_0s of 48 ports, 40 servers per T_0, 8 T_1s per cluster, and 24 T_2s per spine plane. Thus, we started with a single cluster and 24 T_2s and, using the same T_2s, we kept adding clusters to find the cost per server of each network. In figure 3, we can see that the cost per sever tends to diminish with the increase of the number of servers. It can be concluded that this tendency is due to the evolution of cost of the T_2s with the number of ports.

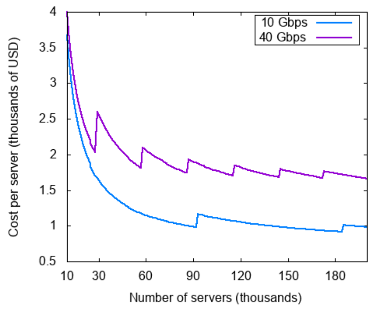

Figure 3. Cost per server of networks.

When we want to add more servers, which translates, in this case, to adding more clusters, we have two options, either there are available ports in T_2s that we can use, or we have to add a line card to each T_2 in order to accommodate the new cluster. Thus, when we add a line card to each T_2, the cost per server rises abruptly due to the additional cost of the line card. However, as we keep adding clusters to the free ports of the line card previously inserted, the cost of the T_2s gets distributed by a bigger number of servers, resulting in a lower cost per server. This can be seen in figure 3, in which the peaks represent the added cost of acquiring more line cards.

Additionally, it can be seen that, as the network grows, the difference between the cost per server diminishes and networks with 40 Gbps links are, at most, 2 times more expensive than networks with 10 Gbps links.

Nonetheless, it is possible to reduce the number of T_2s and, consequently, the total cost of the network, while maintaining the number of links between routers. We can do this by using fewer T_2s and employing parallel links between the same pair of routers T_1s-T_2s, while maintaining the total number of links between a T_1 and all T_2s. For example, if a T_1 has a single link to 24 T_2s, it may be possible to connect it to 12 T_2s and, instead of a single link per T_2, make use of two links in order to maintain the total number of links to T_2s. Since, without parallel links, each T_2 has one connection per cluster, as long as it is possible, either by using free ports or adding line cards, to accommodate 2, 3, or more, connections per cluster, then we can use parallel links. By using parallel links, we eliminate the fixed cost of the chassis of the routers that will not be used and it may even be possible to reduce the total number of line cards used since we may be making use of available ports, and, consequently, the cost of the network will decrease.

In order to evaluate the savings allowed by using parallel links, we developed a program in python that generates all possible topologies for a given number of servers and the number of ports of each router. In order to simplify the execution of the algorithm, we generate topologies using routers with a fixed number of ports. For T_0s and T_1s, the number of ports we use are 24, 48 and 96. For T_2s we use 24, 48, 96 and 192 ports.

We ran the algorithm for networks with 10 000, 50 000, 100 000 and 150 000 servers. Then, in figure 4, we compare the cost of the cheapest networks, with the same level of resilience to link failures, that make, and do not make, use of parallel links.

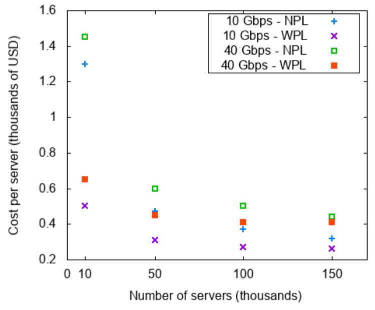

Figure 4. Cost per server of networks with and without parallel links. NPL - No parallel links. WPL - With parallel links.

From figure 4, it can be concluded that networks with parallel links can cost up to 61% less in networks of 10 Gbps links and up to 55% less in networks of 40 Gbps links. The relative savings are higher in the smaller networks due to the cost of the chassis of T_2s being distributed by fewer servers.

3.2. Operational Cost

To obtain the operational cost, we analyzed the routers previously used to define the acquisition cost. In each router, there are two power limits, the minimum power is used when the router is idle and the maximum power consumed occurs when the traffic load is maximum. In [11] it is said that the power used varies linearly with the number of ports in use and the traffic load. Thus, in our calculations we assumed a traffic load of 50% with half of the ports in use.

Similarly to what was done in the acquisition cost, we assume that both T_0s and T_1s consume the same, 389 W in the case of routers with 10 Gbps per port and 688.5 W when using routers with 40 Gbps ports. Regarding the T_2s, similarly to what was done to calculate their cost, the total power is the sum of the power consumed by the chassis, 1483 W, and the energy consumption of each line card, 891 W.

In order to compare the power consumed by networks of different sizes, that is, with a different number of servers, we executed the same process that resulted in figure 3. The power per server of the different networks created can be seen in figure 5.

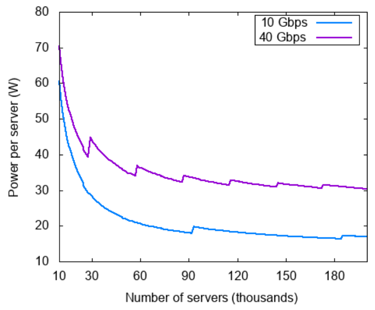

Figure 5. Power per server of different networks.

Similar conclusions to the ones drawn from figure 3 can be derived from figure 5. When we add more clusters to the network, two different situations occur: if we add them to available ports then the overall power of the network can be distributed by more servers and, consequently, the power per server decreases, but, when we need to add more line cards in order to add clusters, the power by server suddenly increases due to the added power of the new line cards. Additionally, we can also conclude from the graph that networks with 40 Gbps links only use up to 2 times more power than 10 Gbps networks.

We also compared the power consumed when using parallel links between T_1s and T_2s and concluded that savings go from, approximately, 15% to 58%, in the case of networks with links of 10 Gbps, and from 7% to 48% in networks with 40 Gbps links. Additionally, the relative savings are, once again, higher in the smaller networks due to the larger impact that the power consumed by the chassis has when distributed by fewer servers.

4. Connectivity Analysis

There are different applications running in data centers and, consequently, the definition of availability in SLAs will differ from application to application. For example, in distributed applications, such as the training of a machine learning model, the availability refers to paths between servers inside the data center. However, when clients are trying to access virtual machines in the data center, the concept of availability is applied to connections between the data center servers and the Internet. Therefore, we study the impact of T_0-T_1 and T_1-T_2 link failures in three different types of connections: between servers of the same cluster, between servers of different clusters and between a server and the Internet.

4.1. Connectivity inside a cluster

In data centers, T_0s and T_1s only connect to other T_1s and T_0s, respectively, of the same cluster. A T_0 connects to every T_1 of the cluster and a T_1 connects to all T_0s of the same cluster, as depicted in figure 1. If k is the number of routers T_0 inside a cluster and m is the number of T_1s inside a cluster, we can conclude that there are m paths, **pairwise disjoint** and all of **length 2**, connecting two T_0s **of the same cluster**.

Thus, to connect 2 T_0s of a cluster via a T_1, as in green in figure 6, both links connecting the T_1 to the T_0s must be available. Since there are m T_1s, it only takes one link failure per T_1, either to the source or destination, to disconnect 2 T_0s, which totals m failures. Additionally, if x is the probability of a link failure between a T_0 and a T_1 and assuming that each failure is independent, the probability of 2 T_0s being able to connect with each other is given by

$$P(connectivity) = 1 - (1 - (1 - x)^2)^m. \qquad (1)$$

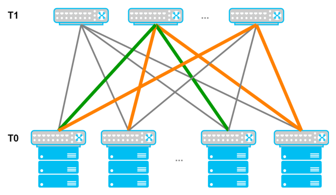

Figure 6. In green, a path of length 2, that connects two ToRs and, in orange, a valley shaped path of length 4.

We are now able to approximate the availability defined in SLAs by the probability of connectivity. In this case, we are studying connectivity between T_0s, which is more important in applications that rely on connections between servers, such as distributed applications.

It is also possible to use non-shortest path routing, which we designate by *valley* paths as a work around to increase availability. Valleys are paths with an up-down-up-down shape that use more than a single T_1 to connect two T_0s, as can be seen in orange in figure 6. It is also worth mentioning that, when two servers of different clusters communicate with each other, only valleys inside clusters are permitted, that is, a path is not allowed to go through T_2s twice.

In order to understand the importance of valleys, we examine now how many link failures it takes for two T_0s **of**

the same cluster to be unable to connect with each other when valleys are allowed. If we can connect 2 T_0s through a single T_1, then there is a T_1 that has active links to both T_0s. Thus, if all T_1s only have a link to the T_0 source or the destination, they can no longer connect, unless we use valleys.

When we use valleys to connect 2 T_0s, we connect them through an intermediate T_0 that has active links to at least 2 T_1s, one that has an active link to the source and another one that has an active link to the destination. Therefore, in order to disconnect 2 T_0s, even if the option of valleys is allowed, each intermediate T_0 must only be able to either connect to T_1s that can only connect to the source, or to T_1s that only have a link to the destination. Thus, if a T_0 can only connect to T_1s that only have active links to the destination, then the T_0 is connected to the same T_1s that the destination is and, consequently, has the exact same link failures that the source does.

If the source is a and has x link failures, the destination is b with, consequently, $m - x$ failures since there needs to be one failure per T_1, and, finally, every remaining T_0 has, at least, the same failures that a or b have, then **the minimum number of failures that completely disconnect 2 T_0s of the same cluster is:**

$$m + \min(x, m - x) \times (k - 2). \qquad (2)$$

This expression can take values considerably higher than the m failures it takes to disconnect two T_0s without the use of valleys and, consequently, allowing valleys should reflect an increase in the maximum probability of link failures that support 99.999% of connectivity.

In order to evaluate the impact of valleys in connectivity between T_0s, we developed a C++ program and implemented algorithm to find the shortest paths between 2 nodes, which are, in this case, T_0s.

To analyze the connectivity between T_0s of a cluster, with and without valleys, we chose a network composed of a single cluster of 48 T_0s and, for each value of m, i.e., T_1s per cluster, ran the algorithm for a given probability of failure of T_0-T_1 links. Then, we evaluated the average of connectivity between T_0s and kept running the algorithm, increasing the probability of failure, in order to find the maximum probability of failure that supports 99.999%, or 5x9, of connectivity. We did this for a thousand samples for each probability, in order to increase the degree of confidence. The results are represented in figure 7, as well as the theoretical results provided by equaling equation 1 to 99.999%.

As can be seen in figure 7, the **use of valleys increases the resilience to link failures of a network without needing to upgrade its equipment**. It is also especially relevant for low values of number of T_1s per cluster. For example, for 8 T_1s per cluster, the maximum probability of a T_0-T_1 link failure increases from 12% to 26% by using valleys.

It could be possible to increase the resilience to link failures by making use of even longer paths and, consequently,

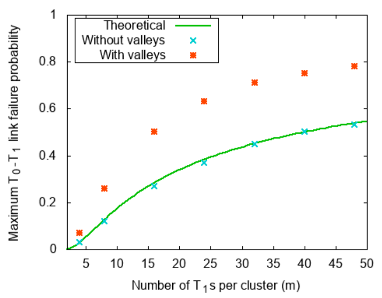

Figure 7. Comparison between the maximum probabilities of a T_0-T_1 link failure that a network supports, for 99.999% connectivity between T_0s, and for different values of T_1s per cluster, m, with and without the use of valleys.

allowing more valleys. However, a single valley already provides a significant number of paths and allowing more valleys would not be as advantageous. Additionally, using valleys also has some considerations that must be taken into account. Firstly, in the T_1s, the size of the RIB and the FIB grows, the latter only in the presence of failures. Secondly, valley paths are longer and use more routers than non-valley paths, which results in paths with higher latency. Finally, when using valleys we are no longer using only disjoint paths and, consequently, when distributing the traffic, the links in common of these paths may end up with a load higher than their capacity.

4.2. Connectivity between clusters

After studying clusters as a unit, we now need to study connectivity between T_0s of different clusters.

Dismissing the option of valleys, if we only consider link failures between T_0s and T_1s, the probability defined in equation 1 still stands. That is, the probability of two T_0s being able to connect with each other, whether or not they are in the same cluster, in the presence of T_0-T_1 link failures is given by equation 1.

Now, if we assume that only links between T_1s and T_2s can fail, there needs to be $m \times n$ link failures to disconnect two T_0s of different clusters. We can determine that, with y being the probability of a T_1-T_2 link failure, the probability of 2 T_0s of different clusters being able to connect is given by

$$P(connectivity) = 1 - (1 - (1 - y)^2)^{m \times n}. \qquad (3)$$

This probability is dependent not only on m, the number of T_1s per cluster, but also on n, the number of spine sets. By comparing equations 1 and 3, it was possible to conclude that, for the same values of probability of connectivity and for the same number of T_1s per cluster, for $n > 1$, the maximum probability of a T_1-T_2 link failure is higher than a T_0-T_1 link failure.

As previously mentioned, it is cheaper to use parallel links between the same T_1-T_2 pair in order to use less T_2s.

In this case, even if the number of connections that a T_1 has to T_2s is the same, the probability of connectivity differs when we use parallel links.

In the case of parallel links, a T_0 can connect to a T_0 of another cluster through two specific T_1s **as long as there is a single active link** from each T_1 to the T_2. Therefore, if there are b links between a T_1 and a T_2, n spine sets and x is the probability of a T_1-T_2 link failure, then the probability of two T_0s being able to connect to connect with each other is given by

$$P(connectivity) = 1 - (1 - (1 - x^b)^2)^{n \times m}. \qquad (4)$$

Given both equations 3 and 4, it is possible to conclude that there is a higher resilience to failures when using parallel links. However, it should be noted that we are only analyzing **link failures**. If we were considering router failures, for the same number of connections, a T_2 router failure would bring down more paths when using parallel links than when not using.

4.3. Connectivity to the Internet

Data center companies also want to maximize the availability of external access to the data center. In order to analyze the connectivity of T_0s to the Internet in the presence of link failures, we first assume that T_1-T_2 links are perfect and evaluate the probability of connectivity in the presence of T_0-T_1 link failures. A T_0 cannot connect to the Internet, in the presence of T_0-T_1 link failures, if any of the m links it has to T_1s fails. Thus, if x the probability of a T_0-T_1 link failing, then the probability that a T_0 can connect to the Internet is given by

$$P(connectivity\ to\ the\ Internet) = 1 - x^m. \qquad (5)$$

As can be concluded, these values are higher than the ones achieved in previous sections. For example, for $m = 8$ and a connectivity of 5x9, the maximum probability of failure that supported connectivity between a pair of T_0s is, approximately, 12 % but, in this case, it is 23%.

Now, we assume that T_0-T_1 links are perfect and that only T_1-T_2 links suffer failures. In this scenario, all $m \times n$ T_1-T_2 links must fail to disconnect a T_0 from the Internet. If y is the probability of a T_1-T_2 link failing, then the probability of a T_0 connecting to the Internet is

$$P(connectivity\ to\ the\ Internet) = 1 - y^{m \times n}. \qquad (6)$$

Finally, we can also compare the cost of supporting different levels of availability, that is, the probability of a T_0 being able to connect with another T_0 or with the Internet. Therefore, we are able to calculate the cheapest network that, for example, supports, at least, 10% of the T_0-T_1 links down while achieving an availability of 5x9.

In figure 8, we analyze the cost per server of networks with 3 levels of availability, 3x9, 4x9, and 5x9. Additionally,

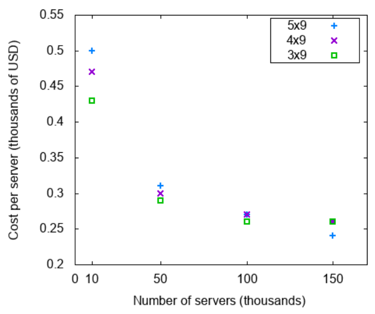

Figure 8. Network that supports 10% of T_0-T_1 link failures and 60% of T_1-T_2 link failures.

we compare two networks, one that supports 10% of T_0-T_1 link failures and 60% of T_1-T_2 link failures.

It is possible to determine, by these graphs, that the difference between the cost per server of the different availabilities diminishes with the increase of the number of servers.

4.4. Path diversity

Folded Clos topologies have many redundant paths between the different nodes. In data centers, we want to take advantage of this characteristic and use different paths to reach the same destination. This way, we can distribute the traffic by many paths and, consequently, enable load-balancing in the network. In BGP, there is an extension called *multi-path* that allows the installation of multiple routes **with equal cost** to the same prefix in the FIB, and, afterwards, makes use of ECMP to use, in simultaneous, these routes. In this case, the cost is defined by the length of the *AS-PATH*. Consequently, only the best paths with the same length are stored.

In the presence of link failures, some paths will no longer be available. In folded Clos topologies, given the fact that every path originated in T_0s is made through T_1s, the number of redundant paths, i.e., the ECMP value, will be determined by how many T_1s can be used to reach the destination. As a consequence, we only study link failures between T_0s and T_1s in this work.

When valleys are not enabled, the value of ECMP, that is, number of paths stored in the FIB for the same prefix, for each probability *x* of a T_0-T_1 link failure is given by

$$(1 - x)^2 \times m. \qquad (7)$$

Similarly to what is done to increase connectivity, we can use valleys to augment the ECMP value in the presence of link failures. In figure 9, we can see the theoretical value for $m = 8$, obtained with equation 7 and the simulated values with and without valleys, obtained by running the previously mentioned C++ program.

Figure 9 allows us to conclude that, given the fact that valleys only start being used for high probabilities of a T_0-T_1 link failure, the ECMP value obtained with and without

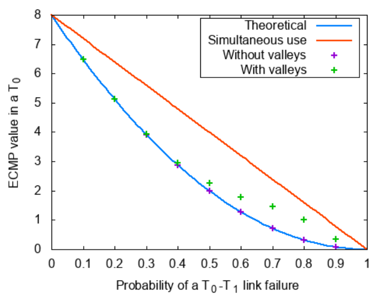

Figure 9. Values of ECMP with and without the use of valleys, as provided by BGP, and comparison with the simultaneous use of both types of paths, for $m = 8$.

valleys is the same up until a 50% probability of link failure. Past this value, the use of valleys allows the use of more paths. For example, for $m = 16$, a probability of link failure of 80% results in an average of the ECMP value without valleys of 0.64. By using valleys, this value increases to 1.94.

However, since valleys only start being used at high probabilities of failures, one must conclude that, if BGP allowed, it would be preferable to use, in terms of connectivity, non-valley and valley paths simultaneously, that is, 2-hop and 4-hop paths. In this case, an approximation of the value that ECMP would take can be deduced from the fact that two T_0s would be able to connect through a T_1, or a pair of T_1s in the same relative position of both clusters if both T_0s were in different clusters, if there was a **single active link from the source to a T_1 or from the destination to a T_1**. Therefore, if x is the probability of a T_0-T_1 link failure and there are m T_1s per cluster, then the ECMP value is given by

$$(1 - x) \times m. \tag{8}$$

This expression is also represented, for $m = 8$, in figure 9 and it allows us to conclude that using valleys in simultaneous would be advantageous in terms of the number of paths available.

5. Congestion Analysis

Data centers run different types of applications, which will determine the traffic pattern of each network. In this work, we want to examine how link failures affect the congestion of links in the different folded Clos topologies by studying the, assumed by us, traffic patterns of 3 well-known application models:

1) In **High-Performance Computing**, the application is distributed by many servers that process the data and communicate with each other to obtain results. We assume flows are intra-cluster.
2) **Search engines**, which we are treating as a concrete example of an HPC application with additional communication from and to the Internet, such as Google search [16], have specific servers dedicated to accepting requests from the Internet, and then distribute this request by other servers, assemble the results of the computation and reply to the user. We are assuming that servers dedicated to receiving requests from the Internet have 80% of their share of the traffic between themselves and other servers inside the cluster and the remaining 20% between themselves and the Internet.
3) **Streaming services** are assumed to mostly having servers dedicated to serving clients from the Internet. Servers receive requests from the Internet and start sending large amounts of data, obtained from the storage resources, to the clients. For each server, we consider 80% of the share of the traffic is between the server and the Internet and the remaining 20% to servers of the same cluster.

In our work, two important aspects should be taken into account. Firstly, as with link failures, we are only measuring traffic to and from T_0s, that is, we do not study the traffic between servers and the ToR. Secondly, we assume that each traffic flow is perfectly distributed by the ECMP and, consequently, load-balancing is perfect, that is, each traffic flow is equally distributed by all paths at each router. In practice, however, traffic flows are indivisible.

Similarly to the connectivity analysis, we divided the link failures in T_0s-T_1s and T_1s-T_2s. However, since paths from T_1s to T_2s share a T_0-T_1 link, then failures in T_0s-T_1s are the critical ones, as seen in the connectivity analysis and, as a result, our analysis focuses on T_0-T_1 link failures, with the exception of the impact of using parallel links, explained further.

In order to analyze the different traffic patterns, we implemented a C++ program that builds a graph of the network and then simulates the distribution of traffic. We compare the maximum probability of alink failure, for each traffic pattern, for two networks with 10 Gbps of capacity links, network **A** in figure 10, and network **B** in figure 11. Both networks have the same number of servers, approximately a hundred thousand, and the same number of T_0s per cluster, but have T_0s with a different number of ports. On network A, each T_0 connects to 40 servers and 8 T_1s, and on network B, each T_0 connects to 80 servers and 16 T_1s. Consequently, since both networks share the number of T_0s per cluster, network A has double the number of clusters of network B.

We can conclude that network B supports higher probabilities of failure than network A due to the higher number of T_1s per cluster. For example, for HPC, network A only supports a probability of failure of 4% for a traffic volume of 40 Gbps, and network B supports 12%, which is a value 3 times higher, for 80 Gbps of traffic. We can also see in both networks that, in general, the HPC traffic pattern is the one that supports less failures and the search engine the one that supports more. Additionally, the streaming traffic pattern is the pattern that suffers more variation with the number of T_1s per cluster. Although, in network A, the streaming pattern supports similar failures to HPC, in network B it

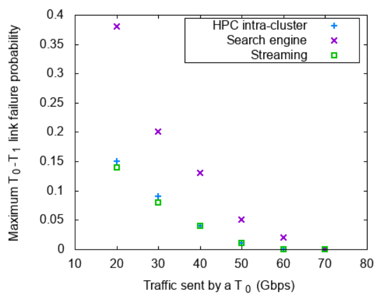

Figure 10. Comparison between the maximum probabilities of failure for the 3 different patterns of communication in network A, with 8 T_1s.

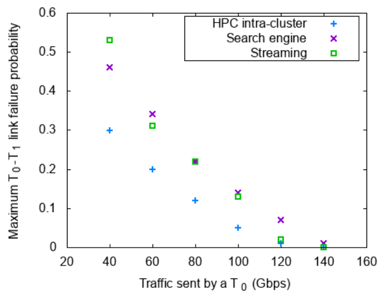

Figure 11. Comparison between the maximum probabilities of failure for the 3 different patterns of communication in network B, with 16 T_1s.

supports more failures than this pattern and has similar values to the search engine pattern.

5.1. Non-shortest path routing

We studied, for identical networks and traffic patterns, the difference in the maximum T_0-T_1 link failure probabilities when using valleys. We can observe the results for the streaming communication pattern in figure 12.

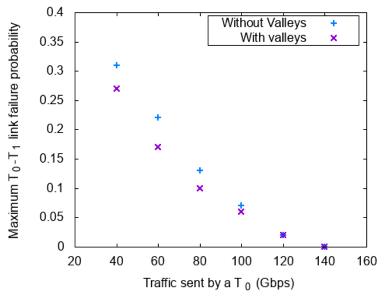

Figure 12. Comparison between the use of valleys, for the streaming patterns, of the maximum probabilities of a T_1-T_2 link failure that a network supports, as a function of the traffic sent by each T_0.

It can be concluded that valleys do not improve the maximum probability of a link failure and can, on the contrary, make the network more susceptible to overcapacitated links. This is due to the fact that valleys only start being used when the probability of failure is very high and all of the shortest paths become unavailable, that is, it will begin to have only three, two, and eventually a single shortest path available before valleys are used. When valleys start being used, even though there may be a high number of paths available, the number of disjoint paths is very small, as seen in the path diversity analysis. Therefore, if a T_0 is sending a large volume of traffic, as the percentage of links down increases, there will come a time when that T_0 may have multiple paths, but the number of disjoint links is so mall that links that share multiple paths will be overcapacitated.

Thus, with the BGP implementation as it is, valleys are mostly advantageous when the amount of traffic being sent is very small, so that it can provide connectivity between T_0s that otherwise would not be able to connect.

5.2. Parallel links

In this work, we consider the option of using parallel links between the same pair of T_1s-T_2s. Although parallel links allow a reduction in the cost of the network, it can be seen in figure 13 that, for a network with 16 T_1s and the streaming pattern, the use of parallel links greatly reduces the maximum probability of failure that a network supports due to BGP being unable to advertise the number of parallel paths.

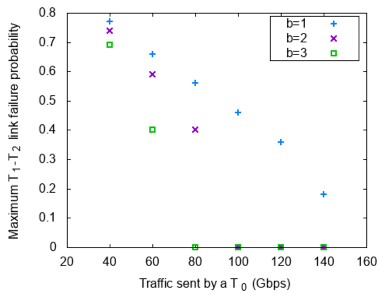

Figure 13. omparison between the use of parallel links between a T_1 and a T_2, given by parameter b, for the streaming pattern, of the maximum probabilities of a T_1-T_2 link failure that a network supports, as a function of the traffic sent by each T_0.

For example, we can see that using parallel links leads to overcapacitated links for any probability of failure for traffic volumes higher than 100 Gbps. When using a single link between pairs of T_1s-T_2s, however, the network supports, for a volume of 100 Gbps of traffic, approximately, 36% of link failures.

6. Conclusions

We studied the cost of the routers in the different tiers of a 3-tier folded Clos topology, T_0s, T_1s and T_2s. We identified that T_2 routers need more memory and forwarding capacity than T_0s and T_1s in order to interconnect the clusters and connect them to external endpoints to the data center, and to install the routes to prefixes from all over the

Internet. We concluded that the cost per server, as well as the power per server, tends to diminish with the increase in the number of servers. We showed that networks with 40 Gbps links cost up to 2 times more than networks with links of 10 Gbps, even though the capacity is 4 times higher.

We presented a detailed analysis of connectivity, in the presence of link failures, in terms of the different parameters of a topology, namely the number of T_1s per cluster and the number of connections from a T_1 to T_2s. We studied T_0-T_1 and T_1-T_2 links separately and found that the crucial links in the network are the ones between T_0s and T_1s since all of the paths from a T_0 to other T_0s, or to endpoints external to the data center, share these links. Therefore, the number of disjoint paths that a T_0 has is limited by the number of T_1s in its cluster.

We assumed and analyzed three different traffic patterns that can be present in data centers: HPC, search engines, and streaming services, and showed that increasing the number of T_1s per cluster increases the support of failures in the network.

We studied non-shortest path routing and concluded that using non-shortest paths, which we designated by valleys, improves end-to-end connectivity in the presence of link failures. We saw how the average of the number of paths that a T_0 has increases by using valleys, but only for high probabilities of link failures. We concluded that the simultaneous use of shortest and non-shortest paths, instead of exclusively using shortest or non-shortest paths, can further increase the number of paths between servers, in the presence of link failures. However, the use of valleys requires more memory in routers, due to the higher number of paths, and it can lead to networks with overcapacitated links for lower values of T_1-T_2 link failure probabilities than when non-shortest path routing is not used.

We presented the option of parallel links and concluded that, while maintaining the number of connections between a T_1 and T_2s, parallel links allow a reduction of the overall cost of the network. By using fewer T_2s and using parallel links between T_1s and T_2s, it is possible, as it was seen, to reduce up to 61% of the overall cost of the network. Even though we saw an improvement in connectivity in the presence of link failures when parallel links are used, we did not consider router failures. If we consider router failures, a T_2 failure has a bigger impact on the network when using parallel links. Additionally, since BGP does not advertise the number of parallel paths, the use of parallel links can lead to overcapacitated links, even for small values of T_1-T_2 link failure probabilities.

Acknowledgments

This work was made at Instituto de Telecomunicações.

References

[1] X. Wang, J.-X. Fan, C.-K. Lin, J.-Y. Zhou, and Z. Liu, "BCDC: a high-performance, server-centric data center network," *Journal of Computer Science and Technology*, vol. 33, no. 2, pp. 400–416, Mar 2018. [Online]. Available: https://doi.org/10.1007/s11390-018-1826-3

[2] P. Mell, T. Grance *et al.*, "The nist definition of cloud computing," 2011.

[3] K. Hazelwood, S. Bird, D. Brooks, S. Chintala, U. Diril, D. Dzhulgakov, M. Fawzy, B. Jia, Y. Jia, A. Kalro, J. Law, K. Lee, J. Lu, P. Noordhuis, M. Smelyanskiy, L. Xiong, and X. Wang, "Applied Machine Learning at Facebook: A Datacenter Infrastructure Perspective," in *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2018, pp. 620–629.

[4] R. Kohavi, A. Deng, B. Frasca, T. Walker, Y. Xu, and N. Pohlmann, "Online controlled experiments at large scale," in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 1168–1176. [Online]. Available: https://doi.org/10.1145/2487575.2488217

[5] K. Jayaraman, N. Bjørner, J. Padhye, A. Agrawal, A. Bhargava, P.-A. C. Bissonnette, S. Foster, A. Helwer, M. Kasten, I. Lee, A. Namdhari, H. Niaz, A. Parkhi, H. Pinnamraju, A. Power, N. M. Raje, and P. Sharma, "Validating datacenters at scale," in *Proceedings of the ACM Special Interest Group on Data Communication*, ser. SIGCOMM '19. New York, NY, USA: ACM, 2019, pp. 200–213. [Online]. Available: http://doi.acm.org/10.1145/3341302.3342094

[6] Andreyev, A., "Introducing data center fabric, the next-generation Facebook data center network," Facebook, Tech. Rep., 2014. [Online]. Available: https://engineering.fb.com/production-engineering/introducing-data-center-fabric-the-next-generation-facebook-data-center-network/

[7] X. Meng, Z. Xu, B. Zhang, G. Huston, S. Lu, and L. Zhang, "IPv4 Address Allocation and the BGP Routing Table Evolution," *SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 1, p. 71–80, Jan. 2005. [Online]. Available: https://doi.org/10.1145/1052812.1052827

[8] D. G. Dutt, *BGP in the datacenter*. O'Reilly Media, 2017. [Online]. Available: https://www.oreilly.com/library/view/bgp-in-the/9781491983416/

[9] "BGP Best Path Selection Algorithm," [Online; accessed November 19, 2020]. [Online]. Available: https://www.cisco.com/c/en/us/support/docs/ip/border-gateway-protocol-bgp/13753-25.html

[10] "Examples: Configuring BGP Multipath," [Online; accessed November 19, 2020]. [Online]. Available: https://www.juniper.net/documentation/en_US/junos/topics/topic-map/bgp-multipath.html

[11] R. R. Reyes, S. Sultana, V. V. Pai, and T. Bauschert, "Analysis and evaluation of capex and opex in intra-data centre network architectures," in *2019 IEEE Latin-American Conference on Communications (LATINCOM)*, 2019, pp. 1–6.

[12] L. Luo, G. Xie, S. Uhlig, L. Mathy, K. Salamatian, and Y. Xie, "Towards TCAM-Based Scalable Virtual Routers," in *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*, ser. CoNEXT '12. New York, NY, USA: Association for Computing Machinery, 2012, p. 73–84. [Online]. Available: https://doi.org/10.1145/2413176.2413186

[13] H. Hwang, S. Ata, K. Yamamoto, K. Inoue, and M. Murata, "A new TCAM architecture for managing ACL in routers," *IEICE Transactions*, vol. 93-B, pp. 3004–3012, 11 2010.

[14] D. Andersen, H. Balakrishnan, N. Feamster, T. Koponen, D. Moon, and S. Shenker, "Accountable Internet Protocol (AIP)," vol. 38, 10 2008, pp. 339–350.

[15] Q. Guo, X. Guo, Y. Bai, and E. İpek, "A resistive TCAM accelerator for data-intensive computing," in *2011 44th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE, 2011, pp. 339–350.

[16] L. A. Barroso, J. Dean, and U. Holzle, "Web search for a planet: The Google cluster architecture," *IEEE Micro*, vol. 23, no. 2, pp. 22–28, 2003.