1 Introduction

Data centers (DCs) are a critical piece of today's networked applications in both private [] and public sectors []. The key factors that have driven this trend are economies of scale, reduced management costs, better utilization of hardware via statistical multiplexing, and the ability to elastically scale applications in response to changing workload patterns.

A robust *datacenter network fabric* is fundamental to the success of DCs and to ensure that the network does not become a bottleneck for high-performance applications [**?**]. In this context, DC network design must satisfy several goals: high performance (e.g., high throughput and low latency) [**?**, **?**], low equipment and management cost [**?**, **?**], robustness to dynamic traffic patterns [**?**, **?**, **?**, **?**], incremental expandability to add new servers or racks [**?**, **?**], and other practical concerns such as cabling complexity, and power and cooling costs [**?**, **?**, **?**].

In this context, traditional DC network architectures can be broadly divided into two categories: (1) *overprovisioned* fabrics (e.g., fat-trees or multi-stage Clos networks) with full-bisection bandwidth []; (2) *oversubscribed* fabrics (e.g., 'leaf-spine") where links may be oversubscribed [**?**]; and (3) *augmented* fabrics where an oversubscribed "core" is augmented with reconfigurable wireless [**?**, **?**] or optical links [**?**]. The first two classes offer extreme cost-performanc tradeoffs as overprovisioning incurs high cost and oversubscription can lead to poor performance [**?**]. While augmented approaches show promise, existing solutions are incremental and only offer limited flexibility. Specifically, optical solutions are only effective for workloads amenable to bipartite matchings between top-of-rack (ToR) switches and introduce a single point of failure []. Similarly, existing wireless technologies are inherently limited by interference and range constraints []. Furthermore, all current architectures incur high cabling cost and complexity [**?**]. (We elaborate on these in §2.)

Our vision: In this proposed research, we consider an *extreme* design point. Instead of trying to incrementally improve the poor cost-performance tradeoffs, high cabling complexity, and limited flexibility of current DC architectures, we envision a *flexible*, *all-wireless* inter-rack fabric.

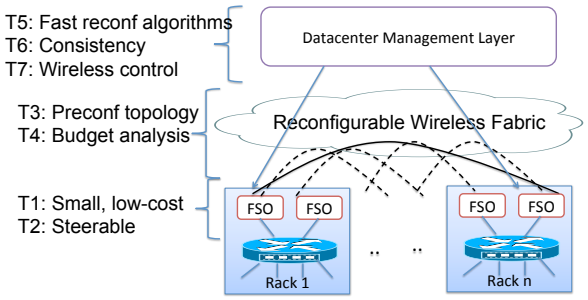


Figure 1: Overview of the FireFly vision

Figure 1 shows a conceptual overview of our vision called FireFly.[1] Each top-of-rack (ToR) switch has *flexible* wireless links that can be reconfigured to connect to other ToRs. The DC management layer reconfigures the topology to adapt to current traffic workloads. This vision offers several benefits that current DC architectures do not offer (§2). First, topological flexibility (if done right) provides a low-cost solution (e.g., fewer switches, links) that obviates the need for overprovisioning. Second, by virtue of being wireless, FireFly eliminates cabling complexity and attendant operational overheads (e.g., obstructed cooling) [**?**] altogether. Third, this vision can facilitate new and radical DC topology structures that would otherwise remain "paper designs" due to cabling complexity [**?**]. Finally, flexibility can reduce energy costs [**?**, **?**] and can enable administrators to incrementally expand the DC infrastructure to add more racks and machines as needed [**?**].

Research plan and Intellectual Merit: To realize the all-wireless vision outlined above, however, we need to look beyond traditional radio-frequency (RF) based (e.g., 60GHz) wireless solutions as they are fundamentally constrained in terms of range, capacity, and interference. To this end, we rely on *Free-Space Optical communications* (FSO) as it can offer very high data rates (tens of Gbps) over long ranges (≈100m) using low transmission power and with low interference footprint.

The unique characteristics of our approach—FSO-based inter-rack links, all-wireless, and topology flexibility—raises novel algorithmic, networking, and system-level challenges along three thrusts (Figure 1):

- Datacenter scale deployments impose new form-factor, cost, and steerability requirements for FSOs that are fundamentally different from their traditional long-haul use-cases. Thus, we need to develop cost-effective solutions (**??**) that can be steered at fine-grained timescales(**??**).

[1] FireFly stands for Free-space optical Inter-Rack nEtwork with high FLexibilitY.

- We need new algorithmic foundations to reason about flexible network design (**??**). Furthermore, the physical and geometric constraints of the steering mechanisms raise new challenges for architecting the DC subject to budget constraints (**??**).
- Our vision imposes new network management challenges as well. In particular, we need novel algorithms for joint topology reconfiguration and traffic engineering (**Task 5**), new consistency abstractions that guarantee connectivity and performance when links may be in flux (**Task 6**). Furthermore, we need new mechanisms to replace traditional wired control channels (**Task 7**).

Team Qualifications: Our team comprising three computer scientists and one mechanical engineer has complementary expertise spanning the domains of network algorithmics [], wireless networking [], network management [], software-defined networking [], and laser-based optical technologies []. Thus, our team is uniquely positioned to tackle the aforementioned challenges. The PIs have an established history of collaboration [] and outreach activities [], and the proposed research will further strengthen these existing collaborations.

2 Motivation and Research Overview

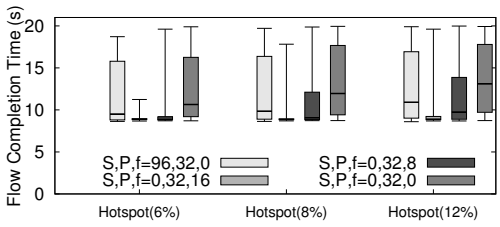
As discussed earlier, there are three key aspects to the FireFly vision for DC networks: *flexibility*, *wireless*, and the use of *free-space optics*. We argue why each of these aspects is needed before providing an overview of our proposed research.

2.1 Flexibility reduces infrastructure needs

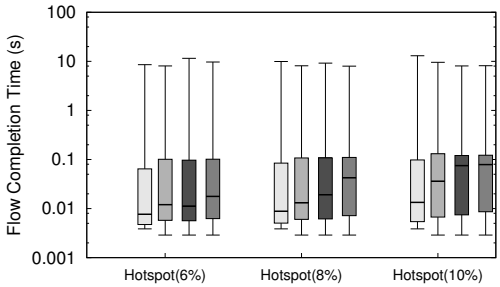
Measurement studies show that DC traffic exhibits low utilization coupled with hotspots of inter-rack activity with a few "heavy" flows [**?**, **?**, **?**]. In light of this, traditional *static* DC designs turn out to be extreme points in the cost-performance tradeoff. Fat-tree [**?**] like solutions that provide full bisection bandwidth are overprovisioned and lead to low link utilization and high cost. On the other hand, conventional leaf-spine architectures are low cost but suffer poor performance as many links are oversubscribed [**?**]. As observed in prior work [], these measurement studies naturally suggest that a *flexible* DC topology can potentially eliminate the need for overprovisioning [**?**].

To provide the basis for this intuition, we consider an abstract model of a DC with n ToR switches and S "core" switches that transit traffic between ToR switches. Each rack has l servers and each switch (ToR and core) has P ports. For the ToR switches, l ports connected to the machines, and $P - l$ for inter-rack connections. Furthermore, out of the $P - l$ inter-rack ports, suppose f can be *flexibly* rewired. For the non-ToR switches, all P ports are connected to other switches. Now, for a given n, l, and P determined by technology or deployment constraints, we can define a broad spectrum of DC designs by specifying the 3-tuple $\langle S, P, f \rangle$. For instance, with $P = 2 \times l$ that is typical of DC designs today, $\langle 1.5n, 2l, 0 \rangle$ corresponds to traditional *static* fat-tree designs and $\langle 0, 2l, l \rangle$ captures an extreme FireFly point with no "core" switches and full flexibility.

Of course, for a given $\langle S, P, f \rangle$, we also need to specify the topology created by the "static" and "dynamic" links. Rather than explore all possible graph structures for the *static* links, for brevity we only consider *random regular graphs* building on the observation that they provide close-to-optimal performance [**?**]. Thus, we randomly wire the "static" links between the ToR and non-ToR switches. Based on our preliminary work, we use a greedy algorithm for the dynamic links where we simply add "short-cut" links between racks with the highest inter-rack demands [**?**].



(a) Long-lived flows



(b) Short-lived flows

Figure 2: A flexible DC provides similar or better performance relative to overprovisioned networks with much less infrastructure.

Given this abstraction, we evaluate the performance of different DC designs by custom extensions to the `htsim` packet-level simulator [**?**]. Building on the prior DC

measurements, we consider realistic DC traffic workloads [**?**], wtih a baseline *uniform* traffic matrix between racks consisting of short-lived flows with an average size of XXXFillmeXXX. We augment this baseline with "elephant" or large flows between a subset of "hotspot" racks with an average size of XXXFillmeXXX. In this configuration, `Hotspot-X` refers to X% of the rack-pairs having elephant flows.

As an illustrative example, we consider the scenario with n=64 racks with P=32 and l=16 in Figure 2 with each link having XXXFillmeXXX Gbps. The result shows a box-and-whiskers plot with the min, max, and quantiles of the the flow completion times for different DC designs. Here, $\langle 96, 32, 0\rangle$ is a static regular random graph that provides full-bisection bandwidth with $S = 96$ "core" switches. We consider three flexible designs with f = 0, $P - l = 16$ (i.e., fully flexible), and $\frac{P-l}{2} = 8$. The key takeaway here is that flexible designs achieve near-optimal or better performance with significantly less infrastructure requirements. Specifically, the $\langle 0, *, *\rangle$ configurations uses 1.5× fewer switches and edges, but achieve very good performance. For instance, $\langle 0, 8\rangle$ configuration increases the median and 75%-ile of the flow completion time for short flows only by XXXFillmeXXXand XXXFillmeXXXrespectively but *decreases* that for long flows by XXXFillmeXXXand XXXFillmeXXXrespectively. We have also confirmed that these results qualitatively hold for other DC configurations as well; e.g., with n =256 racks and P=32 the corresponding improvements are XXXFillmeXXX.

2.2 Case for all-wireless and FSO

Why wireless? To realize a flexible inter-rack fabric, conceptually we need a reconfigurable "patch-panel" between pairs of racks [**?**]. Of course, such a big-switch abstraction is infeasible: (1) it requires very high fanout ($n \times f$, where n is the number of racks and f is the number of flexible links at each ToR) and backplane switching capacity; and (2) the cabling complexity would be prohibitive and create operational overheads w.r.t failures, and cooling/airflow considerations [**?**]. For similar reasons, traditional optical switching is also not viable. Finally, this giant switch introduces a single-point of failure [**?**, **?**, **?**]. To avoid the need for such a massive switch, we turn to *reconfigurable wireless* links between the ToR switches.[2]

Why FSO? The seemingly natural solution then is traditional radio-frequency (RF) wireless technologies (e.g., 60GHz). Unfortunately, these have many fundamental performance limitations: (1) RF links produce a large interference footprint due to a wide beamwidth [**?**]. [SD: This blames one paper only. This does not argue why we are discounting every form of RF. Recall the comment from hotnets reviewer. I can fix this. But the real argument is somewhat involved – more than a oneliner.] (2) The beam-steering technologies to implement flexibility are slow and inaccurate [**?**] and may additionally increase the interference footprint [**?**]; [SD: Need to rephrase, overly sweeping.] and (3) The data rates of RF links fall off rapidly with distance [**?**] and the use of higher transmit power to increase the range will increse interference and is limited by regulations []. To overcome these limitations, we leverage a somewhat non-standard wireless technology—free-space optics (FSO) that uses modulated visible or infrared (IR) laser beams [**?**].[3] We elaborate on the advantages of FSO vs. RF in Section 3.

2.3 Architecture and Proposed Research

Combining the above arguments leads us to the architecture in Figure 1. We eliminate the need for a wired backbone network and rely on a reconfigurable FSO-based wireless fabric. Each ToR switch is equipped with a pre-specified number of FSO devices and each FSO device assembly is capable of precise and fast steering to connect to target FSO devices in other ToRs. To establish an *obstruction-free* optical path, the space above the racks is a natural choice for laser propagation as this space is roughly free from obstruction [**?**]. The FSO transceivers will be anchored on the top of the rack and connected to the ToR switch. To ensure that the devices themselves do not obstruct each other, we propose use of ceiling mirrors as in [**?**] (and possibly additional mirrors on the beam path).

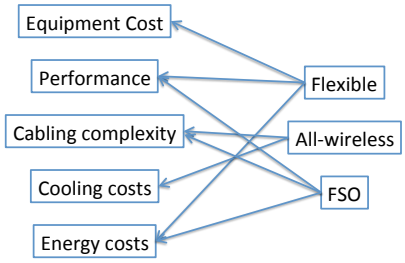


Figure 3: Key ideas underlying the FireFly vision and how they benefit different of DC datacenter considerations from Section 1

[2]Note that we are not proposing a fully wireless data center [**?**]; our focus is on the "inter-rack' fabric.

[3]Unlike traditional optical links, the laser beam in FSO is not enclosed in a glass fiber, but transmitted through the air (and hence "free space").

The DC management layer intelligently reconfigures these devices to adapt to changing network requirements. Figure 3 summarizes how the three key aspects of FireFly—flexibility, all-wireless, and use of FSOs— benefit different considerations of DC network design: (1) Flexibility ensures high-performance with lower cost and enables energy reduction []; (2) A wireless fabric eliminates concerns about cabling complexity and interference with cooling [**?**]; and (3) Using FSOs eliminate performance concerns for a wireless network that might arise from range and interference constraints.

While FireFly is not the first architecture to explore a *flexible* DC architecture, we highlight two key aspects of that make it significantly different from prior work on optical or wireless augmentation. First, we are eliminating the need for a wired core and core switches. Second, the use of FSO-based inter-rack links provides a better alternative to existing RF wireless or optical solutions.

2.4 Preliminary work: Cost vs. performance tradeoff

In a preliminary position paper [**?**], we analyzed the cost-performance tradeoff of candidate DC architectures for a 512 rack DC with 48 machines/rack, based on publicly available cost projections. Specifically, we consider full-bisection bandwidth networks such as FatTree [**?**] and Jellyfish [**?**] and the 3D-Beamforming architecture using 60Ghz wireless and ceiling mirrors [**?**].[4] FatTree/Jellyfish-Y Gbps refers to a network with bisection bandwidth of Y Gbps, with Y=1,2, and 10. Here, FireFly (X) refers to a hypothetical realization of the FireFly vision with X FSO devices on each ToR switch.

Architecture	Cost ($)	Per-server throughput
FireFly (16)	18.1M	1.7 Gbps
3D Beamforming []	17.1M	1.1 Gbps
Jellyfish [] 1Gbps	13M	1 Gbps
Jellyfish 2 Gbps	26M	2 Gbps
FireFly (48)	37.8M	8.5 Gbps
Fattree/Jellyfish 10Gbps	57M	10 Gbps

Table 1: Cost and performance of FireFly vs. state-of-art proposals for a 512 rack, 48 machines/rack DC

We assume that a 64-port 10Gb ToR switch costs $27K: $11K for the bare switch [**?**], and $16K for 64 10Gbps optical transceivers at $250 each [**?**]. We assume that a 48-port 1Gb switch costs $5000 [**?**], and each 60GHz radio costs $1000. We assume a hypothetical steerable FSO device can be constructed with a total cost of roughly $750 (see Section 3). We conservatively ignore cabling costs for the wired core for FatTree, Jellyfish, and 3D-Beamforming. Given the above assumptions, Table 1 summarizes the cost and the throughput that different DC architectures offer based on our simulations. The result shows that FireFly offers good cost-performance tradeoffs w.r.t. state-of-art solutions; e.g., close to 9 Gbps bisection bandwidth at much less cost compared to a Fat-tree, and 90% of the performance for 2 Gbps fat-tree at 70% of the cost.

With this context, we discuss the three broad research thrusts we need to address to turn the benefits (Figure 2,3) into reality: (1) *feasibility of FSOs for FireFly (Section 3)*, (2) *foundations of flexible topology design (Section 4)*, and (3) *effective datacenter management (Section 5).*

[4]We don't consider the all-wireless architecture of [**?**] because it has worse cost-performance tradeoffs; e.g., for 24k machines it costs $\approx 50M$ but only achieves 300-400 Mbps per-server [**?**].

3 Designing FSO Devices for Flexible Inter-Rack Networking

In this section, we begin by specifying the design requirements for FSO devices in FireFly, and then describe our design roadmap for meeting these requirements.

3.1 Overview and Requirements

The design of FSO devices in FireFly must simultaneously meet the following requirements:

- *Size, cost-effectiveness, and power:* A single FSO device assembly (i.e., including steering and alignment machinery) should have a footprint of $\approx 3" \times 8"$, so that a few tens of them can be placed on the ToR. Each device shouldn't cost much more than $1000, for our overall design to be cost-effective [**?**]. Finally, their power consumption should be modest.
- *High (10-100Gbps) data rate over long distances (50-100m), with a "misalignment-tolerance" of about 4mm:* DC traffic rates are growing [**?**], and inter-rack distances in large DCs can easily be of the order of 50-100m [**?**]. Thus, FSO links must be capable to providing high throughput over such distances. Finally, we would like the links to be able to tolerate minor misalignments (between TX and RX) due to rack vibrations, airflow shifts, etc.
- *Fast and precise alignment and steering:* For efficient communication, the transmit/receive devices must be precisely aligned. Thus, we need mechanisms for robust re-aligment in face of environmental effects such as vibrations or changes in airflow. Furthermore, to provide fine-grained and efficient reconfigurability, we need a mechanism to precisely steer the laser beam (in order of tens of msecs [**?**]) to connect to another FSO device.

Unfortunately, commercially available FSO transceivers are very bulky (up to 2 cubic feet of volume []), expensive ($6-15K for a single device), and power hungry (tens of watts). This is because they target outdoor long distance (many miles) communications [], and hence, require additional mechanisms to overcome environmental factors [] and long distances. These issues largely disappear in the context of indoor DCs; but, we also have an additional requirement of fast steering.

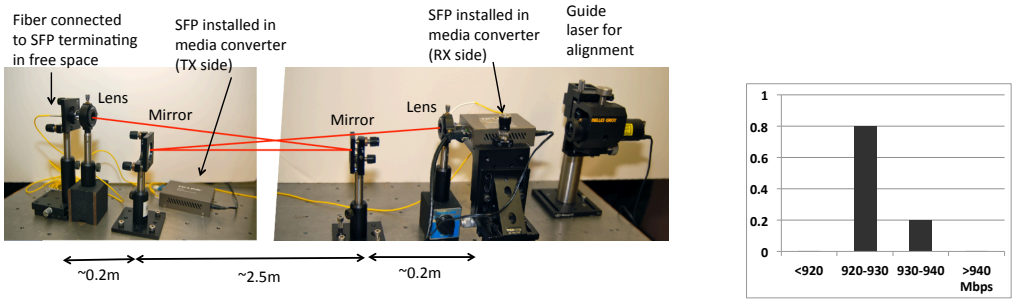
Next, we outline our plan to design an FSO device that meets the above requirements. We first address the first two requirements, and then, address choice and design of alignment/steering mechanisms.

3.2 Cost-Effective, Small-Form Factor, and High-Throughput FSO Devices

Task 1: *We will design FSO devices that satisfy the size, power, cost, and throughput requirements. We will investigate the size and cost vs. performance tradeoff in a DC-specific context.*

Converting Optical SFPs to FSOs. An FSO communication link has three basic components: (i) a modulated laser source (typically infrared) as a transmitter (TX); (ii) a high-speed optical detector/demodulator receiver (RX); and (iii) a robust optical path between TX and RX. To demonstrate feasibility of a small and cost-effective FSO link, we would build an FSO link using the commodity optical SFP (small form-factor pluggable) transceivers []. Optical SFPs are widely used to interface optical fibers with (electrical) packet switches. They are small ($1'' \times 2'' \times 3''$) and low-cost ($100-200). FSOs based on optical SFPs would easily satisfy our size, cost, and datarate requirements, and would not create an additional power burden since SFPs are likely to be used for high-throughput links in DCs anyway. The key difference between optical SFPs and FSOs is that in SFPs, the laser beam is launched directly into the fiber, while in FSOs, the laser beam would be launched into free space. While the idea of converting SFPs to FSOs has been addressed before [**?**], the context of DCs brings in new challenges. The main challenges that arise in converting an SFP into an FSO link are (i) minimizing beam divergence, and (ii) need for precise alignment between the TX and RX. We discuss the first challenge below, and discuss alignment in the next subsection (with the related issue of steering mechanisms).

Minimizing Divergence. An electromagnetic wave diverges in a cone when launched in free space. In our context, this divergence of laser beams causes two problems: (a) the power density on the transverse plane decreases more significantly over distance, (b) the interference "footprint" at the receiver increases significantly. To minimize divergence, we need to design suitable *collimation lens solutions* on the optical path near the TX, to make the laser beam cylindrical of sufficient diameter (since such beams diverge very slowly []). To create a beam of large-enough diameter, we require a lens with a long-enough focal length—which in turn increases our device assembly's length. Note that a large beam diameter also helps tolerate minor



(a) FSO prototype on an optical bench set up

(b) Throughput stability

Figure 4: (a) Experimental prototype showing FSO communication using SFP over 7.5m. Note use of mirrors to achieve a long beam path on a standard size optical bench. (The beam is hand drawn.) (b) Distribution of per-second TCP throughputs (in Mbps) over a continuous 30 hour period over 7.5m.

misalignments, but it also reduces the power density on the traverse plane. Thus, there is a need for careful balance in the design, i.e., to create a beam of appropriate diameter such that it satisfies divergence, misalignment tolerance, and power density requirements under the constraint of total assembly size. Our initial calculations (based on our preliminary prototype, described below) show that achieving this balance in our design is indeed possible, since SFPs are used for very long distance (80-100km fibers) communication and hence their receivers can tolerate enough loss of power.

Preliminary Prototype of FSO Links (1Gpbs over 7.5m, with a misalignment-tolerance of 0.7mm). We have developed a proof-of-concept prototype to demonstrate feasibility of designing FSOs from optical SFPs. See Figure 4. The prototype uses a pair of 1Gbps SFPs using 1310nm lasers. We launch the beam from a single-mode optical fiber connected to the TX SFP on one end with the other end terminating in free space. Due to the narrow fiber diameter ($8 - 10\mu$m), the initial beam divergence is very large. We collimate the beam to a roughly 4mm diameter cylinder using an achromatic doublet lens, with the fiber tip positioned precisely (using optical bench and translating mounts) at the focal point of the lens. The collimated beam propagates to a distance of 7.5m (after reflection through multiple mirrors) and is refocussed using an identical lens on the RX detector.[5] We connect the SFPs to laptops via standard media converters [] and run TCP throughput experiments for 30 hours to test link stability. Figure 4 demonstrates very stable link performance comparable to the wired case. We also analyzed the tolerance of our prototype to misalignment between the TX-RX, and observed that the throughput is stable up to a transverse shift of $\pm 0.7mm$.

3.3 Fast and Precise Steering Mechanisms

Task 2: *Develop fast, precise, and cost-effective steering techniques to achieve reconfiguration in the inter-rack network.*

A wide spectrum of candidate solutions exist for laser beam steering including phased array techniques []. But most of these are not commodity and some are subjects of active research. Feasibility and cost-effectiveness for adapting these solutions for DCs are unknown. For feasibility reasons, we will investigate the following two promising candidate technologies.

Switchable Mirrors. Switchable mirrors (SMs) are made from a special liquid crystal material which can be electrically controlled to rapidly switch between reflection (mirror) and transparent (glass) states at millisecond timescales [**?**]. These are used in various forms of visual aids in niche markets e.g., rear-view video mirrors. Figure 5(a) shows how we can use SMs for steering beams. Each FSO device will be equipped with multiple SMs, with each SM pre-aligned (part of offline pre-configuration) to target a point on the ceiling mirror and thus, a receiving FSO. The desired link is established by switching one of the SMs in the mirror state and the other SMs in the transparent state. This is done at the TX as well as the RX, but only shown at TX in the figure for clarity. A small-size SM is expected to have minimal cost ($<$ \$5 [**?**]), when manufactured at scale.

[5]Since the SFP used here uses two separate optical paths (for duplex operation), the return link is closed using a regular fiber.

Managing Minor Misalignments. SMs are likely to require additional mechanism to tolerate/correct minor misalignments due to vibrations and other factors. Minimal misalignments can be tolerated by using inexpensive vibration isolation, such as rubberized mounts and vibration-absorption construction materials. In addition, we can use micro-positioning systems to manage minor misalignments. Widely available piezoelectric positioning systems would be adequate for our purposes, but they are expensive. Hence, we would design low-cost micro-positioning mechanisms based on bimetallic strip or other material with a high thermal expansion coefficient. By changing the temperature of the material slightly and precisely (e.g., by using an electrical resistor with a varying current), we can micro-adjust the position of the associated device. We will also optimize the mechanical linkage and interface to maximize the range of movement. Feedback needed for the above misalignment correction can be obtained from the available DOM (digital opitcal monitoring) support on optical SFPs.

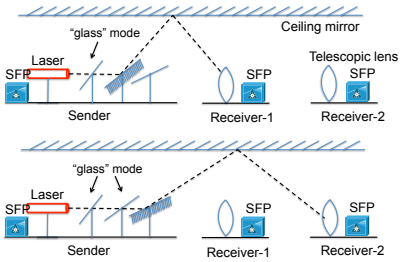


Figure 5: Using Switchable Mirrors with FSOs. In the top-half the second SM is in mirror mode, redirecting the beam to Receiver 1, while the bottom half has SM3 in mirror mode and thus redirecting to Receiver 2.

Preliminary Study. We evaluated the viability of switchable mirrors using a 12″ x 15″ switchable mirror (SM) from Kentoptronics. The switching latency of the SM was found to be around 250 msec. Since the switching latency is proportional to the SMs surface area [**?**], we estimate a < 5 msec latency for a small (1″ x 1″) SM we propose to use.

Galvo Mirrors. Galvo mirrors (GMs) [] are used in various laser scanning applications. GM essentially is capable of steering a (fixed) incident beam into a desired rectangular cone, under computer control; this is achieved using two mirrors mounted at right angles. See Figure 5(b). In our context, equipping an FSO device with a single GM enables us to target any receiver within the pre-configured rectangular cone (chosen offline).

Commercially available GMs [] can provide a total rectangular cone angle of $40°$, with steering latencies of around 100 μsecs. They have a pointing accuracy/precision of within 15 μrad [] which yields a positioning error of up to 1.5mm for up to 100m links; this may be acceptable in our context, since we plan to design links with a misalignment-tolerance of up to 4mm. Commodity GMs are quite large due to the associated machinary. However, only the optical components need to remain on the rack, while the motor and associated electronics can be hidden underneath the rack (using custom-designed extension arms to hold the mirrors; however, additional stability and precision issues must be addressed). We note that emerging MEMS-based galvos [] are also extremely promising for the proposed concepts, and would be explored in our research.

Custom Building a Cost-Effective GM. Commodity GMs used for laser-scanning applications are expensive ($2000 []). However, we see no reasons why the cost cannot be reduced dramatically for our specific context. The core components of a GM include only motor assemblies, mirrors, and some basic positioning control capability. By using commodity hardware such as DC servo motors used in hobby remote control aircraft and low-cost microcontrollers such as the Arduino, we expect that extremely high-precision control can be obtained easily. Based on the above, we will design and build low-cost GM prototypes for our purpose. We have a target price point of $30-50, when fabricated in quantity. Note that the success of our research project would provide significant motivation for additional engineering, cost-reduction, and commoditization of GM hardware which will bring the cost dramatically lower.

SMs vs. GMs. From a mechanical perspective, note that SMs are physically mostly-static (after being pre-aligned), while GMs have mirrors moving in real-time. From a topology design perspective: a GM can reach *any* receiver within the prescribed cone (of limited angle), while use of k' SMs with an FSO provides k' *arbitrary* (but fixed) target receivers. In our research, we will systematically investigate the impact of the above tradeoffs, and use a hybrid architecture consisting of both steering mechanisms.

Pre-Configuration. As mentioned above, each SM needs to be *pre-aligned* to a specific receiver, and each GM needs to be *pre-oriented* to a desired rectangular cone. This is done during the *pre-configuration* phase of FireFly; it is done offline (and very infrequently, e.g., monthly) using an external machinery, so speed and cost is not of the essence. We skip discussion of the required machinery and mechanisms, since it

falls outside the scope of our project. However, we do discuss design of pre-configured topology in the following section.

4 FireFly Network Design

The FireFly hardware, discussed in the previous section, impose physical and geometric constraints on the network design. For instance, the size of the FSO device assembly limits the number of FSOs that can be placed on the rack and the cost/range of steering mechanisms may also come into play. Our goal then is to design the most cost-effective and efficient flexible network design that can work within these constraints.

In our context, there are essentially two stages of network design done at different timescales of operation. First, based on the overall budget, we need to choose the network parameters (e.g., number of machines and FSOs per rack) and "pre-configure" the FSO assembly (i.e., orient/align the GMs/SMs in the system). This needs to be done at coarse time granularity (e.g., monthly), because of the time incurred in changing network parameters or pre-configuration. Second, given this pre-configured network, we need to choose a *runtime* topology by activating a subset of links, at finer timescales (i.e., few millseconds) based on the prevailing traffic load. Thus, we envision the design workflow in Figure 6. We defer the runtime operation (called *reconfiguration*) to Section 5.1. In this section, we focus on the first problem of coarser-timescale network design. We address this problem in two stages. First, we focus on the abstract problem of pre-configuring the FSO assembly, given the network parameters; this abstraction is likely to give us insights into understanding the theoretical implications of topology flexibility. Then, in Section [**?**], we use these insights and address the overall (budget-based) network design problem.

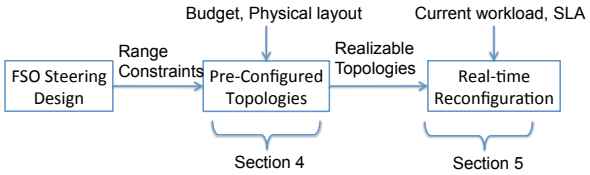


Figure 6: Interaction between the design constraints induced by FSO steering choices, the selection of preconfigured topologies, and the real-time topology selection

4.1 Pre-Configured Flexible Topology Design (PCFT)

To gain insights into the theoretical foundations of the problem, we abstract away the details of the SM or GM or the cost budget and focus on the following problem: Given the number of racks n, number of FSOs m on each rack, and the maximum number of *candidate links* k per FSO that each FSO can be steered to, the *pre-configured flexible topology (PCFT) design problem* is to determine the set of links connecting pairs of FSOs, so as to optimize the "dynamic bisection bandwidth" of the network (defined below).

Terminology. A *PCFT* essentially is a k-regular graph over the m FSOs, where each edge is called a *candidate link* and represents an achievable communication link. However, at any point during runtime, only one candidate link per FSO can be *active*; thus, the set of active links form a matching over the FSOs. Given a PCFT, any matching over the FSOs is called a *realizable topology* of the given PCFT. See Figure 7.

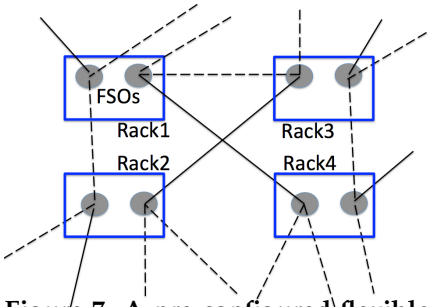


Figure 7: A pre-configured flexible topology (PCFT) with candidate links (solid and dashed). Number of FSOs per rack (m) is 2, and the number of candidate links per FSO (k) is 3. The set of solid (active) represents one possible realizable topology.

Task 3: *We will investigate the theoretical foundations of flexible topology design that maximizes dynamic bisection bandwidth in conjunction with other measures of network goodness (e.g., diameter).*

Rethinking Metrics for Flexible Topologies: The traditional bisection bandwidth metric [] reflects a *static* perspective of the topology. However, in our context, we should instead consider the notion of *dynamic* bisection bandwidth (DBW) since different realizable topologies can be used for different communication requirements. Formally, the dynamic bisection bandwidth of a given pre-configured topology Π can be defined as follows. Let T be the set of realizable topologies of a given pre-configured topology Π, P be the set of partitions of the given network into two equi-sized sets of machines, and $BW(t,p)$ be the bandwidth of a (realizable or static) topology t for the partition p (i.e., the cut-size in t corresponding to p). Then, the traditional notion of bisection bandwidth for a (static) topology t is given by $\min_{p\in P} \mathrm{BW}(t,p)$., while the *dynamic bisection bandwidth* DBW(Π) for the pre-configured topology Π is defined as:

$$\min_{p\in P} \max_{t\in T} \mathrm{BW}(t,p).$$

In addition to just maximizing DBW, we also consider the objective of maximizing DBW under the constraint of bounded network diameter. Network diameter is a reasonable way to bound worst-case latency, which has also been considered as an objective in designing datacenter topologies [**?**]. We also also use an appropriate notion of *dynamic diameter* (as for DBW above) instead of the traditional notion of network diameter.

Related Known Problems: In general, the PCFT problem is in the class of the *network design problem (NDP)* [**?**] (and more specifically, *degree-constrained subgraph* problem []). However, our PCFT problem is very different from the prior-addressed NDP problems, due to our novel objective function. For the special case of $k = 1$, the PCFT problem essentially boils down to constructing an m-regular graph over n nodes with maximum (static) bisection bandwidth. The closest known problems to this special case are: (a) The NP-hard problem of computing the bisection bandwidth of a given regular graph [**?**,**?**], and (b) Determining an upper-bound on the bisection bandwidth of m-regular graphs of size n, for a given m and n; this problem has been studied extensively, with (non-tight) upper bounds determined only for very small values (upto 4) of m [**?**]. Note that this upper-bound problem can be reduced to the special case ($k = 1$) of the PCFT problem. *The above observations suggest that the PCFT problem is very likely intractable, even for* $k = 1$.

Proposed Approaches: We will pursue three different approaches for the PCFT problem. The first approach builds upon designs with high static bisection bandwidth, the second approach builds upon solutions to special cases of the dual-problem, while the third is a heuristic-based approach.

- *Static to Dynamic Conversion.* One reasonable approach to solve the PCFT problem would be to start with constructing a mk-regular graph of n nodes with a high (static) bisection bandwidth, and then group the mk candidate links at each node into m sets of k links each so as to maximize the dynamic bisection bandwidth. For the first step, there are only a few results on explicit construction of general graph classes with high bisection bandwidth [**?**]. In particular, d-regular Ramanujan Graphs [**?**] of are known to have a bisection width of at least $((d/2-\sqrt{(d-1)})n/2$, but their construction is mostly algebraic. Other graph classes of interest are Cage graphs [**?**], which have a large girth (length of smallest cycle). In our specific context of small values (a few hundreds) of km and n, we can investigate bisection bandwidth of certain classes of regular graphs, and pick ones that suggest a high bisection bandwidth with low diameter; in particular, due to symmetry of nodes/racks, we can also restrict ourselves to "symmetric" regular graph such as distance-transitive graphs. For the second step of grouping candidate links, we will employ certain heuristics. E.g., we can number the n nodes from 1 to n and group the mk links into m sets based on the ranges of node numbers they connect to. Such a heuristic will guarantee a "uniform" division of links into sets across the nodes.
- *Dual-based Approach.* If each rack contains l machines and the links (between a machine and the ToR switch, or a pair of FSOs) have a unit bandwidth, then the optimal desired bisection bandwidth is $nl/2$. We note that, for uniform link bandwidths, the inter-rack and inter-machine bisection bandwidths are same. Now, let us consider what values of k and m can enable this DBW value of $nl/2$. We consider two extremes: (a) If $k = 1$, then it can be shown that $m = \min(n/2 + l, 7l)$ suffices[6] (but not necessarily optimal). For large values of n and l, it is known [**?**] that $m = 2l$ would almost always work. (b) If k can be an arbitrarily high, then the optimal value of m required is l (for $k = n/2$); here, for $m = l$, the k required (=$n/2$) is also optimal. The above resuls hold for any DBW value that is an integral multiple of n. The above near-optimal solutions for certain special cases of the "dual" problems (i.e., given a desired DBW value, minimize m or k for a given n value) gives us some insights into solving the PCFT and its dual problems. In particular, if we can solve the above dual problem of minimizing m, given k, n, and a desired DBW value, for arbitrary k values and integral of n values of DBW, then it is easy to derive an approximation algorithm for the PCFT problem that has only an *additive* approximation-factor of n.
- *Simulated Annealing (SA-PCFT).* Simulated Annealing (SA) heuristics [] have been used with great success for optimization problems. In our context, since the PCFT design is done offline, we can afford the convergence time incurred by an SA approach. To design an SA approach for our PCFT problem, we need three key components: First, we need good "seed" (starting) solutions; here, we could use one of our earlier approaches, or as in [**?**], use graphs with a "large spectral gap" [**?**, **?**, **?**] which are known to have desirable properties (e.g., low diameter [**?**]). Second, we need ways to generate "neighboring" solutions; for this, we can use simple transformations that transform a regular graph to another. E.g., the

[6] By using $7l$ ports on a ToR switch, we can simulate a Fat tree architecture.

transformation that changes the edges $(a, b), (c, d)$ to $(a, c), (b, d)$ can be used iteratively to construct any regular graph from another. Lastly, we need an efficient heuristic for computing DBW (PCFT's objective function) of a given graph; for this, we will investigate generalization of the following approaches: (i) Well-known efficient heuristics, viz., SA [] and Kernighan-Lin [] for computing the bisection bandwidth, and (ii) a recent result [**?**] that uses Valiant (or, two-state) load balancing technique [**?**] to compute a *lower bound* on the bisection bandwidth.

The above SA-PCFT approach can also be generalize to maximize an appropriately defined **traffic-aware DBW objective**, based on coarse statistics available on inter-rack traffic. Note that since pre-configuration can only be done on an infrequent basis (e.g., weekly), so we are only interested in coarse traffic knowledge. One simple form of inter-rack traffic statisfics could be in the form of a weights between every pair the racks, and the DBW definition can be appropriately tailored as in [**?**] for multi-commodity min-cut. In our research, we will also consider incoporating more sophisticated traffic models [].

4.2 Budget-Based Network Design (BBND)

Formally, given the total number of machines to interconnect, physical constraints, overall budget, and pricing of relevant hardware devices, the *BBND* problem is to determine the following such that the dynamic bisection bandwith is maximized: (a) Number of machines (l) per rack and thus, the number of racks (n), (b) number of FSOs (m) per rack and thus, the number of ports on the ToR switch, (b) on each rack, the number of FSOs ($g \leq m$) that are each equipped with a GM and the number of SMs (k') on each of the remaining $(m - g)$ FSOs, and (c) the orientation/alignment of each of the GMs and SMs in the system.

Task 4: *We will design efficient algorithms for the Budget-Based Network Design (BBND) Problem with the objective of maximizing the dynamic bisection bandwidth.*

Proposed Approach: Based on our insights from the previous section, we will use the following approaches to address the BBND problem:

- *PCFT-Based Algorithm.* We can use any of the PCFT algorithms as a subroutine to solve the BBND problem as follows. First, we convert the given budget and physical constraints, and the pricing information into a constraint equation over n (number of racks), m (number of FSOs per rack), and k (the number of candidate links per FSO). To constrain k, note that $km = cg + (m - g)k'$, where c is the number of candidate links a GM can be steered to use and can be assume to be a constant. Second, we solve the PCFT problem for various n, m and k that satisfy the above budget constraint for a given g ($\leq m$). Then, we convert each PCFT solution to a design realizable by g GMs and $(m - g)$ sets of k' SMs each, on each rack, and estimate its DBW using one of the approaches described earlier. Lastly, we explore the space of n, m, k, g efficiently using standard search techniques, to compute an efficient network design for the given budget and pricing.
- *Extending SA-PCFT.* We can modify our Simulated Annealing approach for the PCFT problem to solve the BBND problem, by appropriately modifying the transformation operator to generate neighbors of a particular network design. Here, the neighbors of a design may include designs with slightly different values of parameters n, m, k, g, and/or candidate links, under the budget constraints.

5 FireFly Network Management

In this section, we focus on the design of a *datacenter management layer* that uses building blocks from previous sections, to implement a feasible reconfigurable datacenter network. We start with describing the high-level roles of the different components of the management layer. See Figure 8.

- **Monitoring Engine (ME):** ME provides network status information (e.g., links status, traffic patterns, or views of "elephant" flows [**?**,**?**,**?**]) to the management layer.
- **Optimization Engine (OE):** Given the offered traffic workload, a pre-configured flexible topology (PCFT), and the current network state (e.g., link status), the optimization engine devises an efficient *reconfiguration and traffic engineering strategy* so as to optimize desired performance goals.
- **Data Plane Translation Engine (DPE):** DPE translates the OE output into an efficient data plane strategy.
- **Application APIs:** APIs enable the users/tenants to inform application details (e.g., expected traffic patterns, single/multi-path TCP) to the optimization and data plane modules, to best leverage the benefits of FireFly.

Plan. For the ME, we will leverage past work on scalable traffic matrix and elephant flow detection [**?**,**?**,**?**]. Similarly, we will extend prior work on APIs for applications to expose traffic patterns [**?**,**?**]. We address OE and DTE in the following subsections. In the last subsection, we address design of a wireless out-of-band control network for configuration dissemination and data collection.

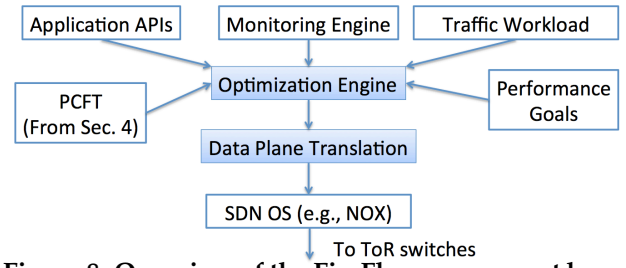


Figure 8: Overview of the FireFly management layer

5.1 Reconfiguration and Traffic Engineering

Task 5: *We will develop fast and efficient algorithms for the joint optimization problem of reconfiguration and traffic engineering.*

Joint Reconfiguration and Traffic Engineering (JRTE) Problem. Given the traffic load, the *JRTE* problem is to: (a) *Reconfigure*, i.e., select a realizable-topology from a given pre-configured flexible topology, and (b) *Traffic Engineer (TE)*, i.e,. route the given inter-rack flows over the realizable-topology selected in (a), to optimize a desired objective. The objective functions of interests could be to minimize link congestion or maximize total flow in conjunction with fairness, latency bound, and/or tenant provided SLAs.

The reconfiguration subproblem of JRTE falls in the general class of *degree-constrainted subgraph* problems [], and is akin to the topology control problem [] addressed for wireless networks. However, the constraints and TE-based objectives of our subproblem makes it very different prior-addressed problems. Note that the TE subproblem of JRTE is essentially solving the NP-hard unsplittable multi-commodity flow problem [**?**,**?**] over the selected topology—thus, the JRTE problem is trivially NP-hard.

Proposed Approaches. Note that the JRTE algorithm is run in real-time, and hence, should not take more than a few milliseconds, for it to be of any real benefit. Thus, the challenge is to design very fast, scalable, and efficient JRTE algorithms.

- *Matching Techniques.* Recall from Section 4.1 that a realizable-topology is a matching over the FSOs. Thus, a reasonable approach to the reconfiguration subproblem could be to select the maximum-weighted matching (solvable in polynomial time) between FSOs, where the link (i, j) is weighted by the inter-rack traffic demand between the correspondin racks. Such a topology maximizes the total inter-rack demand that can be served using one hop. In general, we would like to pick a matching that yields the minimum weighted-average inter-rack distance (where the distances are weighted by the traffic demands); however, standard matching techniques do not apply to this unsubmodular [] objective function. One approach would be to use a greedy strategy that iteratively adds augmenting paths [] based on a suitable benefit-function for augmenting paths. After reconfiguration as above, the TE part can be done using known techniques [] over the selected matching.
- *LP Relaxation Techniques.* Another promising approach is to formulate the JRTE problem as an ILP (using splittable flow-like constraints, and binary variables for link selection) with the objective of minimizing link congestion or maximum "fair" flow [**?**], and solve the relaxed LP. We can then convert the LP

solution to an ILP solution by an appropriate rounding technique while ensuring that the matching-constraint is still satisfied (unsatisfied flow-constraints only result in a sub-optimal but valid solution). To handle the unsplittable aspect of flows, we can: (i) use the classic path-striping technique [**?**], and/or (ii) allow flows to be split, but minimize the number of splits by considering the *aggregate* flow per destination, as suggested in the recent work [**?**]. An alternate approach in a similar vein as above is: First solve the TE problem over the entire PCFT graph, and use the flow-values to select a "good" matching. It would be interesting to compare the performance of above approaches over real traffic traces.

Further Directions. In addition to the above, we are also interested in designing algorithms based on limited traffic information, and incremental or localized approaches.

- *Strategies with Limited Traffic Information.* Our previous discussion implicity assumes availability of traffic demands for the next epoch. However, in reality, traffic predictability may be limited. In the worst case, we may only be able to distinguish between "elephant" (large) and "mice" (small) flows, based on their initial size. In such restricted settings, a reasonable approach would be to solve the JRTE problem only in response to the arriving elephant flows, while relying solely on TE for the mice flows. In our preliminary work [**?**], we employed a simple strategy based on the above idea, and achieved near-optimal performance over randomly generated traffic traces. More information about traffic patterns such as spatial and temporal distribution of elephant flows (or flow sizes in general) would require challenging generalizations of the above approach. Such elephant flow based approaches should be quite effective, since the structure of real workloads suggests that elephant flows are typically long-lived [] and a small number of them carry the most bytes []. An additional objective, in the context of above approach, should be to favor JRTE solutions that cause minimal disruption to existing traffic flows. This could be achieved by developing techniques to compute the "impact" of a solution on the existing traffic; this may be intractable, but computing upper and/or lower bounds of the impact may be sufficient for our purposes.
- *Incremental or Localized Strategies.* One way to develop fast JRTE algorithms is to determine the solution in an incremental manner (e.g., by constraining the number of links that are activated or deactivated). Morever, we can also restrict ourselves to only "localized" distributed strategies. To design incremental JRTE algorithms, we can use augmenting-path techniques to incrementally improve the matching [], with additional constraints and/or a benefit function (as suggested before). To design localized strategies, we would explore extend the recent work [**?**] which develops a simple switch-local routing algorithm. In our context, we will locally (e.g., at each switch) change active links as well as the forwarding hops, in response to sufficient change in arriving flows.

5.2 Data Plane Strategies

Task 6: *We will design and implement efficient data-plane implementations to guarantee reachability and consistency properties, in presence of reconfiguration and link dynamics.*

In translating the JRTE solution into a consistent and efficient data plane forwarding strategy, we build upon recent advances in software-defined networking (SDN), as it provides cleaner management abstractions and open interfaces (e.g., via APIs such as OpenFlow []). However, FireFly introduces unique challenges. In particular, in face of a dynamically changing network due to reconfigurations, we need to ensure that: (a) packets do not use deactivated links (i.e., black holes [] are avoided), (b) the network remains connected at all times, and (c) the packet latency remains bounded. Finally, we also need a data plane strategy to: (d) handle transient misalignment of FSO links. We note that the recent related works either assume a *static* network [**?**, **?**] or focus on a single reconfiguration [**?**], and hence, are not directly applicable to our context.

(a) and (b). Avoiding Black Holes, and Guaranteeing Connectivity. Packets are routed in the network on the basis of forwarding tables, which essentially specify, at each node, the next hop/link to use for each destination. In a dynamic network, forwarding tables will also be changing constantly. Note that activation/deactivation of a link may take a few tens of msecs, and that we cannot directly update the tables across all the network switches in one atomic action. In face of these challenges, we need to ensure that through every possible intermediate state of the switches' tables and links, the forwarding tables always refer to only active links. We can ensure this by a careful ordering of steps as suggested in our preliminary

work [**?**]. In particular, (i) we reflect removal of links in the forwarding tables, before actually deactivating the links, and (ii) reflect addition of links in the tables only after the link activation is complete. The above solution works irrespective of the order in which the forwarding tables are updated across the network, and even in the presence of multiple concurrent reconfigurations.

In addition to above, we also need to ensure that the network remains connected at all times. There are two possible solutions to this: (i) maintain a static "backbone" subnetwork that ensures connectivity, or (ii) reject reconfigurations that disconnect the network. The first approach reduces the degree of flexibility in network design and could result in poor performance. The second approach requires a careful implementation if there are multiple *concurrent* reconfigurations. Concurrent reconfigurations can be handled in three ways: (i) one at a time, (ii) in batches (i.e., queue and combine them into a single reconfiguration); and (iii) execute each reconfiguration individually but *concurrently*. The first two options introduce unnecessary reconfiguration delays, while the third option requires a careful implementation to ensure correctness—we need to keep a single consistent view of the network topology graph and allow only *atomic* access to it (when checking if a reconfiguration disconnents the network). In our research, we will study and compare the performance of such approaches.

(c) Guaranteeing Bounded Packet Latency. The above strategies still do not guarantee a bounded packet latency. In fact, in general, its *impossible* to guarantee bounded packet latency (e.g., see Figure 9). However, such situations can be avoided if we ensure a minimum time interval between consecutive reconfigurations; this minimum interval is also essential for the centralized controlled and the wireless control channel to be able to handle the resulting load (see Section **??**.) In particular, we can formally prove that a minimum interval of $(x + y)$ time units between reconfigurations suffices to ensure an upper bound of $2x$ units on packet latency, where x is the bound on packet latency given a fixed network and y is the maximum time taken to update (not necessarily atomically) the network forwarding tables. We can expect x and y to be of the order of 1-2 msecs. Note that the above claim is independent of the link activation or deactivation latency (which can be a few tens of msecs).

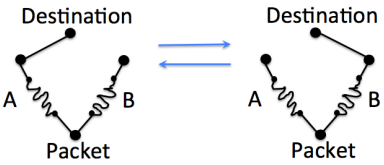


Figure 9: If the network goes back and forth between the above topologies (due to corresponding reconfigurations), then the packet will continue to "swing" between areas A and B—leading to a (perennial) forwarding "loop".

(d) Handling Misalignment of Links. In FireFly, even during a static topology state, links may be temporarily unavailable because of possible misalignment of the FSO links. Such misalignments are fixed in real-time by mechanisms suggested in Section 3 in timescales much smaller than the time needed to update rules [**?**] by an SDN controller. In fact, it may even be counterproductive to report such transient link failures to the controller, as it may cause needless update of forwarding tables. Thus, we need appropriate network layer techniques to recover from such transient link failures. Future SDN roadmaps have provisions for local recovery mechanisms analogous to similar schemes in the MPLS and SONET literature []. We will explore the available alternatives in our research. In the absence of such features, we will investigate design of a local "lightweight" SDN controller on every rack that can quickly react to such misalignments.

5.3 A Wire-free Control Channel

Task 7: *We will design and implement a RF-based architecture to provide a robust, low latency control channel for FireFly.*

Problem Context and Challenges: Existing work in the SDN-style centralized network management literature either implicitly or explicitly assumes the availability of an "out-of-band" control channel that is not managed by the SDN network itself []. This control channel is typically used for the controller-switch protocols—delivering configuration commands and collecting switch statistics. Otherwise, there can be subtle bootstrapping problems with respect to the availability of the control channel itself.

As discussed in the previous section, we will have to engineer a reliability/consistency mechanism even for the regular inter-rack fabric. We can exploit this basic reachability framework as a basis for in-band control. For instance, we can set up static shortest paths between each FSO switch to the controller at the time of pre-configuration and not reconfigure the constituent links. That said, if we are to automate this process, we still have a bootstrapping problem where these switches will need to discover paths to the SDN controller. Furthermore, there is still a concern that the control path may be transiently unavailable

during micro-alignment pauses. While these may not be fundamentally intractable, since the reliability of the control channel is critical to the operation of the entire network we will explore design issues for an out-of-band control channel.

Proposed Approach: A promising candidate for the out-of-band control channel is a separate RF-based wireless network in keeping with the general vision of all-wireless inter-rack fabric. For capacity planning, assume $\approx 10^3$ new flow arrivals per rack with each producing an update of size $\approx 10^2$ bits to the controller and the controller reconfiguring zero or more FSO links in response. Further, assume $\approx 10^3$ reconfiguration commands (of size $\approx 10^2$ bits) from the controller to the FSO subsystem on each rack. This produces control traffic comfortably below 1 Mbps per rack. Back-of-the-envelop estimates using dimensions of data center and reasonable wireless link budgets show that the upcoming 802.11ad standard in the 60 GHz band should be able to support this rate for about 1000 racks using just 1 channel, with the controller connected to an AP and with phased-array antennas on the controller side (likely to be common for future 802.11ad APs). Fixed directional antennas are sufficient on the racks as they talk to only the controller. For larger data centers, multiple channels are needed (supported by multiple NICs) as the aggregate data rate will push beyond the capacity of a single channel. But since at most 3 orthogonal channels are possible in the 60GHz band, for the largest of the data centers ($\approx 10,000$ racks), additional APs ($\approx$ 10, ceiling mounted) distributed across the data center and connected to the controller via fiber links and a switch need to be deployed.[7]

The link layer protocol plays an important role to enable low latency communication. Since the control traffic is somewhat regular, scheduled access – as opposed to more conventional CSMA-based random access – is more meaningful. If designed right, this can guarantee a (small) upper bound of the channel access delay and a completely interference free channel access. Together these properties provide the needed QoS for control traffic. Note that critical control traffic related to micro-alignments also go over this control network. This traffic is expected to be low volume, but should be of the highest priority. The controller schedules access slots for the clients (racks) based the current network state, reconfiguration needs, etc. If multiple APs are needed for larger systems, interference between links is to be pre-determined based on static measurements at the setup time. This allows for conflict-free scheduling of links []. Since there is no mobility either at the end points or in the general RF environment, such measurement-based methods are feasible []. The 802.11ad standard has support for scheduled access, so future data centers will be able to use commodity RF platforms. However, as many of our experiences with commodity systems indicate, appropriate software/tool support may not be available in commodity systems, e.g., driver-level and firmware support to implement special purpose scheduling, measurement support to determine interference, etc. and it will require working with the device manufacturers. Thus, for the purpose of our prototyping, we will custom-build the RF-based control channel on a software radio platform (USRP/GnuRadio) in the 2.4 GHz band[8] and implement the scheduling support for demonstration and evaluation.

6 Possible Extensions to FireFly Architecture

In our project, we would also investigate the following ideas, which impart more flexibility to our design and/or present additional opportunities to improve FireFly's performance.

- *Dynamic Reconfiguration of Link Bandwidths.* In FireFly, the bandwidth of each FSO link is limited by the capacity of the ToR switch port. Facilitating FSO links to have variable bandwidth would add another dimension of flexibility to our design. This can be achieved by having each port of ToR switch associated with a unique wavelength, and using a multiplexer and a WSS (wavelength selective switch) unit between the ToR switch and the FSOs, as in OSA [**?**]. In essence, the multiplexor and WSS allow one or more ToR ports to "feed" into a single FSO link, and hence, facilitating variable-bandwidth FSO links. The WSS can be reconfigured in real-time to determine the allocated ports (and hence, bandwidth) to each FSO link. It would be interesting and challenging to incorporate the above flexibility into our design, and appropriately generalize the techniques from Section 4 and 5.
- *Non-ToR Switches and FSOs.* As described before, our network architecture consists of only ToR switches whose ports are either connected to the rack machines or the FSOs on the rack. Incorporating non-ToR switches (as in most data center architectures []), whose ports are connected to only FSOs, can impart

[7] Wires can be completely eliminated by mesh networking these APs. But since they are very few perhaps the added complexity is not worthwhile.

[8] In a scaled down set up, the 2.4 GHz band is enough as the capacity requirement is much lower.

more flexibility to FireFly's design. The new challenges that need to be addressed in this context are: (a) Physically accomodating the FSOs connected to such non-ToR switches in the data center, (b) PCFT, BBND, and JRTE problems would include determining the inter-connections between these additional switches. Note that the resulting PCFT solutions would now be non-regular graphs.

- *Vertically Steerable FSOs; 45° Mirror Poles.* In our design, we use a ceiling mirror to circumvent physical obstruction for line-of-sight FSO communication. However, in certain contexts such as outdoor scenarios for containerized architectures [**?**] installing a ceiling mirror may not be feasible. In such cases, we need other mechanisms for line-of-sight communications. E.g., we can install FSOs on vertically-steerable poles such that each link operates on a separate horizontal plane. Another possible mechanism could be to have FSOs direct their beams to vertically-steerable small mirrors angled at 45°. To avoid physical obstruction from poles/mirrors in the above approaches, shorter distance links can be operated on a lower horizontal plane than the longer distance links.

7 Prototyping and Evaluation Plan

Task 8: *We will evaluate our approach both at an individual component granularity as well as an end-to-end prototype and testbed demonstration.*

- **Design and protoype compact, cost-effective, steerable FSO devices:** We will prototype a proof-of-concept 10 Gbps SFP-based FSO devices with a small form factor and design optical mechanisms to collimate the laser beam to about 100m. prototype two proposed steering mechanisms: using switchable mirrors and galvo motors. As a starting point, we will decouple these two steps and repurpose our existing commodity/outdoor FSO devices [] to test steering mechanisms.
- **Reliability of steerable FSO in realistic conditions:** Real DCs will have several sources of "disturbances" (e.g., rack vibration, temperature gradients, airflow patterns, etc.) that may cause alignment and performance issues for FSO communication. First, we will create a lab environment that can emulate the effects of different types of disturbances. To estimate the range parameters for these effects, we will engage our industry partners (see letters from Facebook and Microsoft) and add instrumentation sensors to compute clusters at local organizations (e.g., Brookhaven National Lab and CEWIT). Second, we will deploy a small number of FSO links in an actual DC environment (CEWIT cluster in Stony Brook University) and conduct a longitudinal study of the reliability of the links.
  [SD: do we have access to a lab that can emulate "disturbances"??]
- **Performance and benefits under realistic workloads:** We will develop scalable packet- and flow-level simulation platforms extending prior work **[?, ?]** to evaluate the benefits of our topology design (Section 4) and reconfiguration (Section 5) algorithms. We will start with extrapolating from existimg small-scale datasets **[?,?,?,?]** and work with industry supporters (e.g., Facebook and Microsoft) to quantify the benefits at scale.
- **Responsiveness, and correctness of control plane:** We will implement a SDN controller starting with research prototypes **[?]** and port our ideas to open-source platforms such as OpenDayLight [] as the project matures. We will synthesize benchmark suites to "stress-test" the scalability and responsiveness of our controller. We plan to leverage our experiences with emulation platforms such as MiniNet [] and Emulab [] to test the correctness of the proposed recovery and consistent reconfiguration mechanisms in the presence of network dynamics.
- **End-to-end integration and evaluation:** A full-scale DC testbed is outside the scope of the proposal in terms of infrastructure and personnel resources.[9] Within the scope of our budget, we will demonstrate a proof-of-concept testbed of 4 nodes (node represents a rack). Each node will be essentially a NetFPGA card [] on a host computer. Each NetFPGA card has 4 x 10G SFP ports, three of which will connect to a FSO device each with one left for the controller use. We will use OpenFlow switch implementation on the NetFPGA cards [] to represent the ToR switch. Using NetFPGA will enable precise timing and diagnostic information [], link characterization [], as well as aid in high-rate traffic generation [].

  The 4 node setup (along with the 4x3=12 FSO devices) will be deployed on top of the racks in an operational cluster (in CEWIT). realistic environments. The nodes will be moved around on different racks to create various geometric possibilities. This will create various stress cases for studying the stability of the FSO link and steering performance. [SD: will somebody complain that real data centers have real obstructions so such deployment is difficult?]
  [VS: something abt USRP etc?]

8 Broader Impact

Some input from Jon would be helpful too.

Impact on Economy and Environment. With growing interest in Big Data, cloud computing and virtualization, data centers are now common in every sector of the economy. This includes IT industry, government, media, healthcare, financial sector, transportation and the scientific community. The largest of the data centers are known to cost more than a billion USD and are significant power hogs consuming 10s of MW of power []. Overall, recent EPA studies concluded the the total data center electrical power usage is roughly a few percent of the entire electricity consumption in the US and lagging only modestly behind

[9] We plan to develop separate infrastructure proposals to develop at-scale prototypes.

the total household electricity consumption []. We foresee that the Firefly architecture can significantly reduce both cost (by eliminating the need for over-provisioning) and energy consumption (by making the network design energy-proportional and also by improving cooling). This certainly will have perceptible economic impact by making many IT services cost less - both in terms of dollars and carbon footprint - across all sectors in the economy. In addition, success in the proposed project will garner immediate interest in industry for further developing and productizing the proposed FSO-based interconnection. R&D and manufacturing of such interconnections will produce a different form of device industry that will include optical engineers in addition to traditional computer hardware engineers.

Integration of Research and Education. A strength of the project is that it brings together two disparate disciplines, opto-electronics and computer systems. Due to this unique nature, the participating ME students will learn basics of data center networking and the CS students will be exposed to laser communications. From preliminary studies leading to this proposal – that involved several grad students from both CS and ME – it is our experience that the students particularly enjoyed learning a completely new technology and understanding the methods and practices of a different discipline.

The project will directly contribute to relevant graduate courses in both mechanical engineering (ME) and computer science (CS) that the PIs teach regularly. These include advanced courses on "wireless networking" (Gupta/Das), "software-defined networking" (Sekar) and core graduate courses in "networking" (Sekar/Das). The PIs regularly scope out suitable topics from their existing research projects to assign term projects to these graduate level classes. Over the past years, such projects motivated students better due to their direct contribution to a bigger effort. This improved participation as well as the final results, often students continuing working on these topics past the course term, writing papers or developing various artifacts of longer term value.

Engaging High School and Undergraduate Students. Long Island have some of the best public schools in the country and we are keen on tapping into this high school talent. SBU has a Simons Summer Research Program[10] that provides a mechanism to recruit talented high school students. PIs Gupta and Das have mentored students in the past in this program whenever they had projects that high schooler could relate to easily (e.g., on RFID tracking and robot navigation). The proposed project brings in 'hands-on' components such as FSO communications, use of steerable lasers to implement network switching that we believe would be of interest to the participating high schoolers in the Simons program. We hope to be able to recruit 1-2 such students each summer.

The PIs have strong history of involving undergraduates in research and supporting them via REU supplements. This approach motivated a few of them enough in the past that they joined in the graduate program. The PIs will continue this involvement for the proposed project.

Involving Under-Represented Groups. The PIs have history of mentoring students from the traditionally under-represented groups in STEM disciplines. Gupta and Das has graduated a woman PhD student each in the recent past and Das currently advises one woman PhD student. Earlier, Das also graduated a PhD student, Robert Castañeda, from the Hispanic community and Castañeda is currently a faculty member in a minority serving college in his native San Antonio. Das also served in several related NSF organized outreach efforts, e.g., mentoring faculty from minority-serving institutions for writing competitive NSF grants. The PI will continue similar efforts in connection with the proposed project.

9 Results From Prior NSF Support

Samir R. Das and **Himanshu Gupta** are PI/Co-PIs on the following recently concluded/ongoing NSF awards: i) 'A Market-Driven Approach to Dynamic Spectrum Sharing' (2008-13, $406,000), and ii) 'Understanding Traffic Dynamics in Cellular Data Networks and Applications to Resource Management,' (2011-14, $320,425). These projects focus on developing market-driven algorithms and systems for dynamic spectrum access systems (first) and understanding spatio-temporal traffic dynamics in cellular data networks via analysis of network traces and using them for spectrum/energy management applications (second). Over 15 papers were co-authored by the PIs related to these awards and 6 PhD students received direct support. The PIs gave several public lectures based on the results.

Jon Longtin, Co-PI, CBET-1048744, "NSF/DOE Thermoelectrics Partnership: Integrated Design and Manufacturing of Cost-Effective and Industrial-Scalable TEG for Vehicle Applications," $404,226, 10/1/10-

[10] http://www.stonybrook.edu/simons/

09/30/13. This work developed thermoelectric materials and systems for automotive applications using thermal spray technology. Mg_2Si and filled Skudderudites were deposited directly onto exhaust system components to convert exhaust waste heat to electricity. Key tasks on the project included 1) materials development and characterization, 2) short-pulse laser micromachining deposited thermoelectric materials, 3) techniques for fabrication of multi-layer thermoelectric generators, and 4) device characterization and testing. 6 conference, 4 journal publications, 4 Masters thesis and 1 Ph.D. thesis have resulted from the project.

Vyas Sekar is a PI on two recently awarded NSF grants "Enabling Flexible Middlebox Processing in the Cloud" and "Rethinking Security in the Era of Cloud Computing" starting in Sep 2013. The research proposed therein focuses largely on "middlebox" functionality such as IDS, firewall, and proxies and does not focus on the datacenter topology and routing aspects. These projects have just commenced and there are no outputs at this time. As such the proposed research in these projects does not overlap with the management layer/SDN approaches proposed here.