CHISEL: An optical slice of the wide-area network

Abhishek Vijaya Kumar*, Bill Owens¶, Nikolaj Bjørner†, Binbin Guan†, Yawei Yin†, Paramvir Bahl†, Rachee Singh*

*Cornell University, †Microsoft, ¶NYSERnet

Abstract

Network slicing reserves a portion of the physical resources of radio access networks and makes them available to consumers. Slices guarantee traffic isolation, strict bandwidth and quality of service. However, the abstraction of slicing has been limited to access networks. We develop CHISEL, a system that dynamically carves slices of the wide-area network (WAN), enabling an *end-to-end* network slicing abstraction. CHISEL creates *optical slices* between WAN endpoints to avoid queueing and congestion delays inherent in packet switched paths in WANs. CHISEL incrementally allocates optical spectrum on long-haul fiber to provision slices. This task is made challenging by the co-existence of data-carrying channels on the fiber and numerous physical constraints associated with provisioning optical paths *e.g.,* spectrum contiguity, continuity and optical reach constraints. CHISEL leverages the empirical finding that cloud WANs have abundant optical spectrum to spare — 75% of optical spectrum on 75% of fiber spans is unused. CHISEL can optimally allocate terabits of slice requests while consuming minimal optical spectrum within seconds without increasing spectral fragmentation on fiber. CHISEL trades-off optimality of slice bandwidth allocation for faster run-time, provisioning slices within 2% of optimal in less than 30 seconds in a commercial cloud WAN. Finally, CHISEL reduces the latency of provisioning optical slices on hardware by 10X. Compared to IP tunnels of equivalent capacity, CHISEL consumes 3.3X fewer router ports.[1]

1 Introduction

Next generation mobile wireless networks are undergoing a technological revolution. The 3rd Generation Partnership Project (3GPP) has played a key role in this change by defining international standards that govern the capability and implementation of 5G technology [1]. Recently, 3GPP defined a *network slice* as "A logical network that provides specific network capabilities and network characteristics." Network slices guarantee traffic isolation, bandwidth and quality of service (QoS) by dynamically creating logical network instances that share the underlying physical network.

A case for end-to-end network slices. Slicing is primarily limited to access networks in the mobile network architecture (Figure 1) [30]. In this setting, traffic from user equipments (UEs) can be allocated a network slice for guaranteed QoS on the access network. However, the remaining network path from the radio access network (RAN) to the traffic destination does not support the slicing abstraction. Many applications (*e.g.,* voice, live-video, gaming) require strong network performance guarantees for reliable user experience. We aim to enable this by extending network slices from user endpoints all the way to the application server for an end-to-end QoS guarantee. The strict performance requirements of service-oriented traffic flows in next-generation mobile networks compound the need for end-to-end network slices [2].

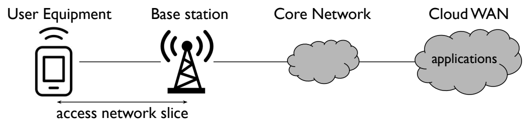


Figure 1: Access network slices.

Slices of the wide-area network. Next-generation mobile networks like 5G, offer a unique opportunity to implement end-to-end network slicing. Unlike the *monolithic* implementation of previous generations of mobile wireless, 5G disaggregates the access and packet core capability. The disaggregated access and packet core can be implemented as network functions on cloud-hosted servers and accelerators (Figure 2). The resulting *cloudification* of 5G networks has extended the placement of mobile network functions from operator base stations to cloud edges and datacenters [8]. Enabling end-to-end network slices from user equipment to cloud edges and datacenters requires a mechanism for slicing the cloud WAN.

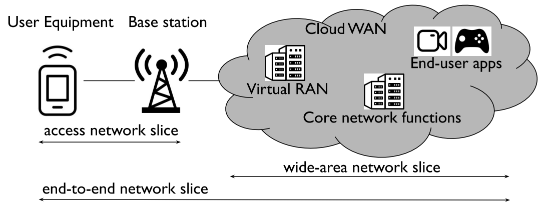


Figure 2: Access network slices vs. end-to-end network slices.

Optical slicing. In this work we undertake the task of slicing the wide-area network to implement end-to-end slices for network transport in the era of disaggregated and cloud-hosted mobile wireless networks. At their core, network slices provide dedicated bandwidth to their clients with strict QoS guar-

[1]CHISEL's code and data is at: http://opticalslice.network/.

antees. While tunneling abstractions (*e.g.,* MPLS tunnels [11], IP-in-IP [35]) make similar promises regarding performance, they fall short in practice (§2). By virtue of being built over packet-switched networks, such tunneling abstractions suffer from queuing delays and congestion at packet switches. These phenomena weaken the latency and throughput guarantees offered by tunneling, making them unsuitable for WAN slicing. Instead, we propose to slice the wide-area network at the physical or the *optical* layer.

We propose CHISEL, a system that provisions on-demand *optical slices* of the WAN. CHISEL's slices expose a network connection of dedicated bandwidth and strict performance guarantees to *network tenants* (*e.g.,* mobile access providers) of the cloud WAN. CHISEL slices provide *one-hop connections* between a source and destination router in the WAN by provisioning an all-optical path between them. CHISEL's slices achieve traffic isolation and performance guarantees on shared network infrastructure by optically bypassing all electrical packet switching between the endpoints of the slice.

Challenges. Provisioning on-demand optical WAN slices is challenging in practice for several reasons. First, while cloud providers are financially incentivized to provide an end-to-end slicing abstraction, the deployed network infrastructure (optical switches, fiber, amplifiers *etc.*) must support existing inter-datacenter traffic. Thus, carving on-demand optical slices requires accommodating slice requests in the partially occupied optical spectrum on fiber without disrupting existing traffic. Second, carving all-optical slices of the spectrum challenges physical limits like optical reach of signal transmission on fiber. Finally, cloud operators use static *channel maps* to assign optical spectrum to packet switches, making dynamic spectrum slicing slow and potentially disruptive.

Contributions. We tackle these technical challenges by first examining the utilization of spectrum in the optical backbone of a large commercial cloud provider. We find that nearly 75% of optical fiber spans in the cloud backbone have more than 75% of their spectrum freely available, showing that sufficient spectrum is free for CHISEL to carve wide-area optical slices (§3). We make the following contributions to leverage the available optical spectrum for enabling WAN slices:

Optimal slice allocation. CHISEL formulates the problem of dynamically allocating optical spectrum in cloud WANs as an optimization. CHISEL's optimization algorithm encodes physical constraints of optical signal transmission — limited optical reach, wavelength continuity and contiguity. CHISEL's algorithm can efficiently allocate optical slices in planet-scale WANs in seconds without fragmenting the spectrum (§4, §5).

Rapid and hitless provisioning of slices. CHISEL develops tools to implement optimal slice allocations computed by the algorithm on WAN optical switches. CHISEL can programmatically provision a slice of spectrum within 10 seconds without disrupting the existing channels on fiber (§6).

Field and lab experiments for evaluation. We build a hardware testbed to mimic a point-to-point wide-area fiber span of 2,600 km to proof-of-concept optical slicing. Additionally, we evaluate CHISEL in the field by provisioning a slice in an educational WAN located in New York, USA. Both laboratory and field experiments show that allocating on-demand optical spectrum is not disruptive to existing traffic and CHISEL can provision an optical slice in tens of seconds (§6).

Results. We show that CHISEL can compute slice allocations for terabits of bandwidth within a few seconds. Moreover, CHISEL takes tens of seconds to allocate an on-demand slice on optical hardware. CHISEL's allocations can be efficiently packed with existing wavelengths without increasing spectrum fragmentation. Not only do CHISEL-provisioned optical slices enable better performance guarantees and isolation than packet-switched tunnels, they offer these benefits at a lower price point. We show that CHISEL slices consume 3.3X fewer router ports compared to traffic engineering (TE) tunnels of the same bandwidth (§7).

General-purpose abstractions for optical circuit switching. We note that slicing optical spectrum on-demand is a general idea that is broadly applicable to other settings where network tenants require strict performance guarantees that go beyond the ability of packet switched networks. We believe that cloudified 5G network operators would benefit from CHISEL's optical slices in the near term (§8).

2 CHISEL Design

CHISEL creates on-demand optical slices of the WAN, in response to user requests. CHISEL's users are the tenants of the WAN, like cloudified access network operators [8]. Optical slices are *optical tunnels*, similar in function as tunnels at higher layers of the networking stack (*e.g.,* IP-in-IP, MPLS, GRE) — they provide a bandwidth pipe between the two ends of the slice. Unlike traditional tunnels, optical slices provide guaranteed network bandwidth and quality of service between endpoints. These strict guarantees are made possible by dynamic reservation of physical resources on the optical WAN. CHISEL's slices eliminate packet switching on the slice since bulk of network delays, packet drops and performance jitter are caused at the router hops in the path [4].

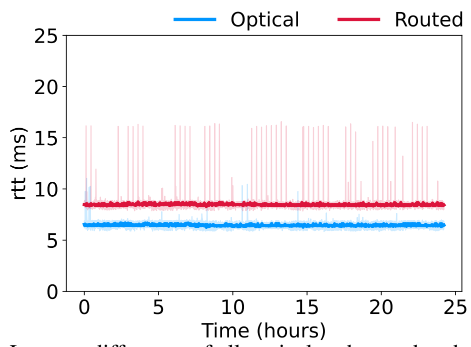


Figure 3: Latency difference of all-optical and routed paths between the same endpoints in a WAN.

Why optical slicing? We quantify the benefit of an optical slice vs. a packet switched path in a production WAN. We describe the detailed setup for this experiment in Appendix A. We use isochronous round-trip testing or *irtt* measurements at every 500 ms over the span of 24 hours to compare the latency between the same endpoints in the WAN over an all-optical path (like the ones enabled by CHISEL) and a routed path. We find that despite keeping all factors the same, *e.g.,* , the underlying fiber path, the endpoint equipment, time of day *etc.*, the all-optical path outperforms the routed path in terms of latency and jitter by a significant amount (Figure 3). Motivated by this experiment and the intuition that optical paths offer more predictable performance, we implement network slicing in the wide-area at the optical layer.

What about statistical multiplexing? CHISEL slices the WAN by carving chunks of optical spectrum, dynamically. This is similar to circuit switching at the physical layer. One might argue that packet switched paths offer a multitude of benefits to networks, like statistical multiplexing [29] over network resources and CHISEL is decidedly avoiding packet switched long-haul paths. Moreover, circuit switching necessitates the reservation of resources which can lead to inefficient utilization of the network. However, the optical layer of cloud WANs is *massively over-provisioned* by design (§3), making abundant optical spectrum available for reservation with no downside. Therefore, by using CHISEL, we are not making the network inefficient in carrying existing traffic and are additionally saving network tenants from the downside inherent to routed paths — unpredictable performance.

Why is this new? Network operators incrementally provision capacity on long-haul fiber (§3). At first glance, these well-known network capacity provisioning mechanisms seem sufficient to achieve CHISEL's goals [31]. However, CHISEL must *dynamically* carve bandwidth on fiber at both line-rate (§6) and sub line-rate granularity (§8) in operational networks without causing side-effects to existing traffic. These requirements pose harder technical challenges in terms of system design, hardware implementation and algorithmic complexity. We focus on these challenges in this work.

Our contribution. CHISEL relies on the availability of optical spectrum in operational cloud networks. We first show that optical spectrum is available in abundance in large parts of a cloud network (§3.3), making it feasible to carve it for optical slicing. Spectrum allocation on an operational network is subject to a number of complex physical constraints of signal transmission on fiber and hardware limitations. We formulate the problem of efficient spectrum slicing by encoding these constraints as an optimization objective (§4) and solve it in a few seconds for large WANs. Finally, implementing the optimal allocation of optical slices computed by CHISEL's optimization requires careful software design that works around the limitations of current optical hardware. CHISEL accelerates this by programmatically creating all-optical slices within seconds on off-the-shelf optical hardware without disrupting existing data-carrying channels (§6).

How will CHISEL enable end-to-end network slicing? Radio-access network slices are created and managed by a Network Slice Subnet Management Function (NSSMF) as per the 3GPP standard [26]. Similarly, CHISEL will interface with the *transport NSSMF* to provision WAN transport slices. Together with the RAN and core NSSMFs, CHISEL will enable end-to-end slices from the user end point to the traffic destination in next-generation mobile wireless networks.

3 Optical spectrum in long-haul fiber

Long-haul connectivity in wide-area networks is built with optical wavelengths on fiber. Cloud providers provision their WANs by laying fiber between cloud datacenters. Such fiber, often referred to as "dark fiber", is acquired by expensive manual effort of laying fiber under the ground. Once acquired, long-haul fiber is incrementally utilized for several decades.

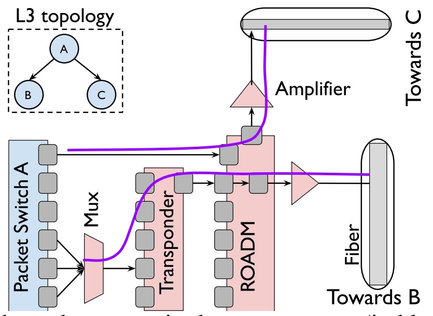


Figure 4: shows how capacity between routers (in blue) is provisioned by adding wavelengths via optical hardware (in pink). Provisioned capacity translates to spectrum allocation on optical fiber.

Capacity is provisioned incrementally. Based on the estimated capacity required between routers, cloud operators provision *wavelengths* on the fiber. Provisioning a wavelength requires connecting ports on a packet router to an optical switch or multiplexer which can *add or drop* an optical wavelength onto connected fiber (Figure 4). A wavelength is a unit of *optical spectrum* on fiber that is encoded with bits egressing from router ports. Provisioning a wavelength incurs bulk of the hardware cost of acquiring network capacity (*e.g.,* router ports, transceivers, optical switch ports) and is done incrementally as the need for more capacity arises.

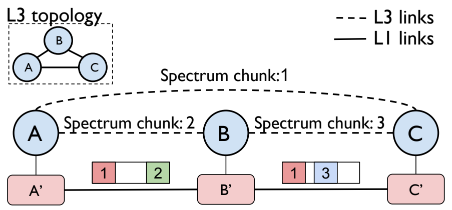


Figure 5: Spectrum allocation to provision router edges $A-B$, $A-C$ and $B-C$. Edge $A-B$ uses spectrum chunk 2 on optical span $A\prime - B\prime$. Edge $B-C$ uses chunk 3 on span $B\prime - C\prime$. Edge $A-C$ uses chunk 1 on both spans $A\prime - B\prime$ and $B\prime - C\prime$.

Optical spectrum is limited. A wavelength on fiber can have variable *spectral widths*, corresponding to the size of optical spectrum they occupy. Advances in optics and signal processing have enabled transmitting over 100 wavelengths on a single strand of fiber. However, the overall spectrum on fiber is limited to the frequencies in the *C-band* of the electromagnetic spectrum. Optical signals with frequencies belonging to the C-band and nearby suffer the lowest loss as they propagate. Therefore, most long-haul signal transmission in WANs is limited to the C-band. Figure 5 shows how multiple wavelengths occupy the optical spectrum on fiber.

3.1 Spectrum translates to network capacity

Higher layers in the networking stack abstract network connections as edges between routers, annotated with capacity. The physical network underlying this graph abstraction translates edge capacities into allocations of optical spectrum in fiber (Figure 5). The Shannon capacity [33] of a wavelength *i.e.,* the theoretical upper bound on the wavelength's capacity, is proportional to the width of spectrum allocated to the wavelength. Thus, allocating larger chunks of spectrum leads to higher data rates and vice versa.

Modulation format. Modulating the wavelength with *higher-order* modulation formats (*e.g.,* 8-QAM, 16-QAM) packs more bits per *symbol* of the wavelength, resulting in higher data rates or capacity of the wavelength. Overall, the choice of spectral width and modulation format decides the data rate of router ports that are the endpoints of a newly provisioned network edge in the logical wide-area network.

Signal quality. Sustaining higher order modulation formats requires a sufficiently high signal-to-noise-ratio or SNR — the second factor that determines the Shannon capacity of a data channel. Signal quality can be engineered by the operator only to a certain extent. Factors outside the control of the operator affect the signal quality *e.g.,* impairments to fiber, noise and power levels of amplifiers *etc.*Thus, the signal quality can often limit the achievable data rates of wavelengths.

Optical reach. A consequence of the selected modulation format is the *reach* of the optical signal. Optical reach is the longest distance a signal can travel on fiber before it needs to be *regenerated*. After traversing distances higher than the optical reach on fiber, it is essential to regenerate signals to recover data bits from the errored bits. Signal regenerations are expensive in terms of hardware and power since they require conversion of optical signals into electrical signals followed by a conversion back into optical signals. Higher order modulation formats are more susceptible to signal attenuation during transmission and therefore have shorter optical reach.

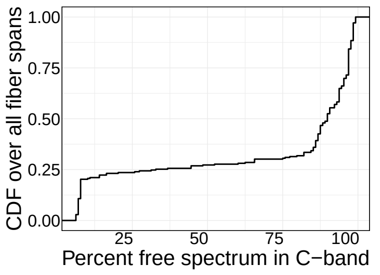
3.2 Spectrum usage in the wild

Operational networks provision wavelengths manually today. Network operators keep offline *channel maps* that map capacity between router ports to wavelengths and their corresponding spectral widths on all fiber spans in the WAN. For instance, in Figure 4 three router ports are multiplexed to feed bits into the chunk of spectrum allocated on fiber towards node B whereas a single port feeds bits to the smaller chunk of spectrum on fiber towards node C.

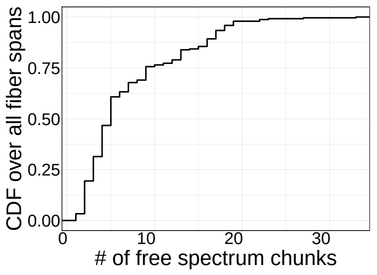
Ideally, spectrum channel maps should be *correct*, *efficient* and *performant*. Correct channel maps ensure that different router ports are not allocated overlapping spectrum. Efficient channel maps reduce the amount of spectrum used to provision the network as electromagnetic spectrum, especially the C-band, is limited. Performant channel maps ensure that the allocated spectrum meets the network capacity and optical reach goals. For example, an operator intends to augment the capacity between router ports in Austin, TX and New York City, NY. In this case, the provisioned wavelength should have an optical reach roughly equivalent to the distance between the two cities $\approx 2,800$ km. In Figure 5, to connect A and C *directly* at the Network layer, an operator must provision the same wavelength on both fiber spans $A\prime - B\prime$ and $B\prime - C\prime$. Maintaining *continuity* of wavelength allocations across spans is essential in fiber transmission to connect endpoints without regenerations at intermediate hops, like B.

Regenerations or OEO conversions. Cloud providers operate *point-to-point* optical networks [37]. In point-to-point optical networks, physically adjacent locations in the WAN are also adjacent electrically. In other words, all wavelengths undergo optical to electrical to optical (OEO) conversion at every hop in the network. Operators run point-to-point networks for flexibility and operational simplicity. The OEO conversion forces correction of errors that happen during transmission of signals. In such networks, capacity between router ports in Austin, TX and New York City, NY can be provisioned using different wavelengths on multiple fiber spans connecting the two cities. For instance, in Figure 5, A and C have indirect connectivity via B. This connection is enabled by two different wavelengths, 2 on $A\prime - B\prime$ and 3 on $B\prime - C\prime$.

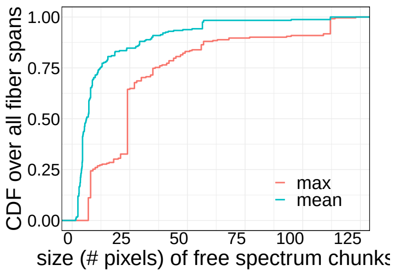
Fully-loaded line systems. Cloud providers operate *fully-loaded* line systems [14]. In such line systems only a fraction of the optical spectrum is provisioned and connected to router ports. However, the *entire* spectrum has optical channels traversing it. The optical channels that are not connected to router ports simply carry *noise* generated by specialized equipment. Network operators fully-load their optical line systems to ensure that the network will function as needed in the future, when high network demands necessitate the allocation of the entire spectrum. The steady-state of optical spectrum in an operational cloud network appears fully utilized since operators use fully-loaded line systems. Therefore, it is important to differentiate between noise-carrying wavelengths and *data-carrying* wavelengths. Data-carrying wavelengths are the ones that contribute to spectrum utilization since they are allocated for provisioning capacity in the network.



(a) Percentage unused spectrum.



(b) Number of unused spetrum chunks.



(c) Size of unused spectrum chunks.

Figure 6: Figure 6a shows the percentage of optical spectrum pixels that are free across all spans in a commercial cloud provider's WAN. Figure 6b shows the number of contiguous unused chunks of spectrum across all spans in the cloud provider's network. Figure 6c shows the size of unused regions of spectrum on fiber.

3.3 Available spectrum

We measure the state of optical spectrum on fiber spans of a large commercial cloud provider. We collect which wavelengths (or frequencies) map to different ports of wide-area optical switches or reconfigurable optical add/drop multiplexers (ROADMs). Since the cloud provider runs a fully-loaded line system, the data collected from optical switches alone will not accurately inform us of the true spectrum allocation. Instead, we augment the Layer-1 data collected from ROADMs with Layer-3 network capacity information that maps router ports to ROADM ports. If we find that no router ports map to a given part of the optical spectrum, we mark that spectrum *available*. In this manner, we differentiate between noise-carrying vs. data-carrying optical channels in the WAN.

The C-band consists of frequencies in the range 191100 – 195900 Ghz. We discretize the frequency band into 128 *pixels*, each corresponding to 37.5 GHz of frequency. Using the cross-layer data, we mark which pixel is free vs. used by data-carrying wavelengths for all spans in the planet-scale cloud WAN. Since CHISEL's goal is to allocate unused portions of spectrum to slice requests, we quantify the unused spectrum on optical spans of the WAN. Figure 6a shows that 75% of optical spans have over 75% of their spectrum available. The free pixels in the spectrum are concentrated in a small number of *contiguous chunks* of free spectrum. Figure 6b shows that free pixels in over 70% of spans are concentrated in less than 10 contiguous chunks. Finally, majority of spans have free spectrum chunks that are less than 25 pixels wide (Figure 6c).

Implications for CHISEL. Our empirical analysis shows that the physical layer of wide-area networks is heavily over-provisioned in terms of optical spectrum on fiber. The main reason for the low utilization of optical spectrum is the large difference between switching capacity of state-of-the-art network routers and data-carrying capacity of modern optical fiber. Fiber can carry tens of terabits of traffic, which is a magnitude higher than the total port capacities of a router linecard. Moreover, operators incrementally deploy more router ports to provision capacity but fiber is typically acquired by multiple strands in the same conduit. Large providers can fill the the spectrum over the next several years from the time they first acquire fiber. This fits well with CHISEL's agenda — there is sufficient free optical spectrum on fiber for CHISEL to carve slices without impacting existing traffic.

4 Bandwidth slices in optical WANs

We formalize the problem of carving optical slices on *operational* optical cloud WANs. Our goal is to establish end-to-end optical slices for access network operators using the available optical spectrum in the cloud backbone.

4.1 Switching in the optical domain

CHISEL's goal is to dynamically carve optical spectrum on end-to-end wide-area paths. This task is subject to physical considerations of signal transmission on fiber. Moreover, while CHISEL's algorithm will be a part of a Layer 1 software-defined controller [9, 12, 20, 34] optical switches should be capable of efficiently implementing decisions made by CHISEL. We briefly discuss the signal transmission and hardware considerations that shape CHISEL's algorithm design.

Discrete units of spectrum. The entire spectrum on fiber is limited to the C-band and parts of the L-band. Roughly, there is 4,800 GHz of usable spectrum on fiber. We discretize the spectrum into chunks of 37.5 GHz. While fiber carries wavelengths, the decisions of routing wavelengths on different fiber paths are made by the optical switches or ROADMs. Each ROADM has a *grid* of *pixels* to represent parts of the spectrum. A pixel is one unit of spectrum that the ROADM can provision towards a wavelength that connects ROADM ports to router ports. We denote the grid of an optical switch (v) using S^v where S^v_i represents the i^{th} pixel's state. If S^v_i is 1, it means the corresponding part of the optical spectrum is already in use and $S^v_i = 0$ implies the corresponding portion of the spectrum is available to carve slices using CHISEL.

Slicing requests. CHISEL receives a set of slicing requests from clients. Each request specifies the desired bandwidth between a pair of source and destination WAN routers. We represent every slice using the source and destination pair (sd) of the slice. We represent the set of fiber paths between source

s and d using P_{sd}. We compute the set of fiber paths using Yen's k-shortest path algorithm [40]. Each fiber path $p \in P_{sd}$ is an ordered set of optical circuit switches or ROADMs and $v \in p$ means that path p contains switch v. The variable y_{sd}^{p} is an indicator that path p is chosen to carve slice sd,where $p \in P_{sd}$. We use x_{sd} to refer to the vector of spectrum pixels allocated to the sd slice. The vector x_{sd} is indexed at position j by x_{sd}^{j}. x_{sd}^{j} is binary.

4.2 Optimal slice allocation

In this section, we discuss the physical, hardware and operational constraints that impact optical slicing with CHISEL.

Pick only one path for a slice request. This constraint ensures that of the possible fiber paths that can be selected for carving the sd slice, CHISEL should select only one.

$$\sum_{p \in P_{sd}} y_{sd}^{p} = 1, \qquad \forall sd \tag{1}$$

Using available parts of the spectrum for slices. The spectrum allocated to a slice on a fiber path must not overlap with the spectrum that has already been allocated for carrying inter-datacenter traffic in the cloud WAN. S^{v} is a binary vector that represents the initial state of spectrum utilization on switch v. S^{v} is a result of the existing capacity provisioned in the cloud network before CHISEL can carve slices. S^{v} is an input to CHISEL. The decision variable x_{sd} in CHISEL's optimization is a binary vector of size M, where M is the total pixels in every optical switch. x_{sd} represents pixels allocated to the sd slice. The following constraint ensures that CHISEL allocates only the available portions of spectrum to slices:

$$(y_{sd}^{p} = 1) \implies x_{sd} \cdot (S^{v})^{\intercal} = 0, \qquad \forall sd, p \in P_{sd}, v \in p \tag{2}$$

Non-overlapping use of spectrum slices. The spectrum allocated to different slices by CHISEL should not overlap. We encode this constraint as:

$$\sum_{sd} \sum_{p \in P_{sd}, p \ni v} y_{sd}^{p} \cdot x_{sd} \leq \mathbf{1}, \qquad \forall v \in V \tag{3}$$

The constraint selects switches (x_{sd}) that are on the chosen path (y_{sd}^{p}) for a slice (sd) and ensures that a pixel on each switch is allocated across all slices at most once. We use **1** to present a vector of length M consisting of all 1s.

Spectral width. The spectral width of an optical slice is the portion of the spectrum assigned to it by CHISEL. We denote the spectral width of the sd slice with w_{sd}.

$$\sum_{j=1}^{j=M} x_{sd}^{j} = w_{sd}, \qquad \forall sd \tag{4}$$

Slice bandwidth. The bandwidth allocated to a slice is decided by two factors: (1) the chosen path for the slice and (2) the spectral width of the slice. The chosen path determines the signal quality of the received signal. The signal quality, in turn, determines which modulation format can the signal be modulated with. Higher order modulation formats can pack more bits on the channel, increasing their bandwidth. We represent the modulation format of a optical path (p) with $mod(p)$. The data rate of a slice is the product of the spectral width of the slice and the modulation format of optical path.

$$(y_{sd}^{p} = 1) \implies w_{sd} \cdot mod(p) \leq B_{sd}, \qquad \forall sd, p \in P_{sd} \tag{5}$$

Wavelength contiguity. Optical spectrum allocation is subject to the physical constraint of *contiguity.* As per this constraint, CHISEL must assign every slice *contiguous* parts of the spectrum on fiber. Wavelength contiguity ensures efficient use of the spectrum since every data channel on the fiber must be separated from the next with *guard bands*. Chopping the spectrum in non-contiguous smaller chunks increases the number of guard bands, wasting the spectrum available for carrying data. By allocating contiguous parts of the spectrum to slices, CHISEL minimizes the number of guard bands.

To represent this constraint we track the first pixel that transitions from used to unused states. We call this state the *toggle* state and denote it by the binary variable t_{sd}^{j}. Pixels after the toggle state should not be used in a slice allocation. We encode the contiguity constraint using three inequalities:

$$x_{sd}^{j-1} \leq x_{sd}^{j} + t_{sd}^{j}, \qquad \forall sd, j \in \{2, \ldots M\} \tag{6}$$

$$t_{sd}^{j-1} \leq t_{sd}^{j}, \qquad \forall sd, j \in \{2, \ldots M\} \tag{7}$$

$$t_{sd}^{j} + x_{sd}^{j} \leq 1, \qquad \forall sd, j \in \{1, \ldots M\} \tag{8}$$

The first inequality enforces the toggle to be 1 whenever the j'th pixel transitions from used to unused. Inequality (7) ensures that the toggle remains 1 after it is set, and (8) prevents pixels to be used after the first transition from used to unused.

What about wavelength continuity? A key physical constraint for spectrum allocation on optical fiber is called *wavelength continuity* constraint. This constraint ensures that the pixels assigned to all switches on the path for a slice should be the same. For instance, if pixels 2 and 3 are assigned to allocate a slice on one optical span of the selected path, all other spans should also use pixels 2 and 3 for the slice. A discontinuous allocation of spectrum across optical spans necessitates a conversion of the signal to the electrical domain to change the wavelength of the channel. OEO conversions are expensive and CHISEL avoids them. Instead of adding spectrum continuity as a constraint to the optimization, CHISEL embeds it in the formulation by assigning one spectrum allocation decision variable (x_{sd}) for all spans on the path of a slice. This reduces the number of decision variables and constraints, making CHISEL's optimization scale to large problem sizes.

Objective. CHISEL's goal is to maximize the bandwidth provisioned towards slices. We express the overall slice bandwidth as the total bandwidth allocated to all slices by CHISEL:

$$\sum_{sd} \sum_{p \in P_{sd}} y_{sd}^{p} \cdot w_{sd} \cdot mod(p) \tag{9}$$

```
Algorithm 1 Optical slicing with CHISEL
Inputs:
    $G\langle V,E \rangle$:   optical network $G$, optical switches $V$ and
                  fiber links $E$
    $S^v$:        binary vector indicating initial spectrum
                  utilization on the optical switch $v$
    $P_{sd}$:     set of fiber paths between optical terminals $s$, $d$
    $B_{sd}$:     bandwidth request for the $sd$ slice
    $mod(p)$:     modulation format supported on path $p$
    $M$:          total pixels available on optical switches
Outputs:
    $y_{sd}^p$   $\in \{0,1\}$      1 if fiber path $p$ in $P_{sd}$ is selected to
                                    carve the $sd$ slice, 0 otherwise
    $x_{sd}$     $\in \{0,1\}^M$    binary vector represents spectrum
                                    allocated to $sd$
    $t_{sd}$     $\in \{0,1\}^M$    binary vector represents toggle of $x_{sd}$
    $w_{sd}$     $\in \{1,\dots M\}$  spectral width of the $sd$ slice
Maximize: $\sum_{sd}\sum_{p\in P_{sd}} y_{sd}^p \cdot w_{sd} \cdot mod(p)$
subject to $\forall sd, p \in P_{sd}, v \in p$:
    (1)  $\sum_{p\in P_{sd}} y_{sd}^p = 1$
    (2)  $(y_{sd}^p = 1) \implies x_{sd} \cdot (S^v)^\intercal = 0$
    (3)  $\sum_{sd}\sum_{p\in P_{sd}, i\ni v} y_{sd}^p \cdot x_{sd} \leq \mathbf{1}$
    (4)  $\sum_{j=1}^{j=M} x_{sd}^j = w_{sd}$
    (5)  $(y_{sd}^p = 1) \implies w_{sd} \cdot mod(p) \leq B_{sd}$
    (6)  $x_{sd}^{j-1} \leq x_{sd}^j + t_{sd}^j$          $j \in \{2,\dots M\}$
    (7)  $t_{sd}^{j-1} \leq t_{sd}^j$                    $j \in \{2,\dots M\}$
    (8)  $t_{sd}^j + x_{sd}^j \leq 1$                    $j \in \{1,\dots M\}$
```

Managing spectral fragmentation. An important consideration for CHISEL is to allocate slices in a way that does not increase *spectral fragmentation* on fiber spans. Higher spectral fragmentation makes it challenging to find large enough contiguous free pixels in the spectrum. The lack of large chunks of contiguous spectrum can hamper provisioning capacity in the network. We encourage CHISEL to reduce spectral fragmentation by pushing the allocated slices to the extreme left of the spectrum on any fiber span.

$$\sum_{sd}\sum_{p\in P_{sd}} y_{sd}^p \cdot w_{sd} \cdot mod(p) + \varepsilon \sum_{sd}\sum_{p} [y_{sd}^p \cdot mod(p) \cdot \sum_{j=1}^{j=M} t_{sd}^j] \tag{10}$$

Key insight. Equation 10 sums the toggle variables for a slice on every span. Recall that the toggle variables (t_{sd}^j) are set for all pixel counts higher compared to the one where the spectrum allocation of a slice ends. Therefore, larger sum of toggle variables means the slice allocation is to the left of the spectrum since that will allow more toggle variables to be set. By encouraging the allocations to be shifted to the left of the spectrum, we aim to close gaps in allocations of CHISEL. We trade-off the two goals *i.e.,* allocating spectrum for most slice requests and reducing spectral fragmentation using the parameter ε. We set the value of epsilon empirically to achieve high slice allocations with less spectral fragmentation in §5.4.

Implementation. Algorithm 1 summarizes CHISEL's optimization formulation for allocating slices in the WAN. We implement Algorithm 1 in Python 3 and solve it using the commercial solver, Gurobi [16]. We evaluate CHISEL on an Ubuntu 22.04.1 server with 128 cores and 1024 GB RAM.

5 Slice allocation with CHISEL

We first evaluate our implementation of CHISEL's optimization formulation (Algorithm 1) on the optical WAN of a large commercial cloud provider. We then show that our results are robust on different publicly available network topologies.

Network topologies. We work with a large commercial cloud provider to collect optical topology information in their WAN. The physical topology consists of optical switches or ROADMs and fiber spans that connect them. In addition to the cloud provider's network topology, we also evaluate CHISEL on publicly available topologies in the Topology Zoo [39]. These topologies ($G\langle V,E \rangle$) are input to an instance of Alg 1. We compute k-shortest paths between all pairs of nodes in the topology to provide possible fiber paths for slices (P_{sd} in Alg 1). We set $k = 4$ for the evaluation.

Initial spectrum state. We infer the spectrum utilization on fiber spans of the commercial cloud WAN (§3.3). For each fiber span, we construct a bitmap (S^v) of 128 pixels (M in Alg 1), each pixel representing 37.5 GHz of the 4,800 GHz of spectrum in the C-band. We collect spectrum channel maps for all fiber spans in the cloud network and mark bits in the bitmap to 1 if they are allocated to a data-carrying channel. Figure 6a in §3.3 shows the percentage of unset bits in the bitmap across all fiber spans. While Topology Zoo provides several real network topologies, it does not have spectrum utilization information for them. Instead, we use the bitmaps from the cloud provider network as a distribution of spectrum allocations. For every edge in a network from Topology Zoo, we randomly sample a bitmap from the distribution to populate the spectrum bitmaps of edges in all network topologies.

Modulation formats and fiber lengths. We use geographic distance between the end points of individual fiber spans as a proxy for the lengths of fiber spans. We derive the length of fiber paths input to CHISEL by adding the lengths of individual fiber spans that compose the path. Modulation formats and the corresponding data rates of wavelengths are a function of the path length (Table 1). Thus, we assign modulation formats to all paths input to CHISEL based on their fiber path length.

Mod. format	QPSK	8-QAM	16-QAM
Data rate	100 Gbps	150 Gbps	200 Gbps
Optical reach	5,000 km	2,500 km	800 km

Table 1: Data rates and optical reach of sigal modulation on fiber.

Hardware pixel counts. State-of-the-art flexgrid ROADMs can allocate spectrum to wavelengths in increments of 6.25 GHz. Ideally, 6.25 GHz should be the width of one pixel for CHISEL. However, optical line-side ports on ROADMs operate at coarser granularity of spectrum, traditionally 37.5 GHz and above. Moreover, very fine-grained allocation of spectrum increases the number of variables that CHISEL has to contend with (x_{sd} in Algorithm 1). Given CHISEL solves a mixed-integer program, a class of problems well known for being NP-Hard, increasing the width of a pixel also improves the scalability of CHISEL. Therefore, we set each pixel to have a width of 37.5 GHz while evaluating CHISEL. We note that this is not a fundamental limitation of CHISEL but a hyperparameter that the operator can set for their network.

5.1 Efficiency of slice allocations

We generate slice requests on the cloud WAN to test what fraction of them can get allocated by CHISEL. We scale the number of slice requests input to CHISEL and observe what fraction of overall bandwidth demands can be successfully allocated on the cloud WAN.

Granularity of slice allocations. In practice, customers of CHISEL can request a variety of slice bandwidths, similar to numerologies in 5G parlance. To test this, we allow each slice request to select a bandwidth value from 50 Gbps to 200 Gbps in discrete increments of 50 Gbps. For each slice request, we randomly sample a bandwidth value from the set [50, 100, 150, 200] Gbps. Figure 7 shows the percentage of overall slice bandwidth requests were successfully allocated by CHISEL on the cloud WAN. The shaded region shows the std. deviation from the mean allocation across five instances of the experiment for each data point. For smaller number of slice requests, CHISEL can allocate nearly 100% of the requests but as we increase the number of slices beyond 50, CHISEL is limited by the availability of contiguous spectrum in the WAN. Interestingly, we find that only a handful of fiber spans are bottlenecks in allocating more spectrum.

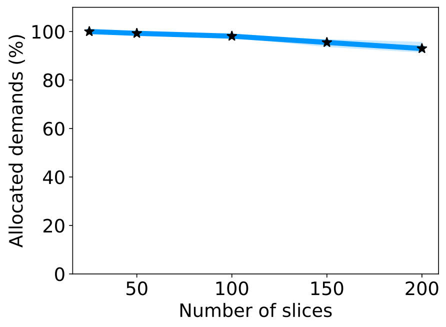


Figure 7: shows the percentage of slice bandwidth requests CHISEL can optimally allocate on the cloud wide-area network.

5.2 Scalability of the algorithm

The mixed integer program (MIP) in Algorithm 1 forms the core of CHISEL. MIPs are difficult to scale to large problem sizes. We pay attention to these scaling challenges while both designing the MIP and providing it inputs. For instance, we obviate the need to encode continuity constraints (§4) for slices and decide appropriate pixel widths to reduce the number of decision variables in Algorithm 1. For MIPs, commercial solvers like Gurobi allow setting a *MIP gap* which is the difference between the solution found by Gurobi and the optimal solution. Gurobi estimates the optimal by relaxing the integer program to a linear program which can be solved quickly. The MIP gap parameter tells Gurobi to find solutions that are close to the optimal. We set the MIP gap to 2%, a small percentage away from optimal, for the next set of experiments. We evaluate the time it takes for the optimization to converge to a solution within 2% of optimal.

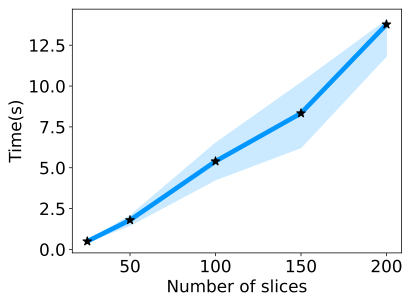
Scaling with the number of slice requests. Figure 8a shows the time needed to solve Algorithm 1 as we increase the number of slice requests. We find that CHISEL's optimization can be solved within 12 seconds on average for up to 200 slice demands, allocating optical spectrum for up to 40 Tbps across all slices. To put this in context, a well-known resource allocation problem in WANs, traffic engineering (TE), routes traffic on an existing Layer 3 network graph. The time budget to compute optimal routes in TE is nearly five minutes [24].

Scaling with slice granularities. Next, we show how long CHISEL takes to allocate slices at different granularities. We allow slices to request bandwidth from ranges [50, 100, 150, 200] Gbps, [150, 200, 250, 300] Gbps, [250, 300, 350, 400] Gbps and [350, 400, 450, 500] Gbps. Figure 8b shows that it takes longer for CHISEL to allocate large slices close to 500 Gbps per slice. This is because it is harder to find large contiguous chunks of spectrum that already do not have data-carrying channels on it vs. allocating smaller chunks needed for small slice granularity. We show the modulation formats of CHISEL-provisioned slices in Appendix B.

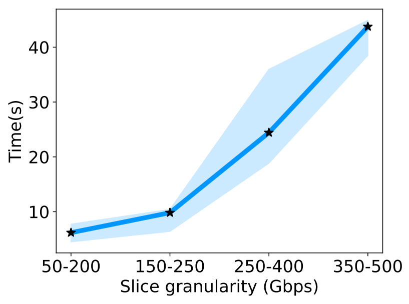
Scaling with network sizes. Figure 8c shows how well our findings generalize to other network topologies aside from the one of the commercial cloud provider. We find that CHISEL can scale to different realistic network topologies and slice counts while staying within 2% of the optimal solution. For all networks in Figure 8c, CHISEL takes less than 30 seconds to converge to a solution that is 2% away from the optimal.

5.3 Bounded sub-optimality for fast runtime

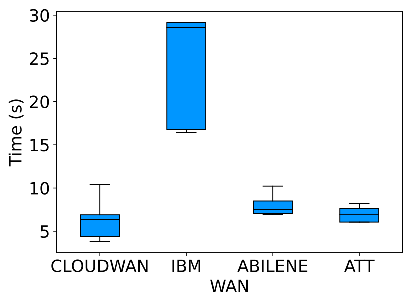
We evaluate how CHISEL performs in tight time budgets of cloud operators who need to provision optical slices rapidly. To do this, we give Gurobi small time budgets and evaluate the quality of solutions found for Algorithm 1 within these time budgets. We find that for small slice bandwidths, CHISEL can find solutions with less than 2% MIP Gap within 5 seconds. So, we focus on the time vs. MIP gap tradeoff for larger slice bandwidths in Figure 9. It shows how far the solution is from the optimal when the solver is given a time budget. We find that CHISEL converges to within 5% of optimal in thirty seconds even for large slice bandwidth requests (*e.g.*,



(a) Solver time vs. slice count.



(b) Solver time vs. slice granularity.



(c) Solver time vs. network topology.

Figure 8: Figure 8a the time taken by CHISEL to allocate different number of slices. CHISEL can efficiently compute slice allocations within 2% of optimal in less than 10 seconds for up to 200 slices. Figure 8b measures the effect of slice granularities on how long CHISEL takes to compute solutions. We show that CHISEL takes longer to find large slice (350 – 500 Gbps per slice) allocations. Figure 8c shows how the time to compute slice allocations with CHISEL changes for different network topologies. All experiments are repeated five times with different randomly sampled slices bandwidths. The shaded region around data points show the 25 and 75 percentile of results.

300–750 Gbps per slice). While there is a small improvement in MIP Gap when the time budget is increased to 60 seconds, increasing the budget past 90 seconds is not needed since the problem has converged to optimal.

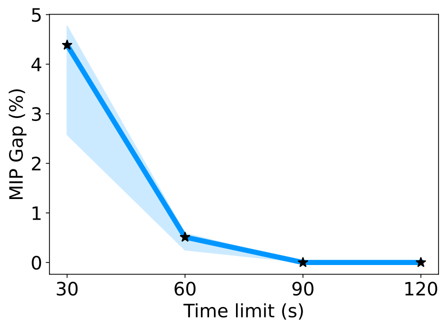


Figure 9: shows how CHISEL can tradeoff optimality for faster runtime in practical deployments.

5.4 Spectral fragmentation

In §4 we formulate two goals for CHISEL. The first (Equation 9) objective maximizes the bandwidth allocation across all slice requests. The second (Equation 10) objective maximizes the bandwidth allocation across slices while attempting to push all allocations on the low end of the optical spectrum. This formulation aims to reduce the fragmentation of spectrum by incentivizing the solver to allocate all slices in the lower frequencies. We evaluate the effect of both objectives on the amount of fragmentation and the allocated bandwidth.

We increase the value of ε in Equation 10 from 0.01 to 1 to vary the importance of fragmentation awareness in CHISEL. To show the effect of fragmentation clearly, we start with empty spectrum allocation on all spans of the cloud network and then allocate 200 slices in the range 50 – 200 Gbps. Figure 10 compares the fragmentation index (Eq 11) for all spans in the network and finds that operators can set ε to minimize fragmentation while allocating all slice requests.

$$F = 1 - \frac{\text{largest block of free spectrum}}{\text{Total free spectrum}} \tag{11}$$

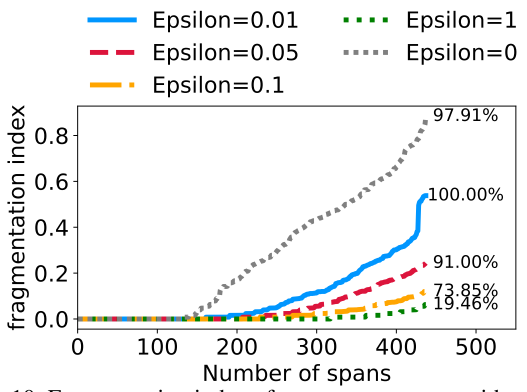


Figure 10: Fragmentation index of spectrum on spans with various degrees of fragmentation awareness in CHISEL. Each curve is annotated with percentage of allocated bandwidth across slices.

6 Hardware Evaluation

Once CHISEL computes slice allocations (§5), it must program these allocations on to the hardware. In this section, we discuss the process of physically allocating spectrum on optical fiber. We experiment with slice allocation on a laboratory hardware testbed that mimics a single optical span in a production network. We then repeat these experiments in the field in the optical WAN of an ISP in NY state.

Lab experiment setup. In the lab we replicate a typical optical span in cloud WANs using the optical line system equipment from the equipment manufacturer, VENDOR-A. The span connects two optical switches with 2,600 km of optical fiber. The two end points of the span have optical switches or ROADMs which can add/drop wavelengths on the fiber. ROADMs come in the form of linecards that fit into optical shelves which are multiple rack unit chassis devices. The 2,600 km fiber span is constructed using fiber spools. Like in a typical production setting, we connect an amplifier node at every 80–100 km of fiber distance on this span. Overall, our testbed has 26 nodes in as many shelves where node 0 and node 25 have ROADM cards. Figure 11a shows one of the 26 shelves in our hardware testbed. We loaded this span with three bi-directional channels added and dropped at the ROADMs at the endpoints. The channels are roughly positioned at 25% 50% and 75% points of the C-band spectrum.

Field experiment setup. We verify our findings by repeating the lab tests in the production network of a regional ISP connecting universities, ISPWAN, in New York state. We were given access to ROADMs in two locations in ISPWAN. These two ROADMs are connected by a multi-hop optical path spanning $\approx$ 300 km. ISPWAN's optical line system is from the equipment manufacturer, VENDOR-B. Figure 11b shows one of shelves of VENDOR-B line system equipment used in ISPWAN.

6.1 Adding optical channels on fiber spans

We begin our experiments in the lab testbed. The optical line system in the lab consists of VENDOR-A equipment. Modifications to the VENDOR-A line system can be done using a GUI (Figure 16 in Appendix C.1). To provision a channel, we first create a software port and assign it a center frequency and spectral width. In all our lab testbed experiments, we use channels of spectral width 37.5 GHz, same as the pixel width in CHISEL's algorithm. We repeat the same steps for configuring ports on the ROADM at the far end of the span.

Equalizing power levels. After provisioning the new channel, the channel has to be built by the line system. Figure 16 in Appendix C.1 shows the GUI status while a channel is being built. Building the channel requires the proprietary vendor line system controller to communicate with all nodes in the span, including the amplifiers and the ROADMs about the newly provisioned channel. Finally, the power levels of all amplifiers on the span are *equalized* to ensure that the new spectrum on the fiber receives adequate amplification power. We note that building of the channel and equalizing its power consumes the bulk of the time to provision the channel. Depending on the vendor ecosystem and number of nodes on the span, it takes roughly five minutes to provision a channel.

6.2 Effect on existing data channels

Previous work has expressed concerns about the adverse effects of adding new wavelengths on existing channels in fiber spans [42]. Since CHISEL must provision channels to meet slicing requests, we set out to measure how addition and removal of channels on the testbed fiber span impacts the quality factor (Q-factor) of existing channels. Recall that the hardware testbed has 3 existing channels (six bidirectional channels) on it. We remove one of the channels by deleting the software port and tearing down the L1 connection between the software ports. We verify, by measuring receive power of the deleted channel at the endpoints of the span, that the channel has indeed been removed. We add the channel back soon after it has been removed. We repeat this experiment with a second channel. We find that the remaining channels on the span, regardless of their position in the spectrum, remain unaffected with both the addition and the removal of new channels (Figure 11c).

Finally, we repeat this experiment in the field. We note that ISPWAN uses a different optical line system than the one in our lab. However, this line system is also primarily programmed using a GUI. We add a wavelength at the ROADM located in CITY-B and drop the wavelength at the ROADM in CITY-A. During the process, the existing telemetry mechanism in the production network is gathering the performance statistics. Figure 11d shows the Q-factor of existing channels on the span (not including the wavelength we added). Confirming the findings from our lab experiment, the figure shows that all wavelengths on the path remain unaffected by the addition of the new channel.

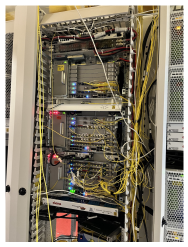
6.3 Latency of provisioning channels

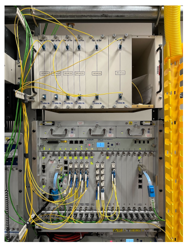
We used the equipment from two major vendors of optical line systems in WANs and found that both use GUI-based management tools for creating and deleting channels. These tools only allow configuring channels and equalizing them serially at every node on the span. In our experiments, the process of creating and equalizing channels took $\approx$5 minutes for spans with moderate number of nodes on the path. We set out to investigate if creating channels programmatically would significantly reduce the delay of provisioning wavelengths.

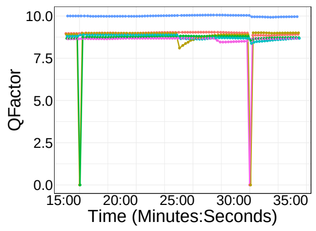
Benefits of automating provisioning. VENDOR-B provides a REST API for configuring their line system. Seldom used in practice by network operators, we leverage the REST interface to programmatically add optical channels in a lab setup of the VENDOR-B line system. Our tool first provisions the software port on the ROADM for a new channel and configures its position in the spectrum and spectral width. Our tool uses channels of 50GHz spectral width since it was the only available width on this ROADM. It then equalizes the power on nodes in the path. We repeat the provisioning of a channel rapidly using our tool and find that on average it takes 8 seconds seconds to add a new channel. Therefore automation shaves off most of the observed latency of provisioning channels seen by network operators.

Parallel equalization of power. In an attempt to reduce the latency of provisioning a channel, CHISEL's automatic provisioning tool equalizes the channels in parallel instead of sequentially. We find equalizing in parallel can further reduce the latency of provisioning wavelengths significantly in practice. We release our tool to aid operators in rapid provisioning of channels in their networks.

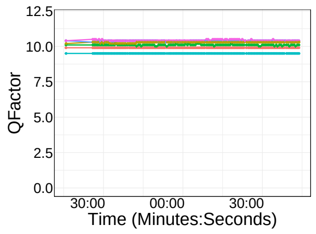
Contrast with heavily-loaded spans. Our findings in §6.1 are from networks with lightly loaded spectrum, both in the lab (Figure 11c) and in the field (Figure 11d). Large portions of the spectrum in these settings were free. To ensure that our findings hold in heavily loaded optical networks, we use


(a) Optical shelf (VENDOR-A).


(b) Optical shelf (VENDOR-B).



(c) Q-factor (lab experiment).



(d) Q-factor (field experiment).

Figure 11: Figure 11a one shelf of equipment that forms one node in the 26 node hardware testbed used by CHISEL. Figure 11c shows that the addition and removal of channels does not affect the Q-factor of existing wavelengths on the same fiber. Figure 11d shows the same effect in the field. Q-factor of the channel that is being added is not shown in Figure 11d.

CHISEL's tool to programmatically provision wavelengths that fill the optical spectrum on fiber and monitor the effect of adding these wavelengths on existing channels on the span. We provide details of this setup in Appendix C.1. Over time, we kept adding wavelengths to the span and monitoring the two channels that are connected to data sources. Figure 12 shows that even with increasing spectrum utilization, addition of new waves does not disrupt signal transmission on the existing wavelengths on the span. Figure 17 shows the fully-loaded spectrum at the end of our experiment. In summary, our experiments show that it is safe to provision wavelengths incrementally as instructed by CHISEL's algorithm.

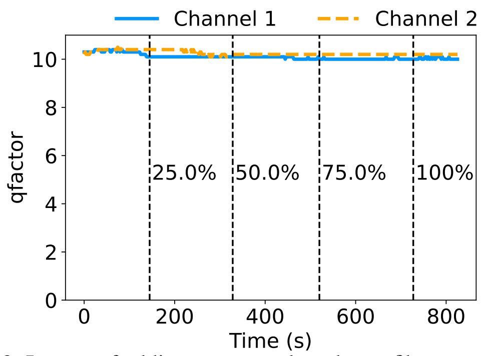


Figure 12: Impact of adding new wavelengths on fiber as a function of percentage spectrum occupied by existing channels.

7 CHISEL vs. wide-area traffic engineering

An alternative to optical slicing with CHISEL is using Layer 2 or Layer 3 tunnels for allocating bandwidth between endpoints in the WAN. This approach is used by WAN traffic engineering (TE) [18, 19, 23]. Cloud providers deploy TE to efficiently use their network infrastructure [18, 19, 23]. TE systems take demands between sources and destinations as input to solve a multi-commodity flow problem that allocates demands on network *tunnels*. Network tunnels are built by tunnelling protocols like MPLS [32], IP-in-IP [35], GRE [25] *etc.*At the optical layer, cloud WANs are point-to-point which means that the tunnel abstraction of Layer-2 and above is enabled by a series of one-hop optical links [37].

We contrast the two approaches of allocating bandwidth, CHISEL vs. WAN TE, along the axis of marginal hardware expense needed to provision the bandwidth. We assume that CHISEL and network TE take the same hop paths in the network with one difference — CHISEL constructs an end-to-end optical slice over the path while TE constructs a packet switched tunnel over the path. We compare the hardware cost of allocating slices using CHISEL with that of allocating equal bandwidth using TE. We implement the most commonly used TE formulation that maximizes network throughput [18, 23]. Both the TE algorithm and CHISEL allocate the bandwidth requests and we compute the number of router ports required to provision the slice requests for both. TE tunnels carry traffic that undergoes OEO conversion at every hop by design and thus consumes router ports not only for adding (dropping) traffic at the source (destination) router but also at all intermediate hops. In contrast, an all-optical slice from CHISEL only needs router ports at the slice source and destination router.

Figure 13 shows the number of network ports used in provisioning different number of slices in the cloud WAN. We find that allocating bandwidth using CHISEL's slices not only improves performance under strict QoS requirements (Figure 3) but is also significantly cheaper in terms of the hardware cost of provisioning network bandwidth compared with cloud WAN TE. On average, CHISEL-provisioned slices consume $2.6-3.3$X fewer router ports than an equivalent TE tunnel.

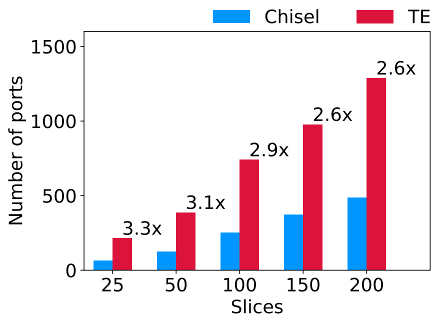


Figure 13: Hardware cost reduction using CHISEL vs. TE.

8 Giving tenants access to optical slices

Once CHISEL computes optimal allocations to slice requests (§5), it provisions the slices on the optical line system (§6).

The final step of provisioning an end-to-end slice is to give the network tenants access to the provisioned optical slice. In this section we discuss the mechanism CHISEL uses to furnish optical slices to its clients. CHISEL exposes the slice's allocated bandwidth using ports on WAN routers. Since slice allocations are dynamic, CHISEL needs a mechanism to map the slice to router ports dynamically. If this mapping is static, CHISEL would need to reserve router ports and potentially waste the allocated ports if they were not needed.

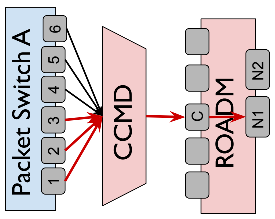


Figure 14: Mapping router ports to optical spectrum.

Dynamically mapping router ports to slices. CHISEL uses *colorless*, *directionless* multiplexers and demultiplexers (CCMDs) which connect multiple ports of WAN routers to the client port of the ROADM using coherent optical transponders. Each of these transponders are tuned to generate signals of specific wavelengths. Once CHISEL has provisioned the slice, it can program the ROADM to select wavelengths incident on the client port that correspond to the appropriate router ports. For instance, Figure 14 shows that the ROADM can select ports 1, 2, 3 to drop traffic from those ports into the allocated optical slice. While the remaining router ports (4, 5, 6) are connected to the same CCMD and ROADM client port C, the ROADM can select alternate routes for traffic from those ports by dropping their channels on a different network port of the ROADM (*e.g.,* $N2$). CCMDs are common and relatively inexpensive equipment that can be leveraged by CHISEL to make allocated spectrum available on router ports.

8.1 Sub-rate optical slices with OTN switching

Our evaluation of CHISEL (§6) allocates slices at the granularity of 37.5 GHz of the optical spectrum. A slice of this spectral width can carry up to 200 Gbps of traffic depending on the signal quality of the channel in fiber, length of the fiber path and modulation format with which the channel is encoded. While we expect cloudified access networks require large optical slices for service-oriented traffic classes (*e.g.,* voice over IP, tele-presence), some clients of CHISEL might want to carve smaller chunks of the optical spectrum.

We discuss whether CHISEL can be used to to carve smaller bandwidth slices. We note that the CHISEL spectrum allocation algorithm (Algorithm 1) is agnostic to pixel width and can compute allocations of spectral widths as small as 6.25 GHz. This is in line with the the capability of state-of-the-art flexgrid ROADMs that can only allocate spectrum in multiples of 6.25 GHz pixels. The challenge of allocating spectrum at this fine granularity lies in the lower bound on the spectrum expected at the client port of a ROADM (Figure 14). The minimum width of spectrum that the client port of a ROADM can add or drop on the fiber is restricted by the minimum spectrum of light produced by optical transponders that connect router ports to ROADM client ports. This lower bound is 37.5 GHz for the line system equipment we work with.

However, the problem of carving *sub-rate* slices *i.e.,* slices that need bandwidth lower than that one a single wavelength on fiber, can be solved using *Optical transport networking* (OTN) switches. OTN switches can multiplex sub-rate channels from router ports in the time domain to feed a larger wavelength to the ROADM client port. The use of OTN switches will not change how the CHISEL algorithm functions but will change how sub-rate slices are made available to clients.

9 Related Work

We now place CHISEL in the context of related work:

Optical reconfigurability in WANs. Researchers have developed techniques to reconfigure the optical backbone to improve throughput [38] and reliability [42]. For instance, RADWAN [38] adapts data rates of long-haul links in response to changes in signal quality. ARROW [42] recovers from fiber cuts by migrating wavelengths to alternate paths. CHISEL uses similar ideas of optical reconfiguration as these systems but solves a very different challenge: that of optical spectrum slicing. Previous work in networked systems has largely ignored the problem of spectrum management, especially at the scale and time budget demanded by network slicing. OWAN [21] schedules transfers on existing optical circuits in the WAN to meet long-running bulk transfers. In contrast, CHISEL establishes new circuits in operational networks for slicing without impacting engineered traffic. The resources allocated by OWAN are optical circuits whereas the resource allocated by CHISEL is optical spectrum, similar to radio spectrum allocation in RAN slicing. As a result, challenges faced by CHISEL are around fast spectrum allocation, spectral fragmentation and hardware support for dynamic spectrum allocation in WANs.

Optical network design. ISPs have studied the design of optical networks in depth [5, 6, 7, 10, 15, 27, 28]. Shoofly [37] solves a similar problem in the context of cloud providers. Shoofly uses optical bypass to lower the cost of provisioning capacity in cloud WANs. In contrast, CHISEL is solving a runtime problem and not a design-time problem.

Spectral allocation. Optical communication researchers have studied the problem of spectral allocation and fragmentation in fiber [41]. However, they approach spectral allocation as a network capacity provisioning problem that is done infrequently. Authors of [31] discuss the problem of assigning wavelength to circuits in a traffic demand matrix without considering existing traffic on the network. In contrast, CHISEL considers existing traffic on the network and also models the hardware constraints into the algorithm. In contrast, CHISEL works in tandem with traffic engineering and allocates spectrum at fine granularity for optical slices. Our goal with CHISEL is to not replace existing network provisioning or network traffic engineering. Instead, CHISEL introduces a new way of rapidly allocating bandwidth at the physical layer for clients who need it.

Elastic spectrum slicing. Researchers have established elastic bandwidth circuits using optical frequency division multiplexing (OFDM) modulation format and variable bandwidth cross connects (OXC) [22]. This work builds a mechanism to dynamically change the bandwidth associated with a circuit in the spectrum domain. It is complementary to our work since CHISEL's algorithm determines how much bandwidth to allocate to each circuit and can be modified to work with a different circuit switch's characteristics.

Wide-area traffic engineering TE is a related problem that allocates bandwidth along tunnels in WANs using a centralized, software-defined controller [19, 23, 36].

RAN slicing. There has been a lot of work on slicing the radio access network and algorithms for slicing with high efficiency [13]. CHISEL is complementary to RAN slicing and in fact extends the reach of RAN slicing to WANs.

10 Conclusions

We develop CHISEL, a system that creates slices of optical spectrum on fiber. CHISEL's algorithm computes bandwidth optimal slice allocations that consume $2.6-3.3$X fewer router ports compared to Layer-3 traffic engineering. CHISEL dynamically allocates spectrum slices while limiting spectral fragmentation. CHISEL programs spectral allocations automatically within seconds without impacting existing data-carrying channels on fiber. We have released CHISEL's experimental data, implementation and automation tools [3].

References

[1] 3GPP. 3GPP TS 23.501 version 16.6.0 Release 16. Technical report, 2020.

[2] 3GPP. Service requirements for enhanced V2X scenarios. Technical report, 2020.

[3] Anonymous. Chisel Code and Data. http://opticalslice.network.

[4] Venkat Arun, Mina Tahmasbi Arashloo, Ahmed Saeed, Mohammad Alizadeh, and Hari Balakrishnan. Toward formally verifying congestion control behavior. In *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*, SIGCOMM '21, page 1–16, New York, NY, USA, 2021. Association for Computing Machinery.

[5] Ajay Kumar Bangla, Alireza Ghaffarkhah, Ben Preskill, Bikash Koley, Christoph Albrecht, Emilie Danna, Joe Jiang, and Xiaoxue Zhao. Capacity planning for the google backbone network. 2015.

[6] M. Birk, G. Choudhury, B. Cortez, A. Goddard, N. Padi, A. Raghuram, K. Tse, S. Tse, A. Wallace, and K. Xi. Evolving to an sdn-enabled isp backbone: key technologies and applications. *IEEE Communications Magazine*, 2016.

[7] A. Brzezinski and E. Modiano. Dynamic reconfiguration and routing algorithms for ip-over-wdm networks with stochastic traffic. *Journal of Lightwave Technology*, 23(10):3188–3205, 2005.

[8] Microsoft News Center. AT&T to run its mobility network on Microsoft's Azure for Operators cloud, delivering cost-efficient 5G services at scale, 2021.

[9] Mayur Channegowda, Reza Nejabati, and Dimitra Simeonidou. Software-defined optical networks technology and infrastructure: Enabling software-defined optical network operations (invited). *J. Opt. Commun. Netw.*, 5(10):A274–A282, Oct 2013.

[10] Angela L Chiu, Gagan Choudhury, George Clapp, Robert Doverspike, Mark Feuer, Joel W Gannett, Janet Jackel, Gi Tae Kim, John G Klincewicz, Taek Jin Kwon, et al. Architectures and protocols for capacity efficient, highly dynamic and highly resilient core networks. *IEEE/OSA Journal of Optical Communications and Networking*, 2011.

[11] Cisco. What is MPLS - Multiprotocol Label Switching. https://www.cisco.com/c/en/us/products/ios-nx-os-software/multiprotocol-label-switching-mpls/index.html, (Accessed on 2021-01-20).

[12] N. Cvijetic, A. Tanaka, P. N. Ji, K. Sethuraman, S. Murakami, and T. Wang. SDN and OpenFlow for dynamic flex-grid optical access and aggregation networks. *Journal of Lightwave Technology*, 32(4):864–870, Feb 2014.

[13] A. Destounis, G. Paschos, S. Paris, J. Leguay, L. Gkatzikis, S. Vassilaras, M. Leconte, and P. Medagliani. Slice-based column generation for network slicing. pages 1–2, 2018.

[14] Mark Filer, Hacene Chaouch, and Xiaoxia Wu. Toward transport ecosystem interoperability enabled by vendor-diverse coherent optical sources over an open line system. *Journal of Optical Communications and Networking*, 10(2):A216–A224, 2018.

[15] Jennifer Gossels, Gagan Choudhury, and Jennifer Rexford. Robust network design for ip/optical backbones. *IEEE/OSA Journal of Optical Communications and Networking*, 11(8):478–490, 2019.

[16] Gurobi. GUROBI Optimization. https://www.gurobi.com/, (Accessed on 2019-10-02).

[17] Pete Heist. IRTT (Isochronous Round-Trip Tester) . https://github.com/heistp/irtt, 2021.

[18] Chi-Yao Hong, Srikanth Kandula, Ratul Mahajan, Ming Zhang, Vijay Gill, Mohan Nanduri, and Roger Wattenhofer. Achieving high utilization with Software-driven WAN. *SIGCOMM*, 2013.

[19] Chi-Yao Hong, Subhasree Mandal, Mohammad Al-Fares, Min Zhu, Richard Alimi, Kondapa Naidu B., Chandan Bhagat, Sourabh Jain, Jay Kaimal, Shiyu Liang, Kirill Mendelev, Steve Padgett, Faro Rabe, Saikat Ray, Malveeka Tewari, Matt Tierney, Monika Zahn, Jonathan Zolla, Joon Ong, and Amin Vahdat. B4 and after: Managing hierarchy, partitioning, and asymmetry for availability and scale in google's software-defined wan. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*, SIGCOMM '18, page 74–87, New York, NY, USA, 2018. Association for Computing Machinery.

[20] Chris R. Jackson, Reza Nejabati, Fernando Agraz, Albert Pagès, Michael Galili, Salvatore Spadaro, and Dimitra E. Simeonidou. Demonstration of the benefits of SDN technology for all-optical data centre virtualisation. *Optical Fiber Communication Conference*, page Tu3L.3, 2017.

[21] Xin Jin, Yiran Li, Da Wei, Siming Li, Jie Gao, Lei Xu, Guangzhi Li, Wei Xu, and Jennifer Rexford. Optimizing bulk transfers with software-defined optical WAN. *SIGCOMM*, 2016.

[22] Bartłomiej Kozicki, Hidehiko Takara, Yukio Tsukishima, Toshihide Yoshimatsu, Kazushige Yonenaga, and Masahiko Jinno. Experimental demonstration of spectrum-sliced elastic optical path network (slice). *Opt. Express*, 18(21):22105–22118, Oct 2010.

[23] Umesh Krishnaswamy, Rachee Singh, Nikolaj Bjørner, and Himanshu Raj. Decentralized cloud wide-area network traffic engineering with BLASTSHIELD. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*, pages 325–338, Renton, WA, April 2022. USENIX Association.

[24] Umesh Krishnaswamy, Rachee Singh, Paul Mattes, Paul-Andre C Bissonnette, Nikolaj Bjørner, Zahira Nasrin, Sonal Kothari, Prabhakar Reddy, John Abeln, Srikanth Kandula, Himanshu Raj, Luis Irun-Briz, Jamie Gaudette, and Erica Lan. OneWAN is better than two: Unifying a split WAN architecture. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*, pages 515–529, Boston, MA, April 2023. USENIX Association.

[25] Tony Li, Dino Farinacci, Stanley P. Hanks, David Meyer, and Paul S. Traina. Generic Routing Encapsulation (GRE). RFC 2784, March 2000.

[26] Affirmed Networks. Network slice management . Technical report, 2020.

[27] P. Papanikolaou, K. Christodoulopoulos, and E. Varvarigos. Joint multi-layer survivability techniques for ip-over-elastic-optical- networks. *IEEE/OSA Journal of Optical Communications and Networking*, 9(1):A85–A98, 2017.

[28] P. Papanikolaou, K. Christodoulopoulos, and E. Varvarigos. Optimization techniques for incremental planning of multilayer elastic optical networks. *IEEE/OSA Journal of Optical Communications and Networking*, 10(3):183–194, 2018.

[29] Larry L. Peterson and Bruce S. Davie. *Computer Networks, Fifth Edition: A Systems Approach*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 5th edition, 2011.

[30] Petar Popovski, Kasper F. Trillingsgaard, Osvaldo Simeone, and Giuseppe Durisi. 5G Wireless Network Slicing for eMBB, URLLC, and mMTC: A Communication-Theoretic View. In *IEEE Access*, 2018.

[31] R. Ramaswami and K.N. Sivarajan. Routing and wavelength assignment in all-optical networks. *IEEE/ACM Transactions on Networking*, 3(5):489–500, 1995.

[32] Eric C. Rosen, Arun Viswanathan, and Ross Callon. Multiprotocol label switching architecture, January 2001. RFC 3031.

[33] Claude E. Shannon. Two-way communication channels. 1961.

[34] D. Simeonidou, R. Nejabati, and S. Azodolmolky. Enabling the future optical Internet with OpenFlow: A paradigm shift in providing intelligent optical network services. *International Conference on Transparent Optical Networks*, pages 1–4, June 2011.

[35] W. Simpson. IP in IP Tunneling. RFC 1853, October 1995.

[36] Rachee Singh, Sharad Agarwal, Matt Calder, and Paramvir Bahl. Cost-effective cloud edge traffic engineering with cascara. In *18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*, pages 201–216. USENIX Association, April 2021.

[37] Rachee Singh, Nikolaj Bjørner, Sharon Shoham, Yawei Yin, John Arnold, and Jamie Gaudette. Cost-effective capacity provisioning in wide area networks with shoofly.

In *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*, SIGCOMM '21, page 534–546, New York, NY, USA, 2021. Association for Computing Machinery.

[38] Rachee Singh, Manya Ghobadi, Klaus-Tycho Foerster, Mark Filer, and Phillipa Gill. Radwan: Rate adaptive wide area network. ACM SIGCOMM, August 2018.

[39] The Internet Topology Zoo . The Internet Topology Zoo. http://www.topology-zoo.org/dataset.html, (Accessed on 2013-03-02).

[40] Jin Y. Yen. An algorithm for finding shortest routes from all source nodes to a given destination in general networks. *Quarterly of Applied Mathematics*, 27:526–530, 1970.

[41] Yawei Yin, Mingyang Zhang, Zuqing Zhu, and S. J. B. Yoo. Fragmentation-aware routing, modulation and spectrum assignment algorithms in elastic optical networks. In *Optical Fiber Communication Conference/National Fiber Optic Engineers Conference 2013*, page OW3A.5. Optica Publishing Group, 2013.

[42] Zhizhen Zhong, Manya Ghobadi, Alaa Khaddaj, Jonathan Leach, Yiting Xia, and Ying Zhang. Arrow: Restoration-aware traffic engineering. In *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*, SIGCOMM '21, page 560–579, New York, NY, USA, 2021. Association for Computing Machinery.

A Optical vs. Electrical paths

In this section we provide an in-depth detail about the setup for Figure 3 in §2. For this experiment, we gained access to two machines connected to two routers in ISPWAN, a NY state based WAN. We established two routes between the two routers, one that goes over a routed path and one that is an all-optical path. The routed path has 4 intermediate routers. The all-optical path bypasses conversion to the electrical domain entirely. The length of the fiber path in between these routers is the same — roughly 500 km.

We set out to contrast the performance of the two paths using isosynchronous RTT measurements [17]. We start an irtt server on one of the machines. We start measurement of both paths from the other machine at the same time and run it for roughly 24 hours. We measure every 500 ms and track the mean and standard deviation of RTT between the machines. Figure 3 in §2 shows that the routed path performs worse compared to the all-optical path consistently. Moreover, the jitter on the routed path is significantly higher than the optical path. This paves the way for CHISEL's design. CHISEL implements slicing in the wide-area by carving chunks of the optical spectrum.

B CHISEL algorithm evaluation

In this section we characterize the slices carved by CHISEL. This is in addition to the experiments in §5.

B.1 Modulation formats of slices

Figure 15 shows the modulation formats of slices allocated by CHISEL. Most slices are short enough to have the modulation format of 8-QAM (150G), some slices with shorter fiber paths have the modulation format 16-QAM (200G) and some have the modulation format of QPSK (100G).

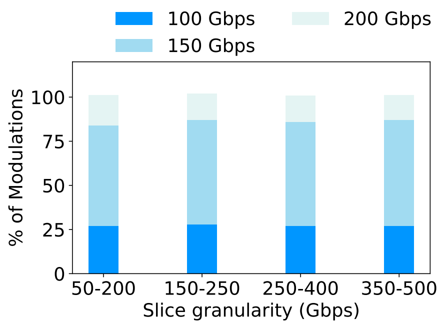


Figure 15: Modulation formats of slices.

B.2 Comparison with TE

We provide a detailed description of the setup for comparing CHISEL with wide-area traffic engineering in §7. We evaluate TE as an alternative to CHISEL on the same network graph with the same set of slice requests. The slice requests between router pairs and their corresponding bandwidths form the demand matrix for TE. We compute k-shortest paths for both TE and CHISEL. For TE, we solve the most common optimization formulation — the max throughput edge formulation [18].

TE computes allocations along paths it considers tunnels and CHISEL allocates spectrum along paths it considers optical slices. While both can meet the demands, we compare the number of router ports the two approaches will use across the entire network. We show in Figure 13 that TE requires a significantly larger number of router ports. This large difference in ports consumed by CHISEL and TE arises from the fact that TE tunnels are constructed over point to point optical links in the WAN. Therefore, traffic carried over TE tunnels consumes router ports at ingress as well as egress at each intermediate router hop in the tunnel.

C Hardware Evaluation

We evaluated CHISEL using the optical line system from two major equipment vendors: VENDOR-A and VENDOR-B. We found that line systems from both vendors were primarily programmed in the field using graphical interfaces.

C.1 VENDOR-A hardware testbed

The VENDOR-A lab testbed consisted of a point-to-point optical link typical of cloud WANs. The two end points of the link were ROADMs with add/drop capability. In between the two ROADMs were amplification sites. Overall, the two ROADMs were connected using fiber spools of 2,600 km. The testbed took 26 shelves in our lab and Figure 11a shows one of the shelves. Configuration on the VENDOR-A line system was done over a GUI. Figure 16 shows the status of the GUI as we added a channel between the ROADMs and waited for the line system to *build* the channel.

C.2 VENDOR-B hardware testbed

We used the VENDOR-B line system in our field experiment with sysname. This line system had a GUI similar to VENDOR-A's GUI for configuration but it also provided a REST API to make the same configuration.

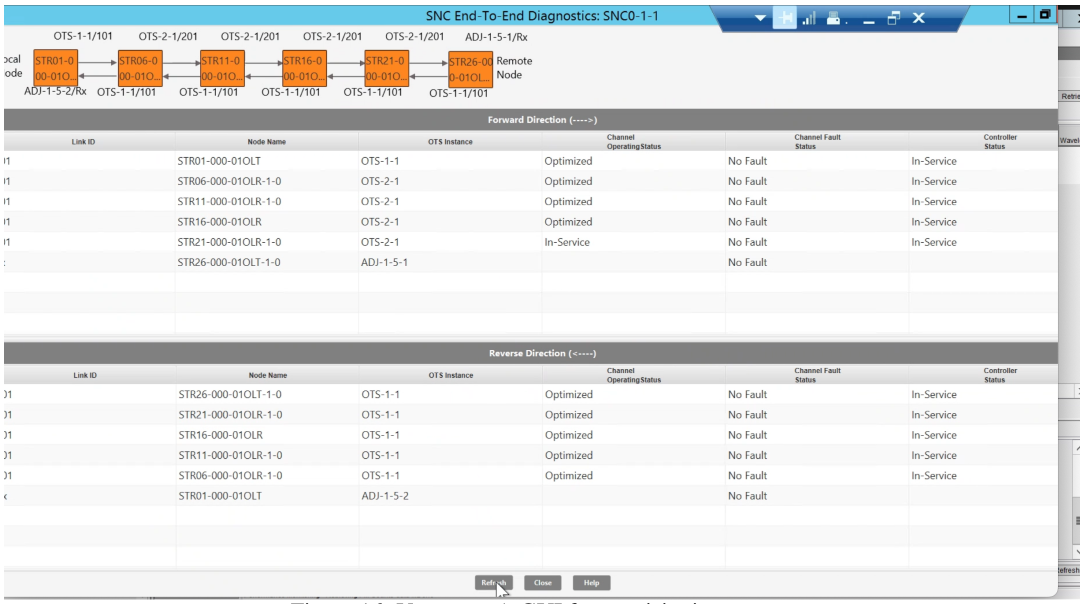
We also used VENDOR-B line system in the lab setting to test automatic addition and removal of optical slices through code. To do this, we created an optical span in the lab with two amplifiers and ROADM ports on each end of the span. This span had two legitimate light sources and configured channels. Our goal was fully populate the spectrum on this span. To do this, we used an amplifier as a source of noise channels to the ROADM. Once we had the amplifier setup as a light source, we developed the software tool to populate the entire spectrum on the span.

C.3 Fully loading the spectrum

We automated the creation of slices on VENDOR-B ROADM cards using the REST interface exposed by the card. The creation of a slice involves three main steps, first, we create a channel on the frequency picked by CHISEL in either the *add/drop* or *passthrough* modes, second, we enable the components of the channel so that they start receiving or sending signals and finally, we equalize the channel. We configure the terminal ROADMs in the setup in add/drop mode. the passthrough mode is for the intermediate ROADMs in the path to enable light to pass through them.

We now describe the process in detail for a single ROADM. The channel creation step invokes a POST request with the mode of the channel which can either be add/drop or passthrough and the spectrum frequency picked by CHISEL. In the second step we issue a GET request to list the components of the channel and enable them to be in service. Finally, we make a GET request to start the equalization process. During this, we periodically (1s) monitor the power level of the newly provisioned wavelength using a different GET endpoint. The first two steps complete within 2 seconds and the third step completes within 5-7 seconds.

We run the script in a loop to completely fill the spectrum of the ROADM form VENDOR-B. We can see in figure 17 that all the channels that have been provisioned are equalized to have power higher than -16dBm.


SNC End-To-End Diagnostics: SNC0-1-1

Forward Direction (---->)

Link ID	Node Name	OTS Instance	Channel Operating Status	Channel Fault Status	Controller Status
1	STR01-000-01OLT	OTS-1-1	Optimized	No Fault	In-Service
1	STR06-000-01OLR-1-0	OTS-2-1	Optimized	No Fault	In-Service
1	STR11-000-01OLR-1-0	OTS-2-1	Optimized	No Fault	In-Service
1	STR16-000-01OLR	OTS-2-1	Optimized	No Fault	In-Service
1	STR21-000-01OLR-1-0	OTS-2-1	In-Service	No Fault	In-Service
:	STR26-000-01OLT-1-0	ADJ-1-5-1		No Fault	

Reverse Direction (<----)

Link ID	Node Name	OTS Instance	Channel Operating Status	Channel Fault Status	Controller Status
01	STR26-000-01OLT-1-0	OTS-1-1	Optimized	No Fault	In-Service
01	STR21-000-01OLR-1-0	OTS-1-1	Optimized	No Fault	In-Service
01	STR16-000-01OLR	OTS-1-1	Optimized	No Fault	In-Service
01	STR11-000-01OLR-1-0	OTS-1-1	Optimized	No Fault	In-Service
01	STR06-000-01OLR-1-0	OTS-1-1	Optimized	No Fault	In-Service
<	STR01-000-01OLT	ADJ-1-5-2		No Fault	

Figure 16: VENDOR-A GUI for provisioning waves.

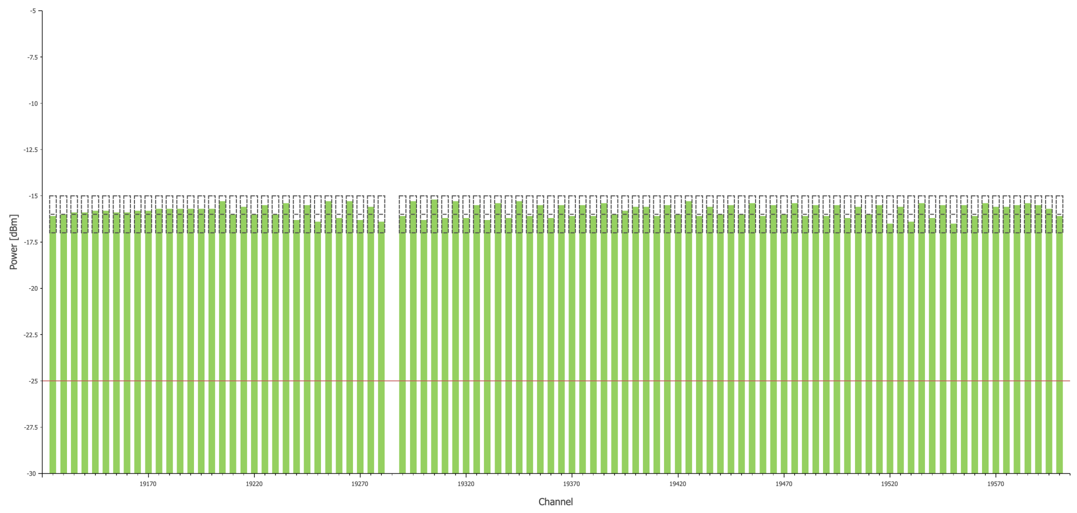


Figure 17: GUI on the ROADM showing that the complete spectrum is filled. The power levels of all the created channels are around -16dBm. The power levels of existing channels carrying data are not affected throughout the experiment.