Computer Communications 170 (2021) 95–108

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Fly-path: Traffic-based multi-hop routing approach for hybrid wireless data centers

Cem Mergenci *, Ibrahim Korpeoglu

Department of Computer Engineering, Bilkent University, Ankara, Turkey

ARTICLE INFO

Keywords:
802.11ad
60 GHz
Wireless data center networking
Data center traffic
Routing

ABSTRACT

High data transfer rates achieved by 802.11ad at 60 GHz ISM band enables use of wireless communication in data centers. In this paper, we investigate the possibility of offloading traffic from wired to wireless network in hybrid data centers. By understanding the capabilities of the wireless network, we can design the hybrid data center network accordingly, to achieve better construction and operating efficiency. First, we propose a system model in which each top-of-the-rack switch is equipped with two radios, so that three non-overlapping channels of 802.11ad that are available worldwide can be assigned in an interference-free manner to any configuration of wireless links. Then, we propose multi-hop routing algorithms that assign traffic to wireless infrastructure. These algorithms consist of two families. SP family of algorithms route traffic only over shortest-paths between source and destination pairs. LP algorithms relax this restriction and assign traffic to longer paths when necessary. In order to evaluate the performance of our routing algorithms, we also propose a random data center traffic generation method, based on an analysis of a real-world data center traffic pattern. We evaluate the performance of our allocation methods in terms of different metrics for various network sizes. Results show that our methods can offload significant amount of traffic from wired to wireless network, can achieve quite high throughput, and can utilize wireless links very well.

1. Introduction

Unlicensed 60 GHz ISM band offers high-bandwidth line-of-sight wireless communication over short distances [1]. Line-of-sight requirement and short communication distance are handicaps for a general-purpose wireless communication protocol, but they become advantageous in a densely-packed data center network (DCN) by reducing the interference with nearby concurrent communications, therefore increasing throughput across the data center [2]. IEEE 802.11ad standardizes the use of 60 GHz band [3,4].

There are two approaches to using wireless networking in data centers: completely wireless and hybrid. In completely wireless data centers (WDCs), all communication between servers is wireless — there is no wired communication [5,6]. A completely wireless data center has a very different physical organization than a traditional data center. In hybrid wireless data centers, wireless communication is used to assist wired network [7,8].

In this paper, we focus on hybrid wireless data centers because they are more applicable in short-term than completely wireless data centers. Existing data centers could be equipped with wireless networking devices with little effort. Top-of-the-rack (ToR) switches are good candidates for radio placement, so that racks can communicate wirelessly in addition the wired network.

Wireless resources are used in data centers to increase capacity at the bottlenecks of the wired network [9–11]. The traditional method of addressing bottlenecks in the wired network is to increase capacity by eliminating oversubscription to meet worst-case traffic requirements [12,13]. Such methods require substantial capital cost because of the increased need for network equipment and the cost of wiring a larger network. Operating costs are also increased because of the power consumption and maintenance cost of a larger infrastructure. Modifying the wired network, or scaling it to increase capacity or to support more devices is also cumbersome [14]. In hybrid data centers, wireless network supports the wired network to address these problems. Wireless links can be established at the bottlenecks dynamically, increasing capacity only when and where needed. Wireless networks are also easier to modify or scale than wired networks.

Bottlenecks, also called hotspots, may occur between 5 – 10 switches in a network of 1500 servers running a Map-Reduce job [9]. Other analyses of data center traffic find that even though core switches carry a higher traffic load than edge switches, edge switches have higher link loss because of high outburst traffic [15,16]. Alleviating hotspots can be achieved by routing the traffic to a neighboring ToR switch and forwarding it to the bottlenecked one using wireless communication,

* Corresponding author.
E-mail addresses: mergenci@cs.bilkent.edu.tr (C. Mergenci), korpe@cs.bilkent.edu.tr (I. Korpeoglu).

https://doi.org/10.1016/j.comcom.2020.12.029
Received 8 May 2020; Received in revised form 3 December 2020; Accepted 30 December 2020
Available online 23 January 2021

therefore increasing bandwidth only at the bottleneck [11]. An alternative is to connect two sets of hot servers directly over wireless network, using multiple radios when necessary [10].

Existing studies employ single-hop wireless communication. Multi-hop communication utilizes available network-wide wireless bandwidth better than single-hop communication. It also enables a more flexible arrangement of links that can be configured according to changing traffic requirements of the data center.

We examine the problem of offloading as much traffic as possible from wired to wireless network, so that hybrid data centers could be designed accordingly. We aim to analyze capabilities of a multi-hop wireless network by quantifying the amount of traffic carried, multi-hop path length, and throughput in a data center setting. The results can be used by data center designers to design a hybrid wired and wireless network that is more efficient to build, operate, maintain, and expand than traditional data center network designs.

A static arrangement of multi-hop wireless links cannot adapt to varying traffic needs between nodes, whereas a dynamic arrangement offers the flexibility to allocate wireless links to where they are needed. We propose multi-hop routing algorithms that assign traffic flows to wireless links in a hybrid data center. Traffic flows are evaluated in the ascending order of wireless hop distance between their source and destination. Source–destination pairs that are nearby are assigned to wireless links before the ones that are more distant to each other. The basic premise is to create longer routes between distant nodes from shorter routes that connect closer ones. The required wireless link configuration to assign a new flow may conflict with the existing configuration. In that case, the traffic flowing over the conflicting links may need to be deallocated. A cost–benefit analysis determines the result. The wireless link configuration that carries more traffic is preferred. In other words, allocation is greedy with respect to traffic amount.

Our proposed algorithms are run periodically, to assign traffic according to changing traffic needs of the data center. At the beginning of each period, an external traffic estimator outputs expected traffic exchange between ToR switches during that period. Our algorithms take this estimate as input, and output a configuration of wireless links. Reconfiguring wireless links costs bandwidth, because the traffic that has no route to its destination in the new configuration needs to be dropped. In addition, broadcasting the new configuration and making sure that all nodes have completed their configuration takes time, during which the wireless network cannot be used efficiently. Therefore, we aim to maximize the amount of traffic carried for a given configuration. The traffic that is not assigned to wireless network, flows over the wired network as usual. Our extensive simulation results show how much traffic can be offloaded to wireless network, so that data center networks could be designed accordingly.

Rest of the paper is organized as follows. Section 2 summarizes related work, discusses how our proposed methods differ, and lists our contributions. Section 3 presents the system model and its rationale. Section 4 discusses our proposed traffic allocation methods in detail. Section 5 analyzes properties of a real data center traffic and proposes a method to randomly generate traffic with those properties. Section 6 describes our simulations and discusses results. Section 7 concludes the paper.

2. Related work

[9] presents an analysis of data center traffic between ToR pairs. Authors argue that only a few ToR pairs exchange very high amount of traffic at a given time, therefore it is an overkill to eliminate oversubscription in the wired network. Rather, wireless communication can be used on demand to increase the capacity at congested points. The idea of allocating more resources at necessary points is called flyways. Flyways are applied to a real data center environment in [11]. Authors measure capabilities of 60 GHz communication in a data center environment. Based on the results, they propose a system that uses one radio per ToR switch to offer additional bandwidth from one of the neighboring racks in case of network congestion. Authors also propose several methods to determine which wireless flyways to establish based on the traffic demand in the network.

We use a similar deployment of radios as in [9] and [11], except we propose two radios per ToR switch as presented in Section 3.

[10] and [8] address the same problem of alleviating hotspots in the wired network with a different method. Rather than using one radio per ToR switch, a set of servers are grouped into so called Wireless Transmission Units (WTUs). Authors note that although a WTU may correspond to a rack in certain data center architectures, the idea of WTU generalizes to other architectures as well. The problem is to schedule wireless links between WTUs according to a utility value based on the distance between nodes and traffic demand. Min–max and best-effort scheduling methods are proposed as solutions. Results show that both methods perform similarly when the traffic distribution is unbalanced across the network. When the traffic is uniform, min–max performs better than best-effort scheduling.

[17] and [14] present a different deployment scheme of wireless networking equipment. Because 60 GHz is restricted to line-of-sight communication, a signal reflecting surface is installed above the antennas so that they do not block the line of sight between other racks. Nodes are no longer restricted to communicating to their immediate neighbors as in [9,10], and [8], therefore the system achieves better performance.

Wireless networking in DCNs is also used for other purposes. [18] uses wireless communication as a facilities network in the data center. Authors argue that wireless network is more suitable for the control plane of Software Defined Networking [19,20] and management tasks of cloud provider, rather than enhancing capabilities of the data plane. [21] addresses the concern between control and data plane separation for a multiple-input multiple-output (MIMO) wireless DCN setting. Authors propose to replace wires between rows of racks with a MIMO wireless crossbar. [22] presents another take on MIMO wireless data center networking. [23] addresses carrying multicast traffic in wireless DCNs. [24] presents another multicast traffic management approach that uses multiple channels.

[25] and [26] provide surveys of using wireless communication in DCNs.

In this paper, our motivation is to discover the potential of wireless infrastructure when it is used with multi-hop routing. [17] and [14] acknowledge that multi-hop communication could be used as an alternative to their method. They consider the decreased throughput because of half-duplex communication as a potential drawback. We address this issue by using one-way communication in wireless links, so that full bandwidth of the channel can be used. Because the data center traffic displays an asymmetric traffic pattern, as presented in Section 5, the lack of the reverse communication direction is not important. The other drawback they present is the latency introduced by multi-hop forwarding, nevertheless they do not quantify it. In this study, we show that the latency could be kept as low as few hops even in large networks while carrying a big portion of the traffic wirelessly.

We note, however, that communicating over a reflective surface and over multiple hops are not mutually exclusive methods. To the contrary, these methods complement each other. By using a reflective surface, each hop could reach farther physical distance in the data center floor, therefore the benefits of multi-hop communication would be augmented.

To sum up, our contributions are as follows:

- We define a practical system model for hybrid wireless data centers.
- We propose multi-hop routing algorithms that utilize the wireless infrastructure to its potential.
- We propose a method to randomly generate data center traffic based on a real data center traffic pattern.
- We verify and evaluate our proposed methods with simulations.

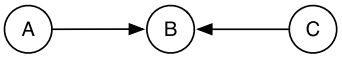

Fig. 1. An example of interference at node B with a single channel.

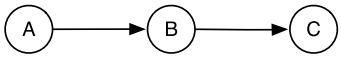

Fig. 2. Another example of interference at node B with a single channel.

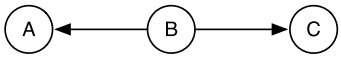

Fig. 3. An example of no interference at node B with a single channel.

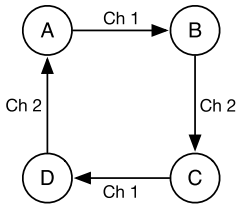

Fig. 4. An example of a traffic pattern that could be carried over two channels.

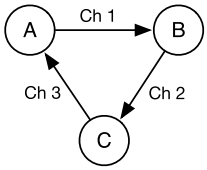

Fig. 5. An example of a traffic pattern that cannot be carried using two channels.

3. System model

The properties of a wireless network in our hybrid data center system model can be defined by the following parameters. We provide a detailed discussion of how these parameters are chosen in the following subsection.

1. Wireless communication technology: Data center networks carry a lot of traffic; therefore, bandwidth is one of the most important parameters of wireless communication technology. Like many other studies, we assume use of 802.11ad, which works in 60 GHz ISM band.
2. Number of radios per node: The number of radios per node determines the maximum number of wireless connections a node can establish with its neighbors. We assume the availability of two radios per node, per ToR switch.
3. Number of channels: Dividing the frequency band into channels enables multiple communications to work simultaneously. The number of radios also affects the number of required channels. In our work, we show that using three non-overlapping channels of 802.11ad is a good choice when there are two radios per node.
4. Communication distance: The physical distance between nodes affects wireless communication quality significantly. A good assignment of wireless links should consider the effect of the distance between nodes. We safely assume that wireless communication could be performed with necessary quality up to 5 meters.
5. Unit of traffic: The traffic flow requirement between two nodes in a configuration period (i.e., until the next execution of the allocation algorithm to reconfigure the wireless radios and links) can be expressed as the total amount of traffic to be carried or as the required bandwidth of the flow. In this study, we consider traffic requirement as the total amount of traffic to be communicated between nodes in one configuration.

3.1. Rationale

When wireless links are assigned in a centralized way, there is no need to run a full-scale medium access control (MAC) protocol between nodes. Nodes would know with which other nodes they would communicate with, using which radio and which channel. Therefore, the cost of using a MAC protocol could be reduced.

Fig. 1 shows a network in which nodes A and C communicate with node B. If nodes have multiple radios, A and C can simultaneously communicate with B provided that they use different channels. Otherwise, transmissions will interfere at node B. We consider two counter-arguments to this statement.

By using directional antennas the effect of interference at node B could be reduced. The amount of interference depends on the angular difference between two wireless links. This requires that the spatial location of the nodes to be considered when assigning wireless paths. We evaluate that such a consideration would bring unnecessary complexity compared to using different channels for such cases.

A MAC protocol is also not enough to solve the problem of interference at node B. It is very likely that nodes A and C cannot hear each other, when the physical arrangement of data centers and the short distance communication properties of 802.11ad technology is considered. Therefore, we prefer to use a different channel rather than using a MAC protocol in such cases.

Fig. 2 shows an example of interference at node B that could be addressed by using a MAC protocol. This configuration could be for a traffic pattern in which A is sending traffic to C over B, or A is sending traffic to B while B is sending another traffic to C independent of the traffic A is sending to B. Even when directional antennas are used and when the angular difference between nodes are at an optimum value, it is not possible to conduct both communications simultaneously over the same channel. The transmission power from B to C would be too high compared to the power of the signal heard at the other radio of B. If a single channel is to be used, one of the traffics should be canceled if they are separate. If there is a single traffic flowing from A to C, then it should be canceled.

A third pattern of communication is depicted in Fig. 3. Node B is transmitting to both A and C simultaneously. Regardless of the spatial location of the nodes and the MAC protocol, B could use the same channel for both communications. We assume that nodes correspond to top-of-the-rack (ToR) switches and the physical distance between racks allow both traffic to be transmitted without interference [11].

Fig. 4 shows the channel assignment structure in a larger context. Wireless links between nodes can carry multi- or single-hop traffic. Each node has two radios, one for incoming and one for outgoing traffic. Two channels are necessary to transmit all traffic simultaneously. Channels can be assigned in an alternating fashion as shown in the figure.

Fig. 5 shows a slightly modified version. Even though each node has two radios, one for incoming and one for outgoing traffic as in the previous case, this time two channels are not enough to transmit all traffic simultaneously. When channels are assigned in an alternating fashion, it is clear that at least three channels are needed.

As demonstrated in Fig. 3, the direction of flows may reduce interference, and therefore, the required number of channels. However, in the worst-case, each link established at a node may interfere with each other. Therefore, we use an undirected graph to represent wireless connections. Channels assigned to wireless links could be thought of as colors assigned to edges of the graph (edge-coloring problem). According to [27], the edge chromatic number, $\chi'(G)$, of an undirected graph is determined by the formula

$$\Delta(G) \leq \chi'(G) \leq \Delta(G) + 1 \quad (1)$$

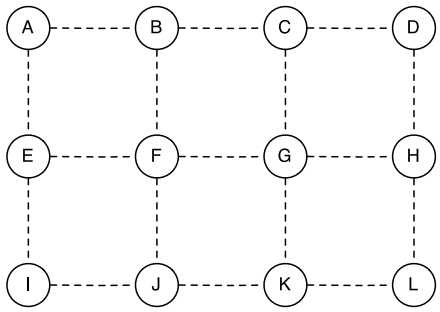

Fig. 6. Possible wireless connections between ToR switches.

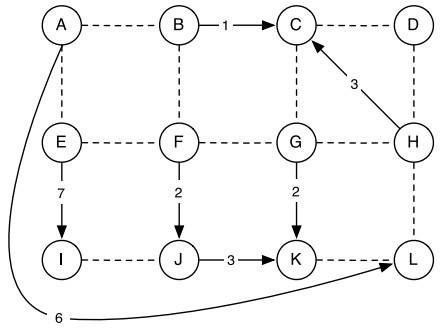

Fig. 7. Amount of traffic flowing between nodes.

where $\Delta(G)$ is the degree of the maximum degree vertex in the graph. In our system, each node has N radios, therefore the maximum number of connections a node could establish with other nodes is N. We conclude that $\Delta(G) = N$.

Given any graph, determining if $\chi'(G) = \Delta(G)$ or $\chi'(G) = \Delta(G)+1$ is NP-complete [28]. Therefore, there is no guarantee that an assignment of wireless links could be allocated N channels. Finding an assignment that could be satisfied with N channels is not practical either.

As a result, we conclude that N+1 channels is an appropriate choice. The number of channels in 802.11ad varies between 3 and 6 by region. In order to have the widest applicability of our system model, we decide to use 2-radio nodes with each radio using one of 3 wireless channels.

4. Proposed method

Fig. 6 shows, in dashed lines, possible wireless links that could be established between ToR switches. In practice, ToR switches could establish wireless connections to more distant ones as well, but to simplify the example it is assumed that only immediate neighbors could communicate wirelessly.

Fig. 7 displays the amounts of traffic that needs to flow between nodes. Table 1 shows the corresponding traffic matrix. Empty cells have a value of zero. Rows and columns that consist of only zero elements are omitted for brevity. 1 unit of traffic flows from Rack B to C, while Rack H sends 3 units of traffic to C. A single wireless link is needed to carry the traffic from Rack B to C. To carry the traffic from H to C, a multi-hop path of at least 2 hops should be allocated.

Routing traffic over the shortest path has several benefits. First of all, latency would be minimal. Secondly, resources would be used more efficiently by not allocating more links than necessary. Because the number of transmissions is minimal, it would cause the minimum amount of potential interference with other transmissions. Finally, it would conflict with fewer number of other flows. Nevertheless, it might be possible to have less conflict by allocating more hops to some of the flows. We compare both methods in our evaluations.

Assume that all single-hop traffic is already allocated. Considering that every node has 2 radios, when the traffic from H to C is routed

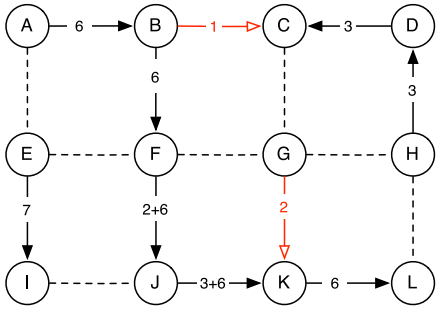

Fig. 8. Allocation of wireless links to flows maximizing total traffic flow.

Table 1
Traffic matrix corresponding to Fig. 7.

		From						
		A	B	E	F	G	H	J
To	C		1				3	
	I			7				
	J				2			
	K					2		3
	L	6						

over G, the traffic from G to K should be canceled, because one of the radios of G will be allocated to receive traffic from H, and the other will be allocated to transmit to C. The amount of traffic flowing from H to C, 3, is higher than the amount of traffic from G to K, 2. Therefore, all other things being equal, allocating larger traffic instead of the smaller one makes wireless network carry more traffic without requiring reconfiguration of wireless radios and links, which involves steering the radio antennas to right directions (neighbors), setting channels, and establishing links. On the other hand, the traffic from H to C could be routed over D. Node D is not going to carry any other traffic, so the cost of routing the traffic over D is less than the cost of routing it over G.

The lowest conflict path for the traffic from A to L is over nodes B, F, J, K respectively. In that case, flows from B to C, and G to K will be deallocated, costing 3 units of traffic in total. Though, the benefit of allocating the traffic from A to L is 6 units, which is greater than its cost.

Fig. 8 presents a multi-hop allocation of wireless links to traffic flows that maximizes the total amount of traffic carried over the network. If only single-hop communication were used, the total amount of traffic would be 15 units. Using multi-hop communication the total traffic amount is increased to 21 units.

A greedy algorithm that assigns multi-hop wireless links to traffic flows is given in Algorithm 1. The algorithm accepts as input a graph (G), a set of traffic flows (F), and the amount of traffic flows (τ). It returns a set of paths corresponding to assigned flows. G is an undirected graph of all possible wireless links between racks, similar to the one depicted in Fig. 6. F is a set of source–target pairs. τ is a mapping from source–target pairs to their traffic amount (cost).

After such an allocation is done, the wireless network is configured to carry the flows. Each radio in each node should steer its antenna to the right direction (to the relevant neighbor) to send or receive data traffic, so that the links decided by the allocation algorithm to carry assigned flows can be established among the nodes.

At the beginning of the algorithm, the set of wireless link allocations, *alloc*, for each flow is initialized. A wireless link is allocated to a flow in Line 21. Cost calculation of a potential allocation uses current set of allocations in Line 14.

Flows are assigned a shortest path from source to target. Lines 5–10 calculate the distance from source to target, δ, using a breadth-first

Algorithm 1 Allocate fly-paths to flows over shortest paths

```
FLY-PATH-SP(G, F, τ)
    ▷ G: Wireless graph, F: set of flows, τ: traffic demands
 1: for all (u, v) ∈ E[G] do
 2:     alloc(u, v) ← ∅                          ▷ An allocation is a set of flows
 3:     alloc(v, u) ← ∅
 4: end for
    ▷ Calculate length of flows as BFS distances
 5: for all s ∈ {s | (s, t) ∈ F} do
 6:     BFS(G, s)
 7:     for all t ∈ {t | (s, t) ∈ F} do
 8:         δ(s, t) ← d[t]                       ▷ d[t] is BFS distance from s to t
 9:     end for
10: end for
11: for all (s, t) ∈ SORT_{<(δ,−c)}(F) do
12:     G_BFS ← BFS(G, s)
        ▷ Calculate costs of edges in BFS DAG
13:     for all (u, v) ∈ E[G_BFS] do
14:         cost(u, v) ← POTENTIAL-COST(s, t, u, v, alloc)
15:     end for
16:     DIJKSTRA(G_BFS, cost, s)              ▷ Find path costs to target, d[t]
17:     if d[t] < τ(s, t) then                ▷ Path is feasible: cost < demand
18:         u ← t
19:         path(s, t) ← nil                  ▷ List of allocated edges
20:         do
21:             alloc(π[u], u) ← alloc(π[u], u) ∪ {(s, t)}
22:             INSERT-HEAD(path(s, t), (π[u], u))
                ▷ Deallocate conflicting flows
23:             DEALLOC(π[u], u, alloc, path, cost)
24:             u ← π[u]
25:         while u ≠ s
26:     end if
27: end for
28: return path
```

search on G. The for loop in Line 11 iterates over flows in increasing hop count and decreasing traffic. Beginning from a single-hop flow with highest traffic demand, first the other single-hop flows with lower traffic demands are assigned wireless connections, and then flows with higher number of flows are considered.

Paths of flows are chosen among the paths that BFS algorithm traverses. BFS begins from the source, and progresses towards the target hop by hop. Potential cost of assigning a link (hop) to the flow is calculated in Line 14. Potential cost is the cost that will be incurred because of a conflict with other flows, in case the link is assigned to the flow.

Potential cost is kept as a set of flows that need to be canceled, if the link is assigned to the current flow. The value of potential cost is the sum of costs (traffic demands) of these flows. When path costs, $d[t]$, are calculated by Dijkstra's algorithm in Line 16, the costs are aggregated as a set (using set union operation) and finally converted to its numeric value using Eq. (2).

$$\sum_{(x,y) \in cost(u,v)} \tau(x, y) \tag{2}$$

We cannot keep track of only the value of potential cost, because in that case the value of a flow that conflicts on multiple vertices would be counted multiple times. Keeping potential cost as a set of flows and then calculating the value when needed prevents multiple counting.

Fig. 9 illustrates the case when edge (A, B) is assigned to flow (s, t), $f_{s,t}$. The nodes that are neighbors of A or B, except A or B themselves, are called x. We also assume that neither A nor B are s or t; therefore, they are called intermediate nodes.

As a convention, we consider that, for intermediate nodes, incoming traffic conflicts with incoming traffic and outgoing traffic conflicts with outgoing traffic. Because intermediate nodes need to assign two radios to a flow, one for receiving and one for transmitting, this assumption covers all possible conflicts at each intermediate node. The fact that flow (s, t) is assigned link (A, B) implies that A has already used one of its radios to receive the traffic. When A transmits it to B, the flows that are transmitted to other neighbors of A need to be canceled. Therefore, the cost includes the set of flows transmitted from A to x. A similar argument applies to B. B will use one of its radios to transmit $f_{s,t}$, therefore it should use the other one to receive it. By our convention, the flows that are received by B from x need to be canceled.

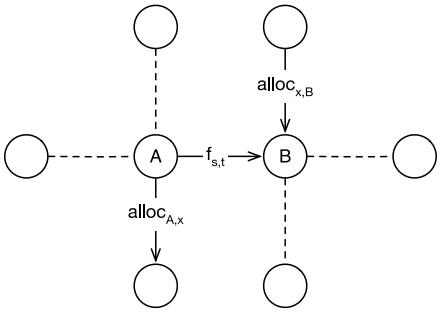

Fig. 9. Potential cost of assigning (A, B) to flow (s, t): Potential-cost(s, t, A, B, alloc).

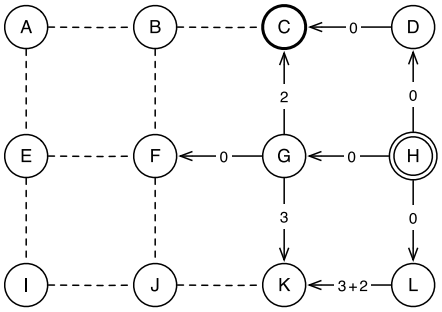

Fig. 10. BFS graph for traffic flow from H to C, and the values of potential conflicts as edge labels.

Formally, potential cost is defined in Eq. (3).

$$\text{Potential-cost}(s, t, u, v, alloc) = \bigcup_x alloc(u, x) \cup \bigcup_x alloc(x, v),$$
$$\forall x \in Adj[u] \cup Adj[v] - \{u, v\} \text{ if } u \neq s, v \neq t \tag{3}$$

When node A is s, or node B is t, potential cost calculation is similar to the ones for intermediate nodes with some differences. Source and target nodes use a single radio for transmission or reception. When at least one of the radios are unassigned, the cost of allocating a new flow to the node is zero. When both radios are in use, the radio with the lower-cost flow set is considered to be conflicting.

Fig. 10 shows BFS graph, calculated in Line 12, during the allocation of flow (H, C). Edge labels denote the values (costs) of potential conflicts. Because the traffic flow ends at C, which is 2 hops away from H, edges beyond 2 hops are not traversed.

Fig. 11 presents existing allocations during the allocation of flow (H, C). $f_{H,C}$ is the first 2-hop flow to be allocated, therefore all single-hop flows have been allocated. Given the allocation, we can calculate the potential cost of allocating any edge to a new flow.

Consider routing the flow (H, C) over G. First, we examine the cost of allocating link (H, G). By definition, it consists of flows leaving H and flows arriving at G. There are no flows associated with H, so the first component does not contribute to the cost. There are no flows arriving at G either. There is one leaving G, $f_{G,K}$, but its cost contributes to the cost of allocating (G, C), not (H, G); because, by definition, outgoing traffic is considered to conflict with outgoing traffic. Therefore, the cost

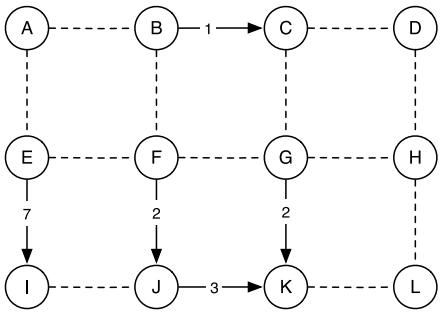

Fig. 11. State of allocations when trying to allocate flow (H, C).

of allocating link (H, G) to flow (H, C) is ∅ with value zero, as shown in Fig. 10.

Next, we calculate the cost of allocating link (G, C). We consider that one radio of G has already been allocated to the current flow, $f_{H,C}$, to receive the data form H. Therefore, we need to cancel other transmissions leaving G, to transmit data to C. There is one such transmission, from G to K, that belongs to flow (G, K). The other component of the cost consists of flows arriving at C. There is one such flow, $f_{B,C}$. Because C is the target node, it does not need to forward $f_{H,C}$ further, therefore it can use its other radio to receive from G without any conflicts. We conclude that the cost of allocating link (G, C) to flow (H, C) is $\{f_{G,K}\}$, and its value is 2.

While constructing G_{BFS} we only find the potential cost of allocating links. Therefore, the algorithm also calculates the cost of allocating link (G, K) to $f_{H,C}$, even though the route will not lead to C. In case this edge is used, $f_{G,K}$ need not be canceled. However, the flow (J, K) arriving at node K needs to be canceled because K needs to forward the traffic to reach the target, C. Therefore, the cost of allocating (G, K) to flow (H, C) is $\{f_{J,K}\}$, with value 3.

Once BFS graph is constructed for a flow, lowest cost path from source to destination is found using Dijkstra's shortest path algorithm in Line 16. Because costs are sets of flows, aggregate path cost is calculated by the set union operator rather than the conventional addition of costs. Only when the path cost is compared to traffic demand of the current flow in Line 17, value of the path cost is calculated as defined in Eq. (3).

In case the cost of allocating the flow is less than its traffic demand, we deallocate conflicting flows in Line 23, and allocate the current flow in Line 21. The allocated path is constructed by following parents, π, of each node beginning from the target back to the source, Lines 19 and 22.

4.1. Alternative method

Algorithm 1 routes traffic flows through a shortest path. Considering the fact that each additional hop uses more resources, creates more potential for conflicts, and increases latency, it is wise to use shortest paths. In some cases, though, longer paths may reduce conflicts by routing around busy nodes. Algorithm 2 is an alternative method that may allocate flows to routes longer than shortest paths. Please note that even though SP stands for "shortest path", LP does not stand for "longest path", rather LP stands for "long*er* paths". Any path that is not a shortest path between two nodes is considered to be *long*.

The differences from Algorithm 1 are in Line 12 and Line 16. Rather than calculating edge costs for edges in the BFS DAG, it calculates costs of all edges in the graph, so that if any path is feasible for this flow, it will be allocated. This method is costlier in terms of computation, though it might allocate more traffic. Performances of these two methods will be compared in simulations.

Algorithm 2 Allocate fly-paths to flows over longer paths

```
FLY-PATH-LP(G, F, τ)
    ▷ G: Wireless graph, F: set of flows, τ: traffic demands
    ...
11: for all (s, t) ∈ SORT_{≺(δ,−c)}(F) do
12:                                          ▷ This line is removed
13:     for all (u, v) ∈ E[G] do             ▷ Calculate costs of edges
14:         cost(u, v) ← POTENTIAL-COST(s, t, u, v, alloc)
15:     end for
16:     DIJKSTRA(G, cost, s)                 ▷ Find path costs to target, d[t]
        ...
27: end for
28: return path
```

4.2. Unassigned flows

Our proposed methods do not guarantee allocating a path to a flow. Such a guarantee is possible, only when G is Hamiltonian, i.e., links could be assigned such that a single path visits all nodes. Obviously, a Hamiltonian route maximizes the amount of traffic carried, but it is very impractical because of the latency it would introduce even between physically close nodes. Considering that the primary purpose of the wireless connections in a hybrid data center is to assist the wired infrastructure, it is acceptable that only the most important flows be allocated. Unassigned traffic could still be routed over the wired network.

A practical issue in deployment of proposed methods is when to use them. Wireless links allocated at one time might not be useful in future, so the allocation algorithms should be run periodically with the estimated traffic for that period. Configuration of wireless radios and channels to establish links is done once at the beginning of each period and is valid until the end of the period. A period should be long enough to compensate the cost of reconfiguring the wireless radios and links, and short enough to make traffic estimates reliable.

4.3. Algorithm variations

In both of the algorithm versions, it is possible that a flow is unassigned even though a path is established between its source and target. This can happen in two cases. In the first case, the flow was never allocated, because the cost of allocating it was greater than the amount of traffic it carries. The path between its source and target is assigned to a conflicting flow with larger traffic. In the second case, the flow f_1 had been allocated initially, but later, was deallocated because of a conflicting higher-traffic flow f_2. f_2 gets deallocated in turn, because of an even higher-traffic flow f_3 that reestablishes the path of f_1. In both cases, these unassigned flows can be allocated a path without incurring any cost, because a path between their source and target is already established by other flows. We call such allocations cost-free, or conflict-free allocations.

Cost-free allocations can be handled at different times during the allocation algorithm. The simplest case is after all the flows have been assigned, as shown in Algorithm 3. We designate such versions with "+CF" suffix to the algorithm names (fly-path-SP+CF, fly-path-LP+CF).

Cost-free allocations can also be evaluated after each allocation. In this case, there are two types of cost-free allocations: (1) flows that have been deemed infeasible up to that point during the execution of the algorithm (Algorithm 4), and (2) all unassigned flows, including infeasible ones and the ones that have not yet been considered for allocation (Algorithm 5). We suffix the former case versions with "-CF-infeasible" (fly-path-SP-CF-infeasible, fly-path-LP-CF-infeasible), and the latter case versions with "-CF-unassigned" (fly-path-SP-CF-unassigned, fly-path-LP-CF-unassigned). Together with the original versions we propose 8 algorithms in total.

Algorithm 3 Allocate cost-free paths (CF version).

```
FLY-PATH+CF(G, F, τ)
    ▷ G: Wireless graph, F: set of flows, τ: traffic demands
    ...
28: ALLOCATE-COST-FREE(G, F, τ, alloc, path)
29: return path
```

Algorithm 4 Allocate cost-free paths (CF-infeasible version).

```
FLY-PATH-CF-INFEASIBLE(G, F, τ)
    ▷ G: Wireless graph, F: set of flows, τ: traffic demands
1:  for all (u, v) ∈ E[G] do
        ...
4:      infeasible ← ∅                          ▷ Set of infeasible flows
5:  end for
    ...
12: for all (s, t) ∈ SORT_{<(δ,−c)}(F) do
        ...
18:     if d[t] < τ(s, t) then                  ▷ Path is feasible: cost < demand
            ...
21:         do
22:             infeasible ← infeasible ∪ cost(π[u], u)
                ...
27:         while u ≠ s
28:         ALLOCATE-COST-FREE(G, F, τ, alloc, path, infeasible)
29:     else
30:         infeasible ← infeasible ∪ {(s, t)}
31:     end if
28: end for
29: return path
```

Algorithm 5 Allocate cost-free paths (CF-unassigned version).

```
FLY-PATH-CF-UNASSIGNED(G, F, τ)
    ▷ G: Wireless graph, F: set of flows, τ: traffic demands
    ...
11: for all (s, t) ∈ SORT_{<(δ,−c)}(F) do
        ...
17:     if d[t] < τ(s, t) then                  ▷ Path is feasible: cost < demand
            ...
26:         ALLOCATE-COST-FREE(G, F, τ, alloc, path)
27:     end if
28: end for
29: return path
```

Cost-free allocation modifications presented in Algorithms 3, 4, and 5 are the same for both SP and LP versions given in Algorithms 1 and 2 respectively. Algorithm 3 allocates cost-free traffic once at the end of the algorithm in Line 28. Algorithm 5 performs the same task after each allocation in Line 26. ALLOCATE-COST-FREE function first determines the set of unassigned flows. Then, it checks if existing allocations constitute a path between source and destination pairs of unassigned flows. Unassigned flows that have a path between their source and destination are allocated in a cost-free manner by updating alloc and path data structures.

Algorithm 4 keeps track of infeasible flows in order to be able to allocate them in a cost-free manner. The flows that have never been allocated are added to the set of infeasible flows in Line 30. The flows that are deallocated because of a conflict are added to the set of infeasible flows in Line 22 ($\text{cost}(\pi[u], u)$ is the set of conflicting flows that are deallocated). ALLOCATE-COST-FREE function, in this case, is assumed to operate on the set of infeasible flows given as an argument, rather than determining the set of unassigned flows on its own.

Computational cost of cost-free allocations is worth discussion. Algorithm 3 allocates cost-free flows once, therefore has a negligible overhead compared to the original algorithms. Algorithms 4 and 5 are computationally costlier than Algorithm 3, because they perform cost-free allocations after each allocation. Algorithm 4 performs fewer computations than Algorithm 5, because the set of infeasible flows is a subset of unassigned flows. Runtime performance is not a main concern of our study, therefore we do not report it in our results. Nevertheless, we take computational cost into account when deciding on which algorithm to use in practice in Section 6.

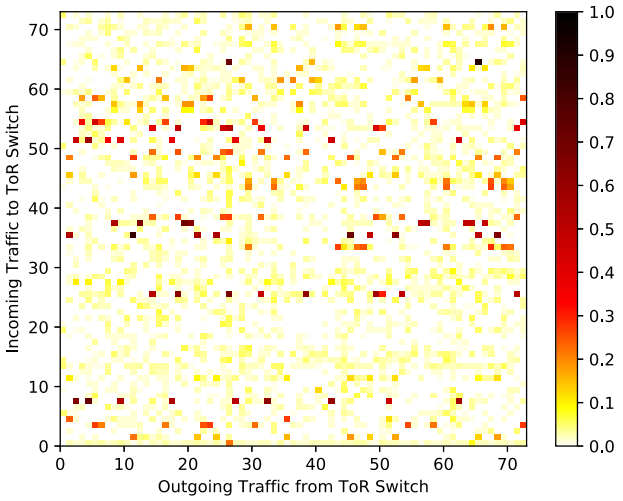

Fig. 12. The original sample Cosmos traffic pattern. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

5. Data center traffic

Performance of the algorithms depend on the data center traffic. We generate a random traffic pattern between nodes according to characteristics of Cosmos data center traffic presented in [11] and [9]. Cosmos dataset is the traffic of a Map-Reduce [29] workflow running on O(1K) servers. In order to generate a similar traffic pattern to Cosmos, we first analyze its properties.

5.1. Cosmos data center traffic analysis

Cosmos dataset is not public. We reverse-engineer the traffic pattern given in Figure 11 of [11], reproduced in Fig. 12.

Colors represent the amount of traffic exchanged between ToR switches in logarithmic scale with largest value D corresponding to black. The deep red color at value 0.5 represents a traffic amount of $\sqrt{D}$, and traffic less than $D^{0.1}$ are represented with shades of yellow.

In order to find the amount of traffic flowing between ToR switches, we first map RGB values of the color scale to exponent values and then interpolate colors of the plot according to the mapping. Finally, we use exponent values to find the amount of traffic in linear scale.

RGB values of the color scale, however, does not match with the description above, because RGB values are in AdobeRGB color space, which is the color space of the Portable Document Format (PDF). When we transform colors into sRGB color space, we observe that the values below 0.1 correspond to yellow, as in the description. The fact that the plot seems to be generated with Python matplotlib library verifies the color transformation into sRGB space, because transformed values can easily be represented with a LinearSegmentedColormap object of matplotlib.

We use the linear transformation defined in Eq. (4) to find exponents. The linear coefficient results from our examination of the sRGB color scale. Exponent values from 1.00 to 0.33 are represented by an increase in red component from 0 to 255. Green and blue components

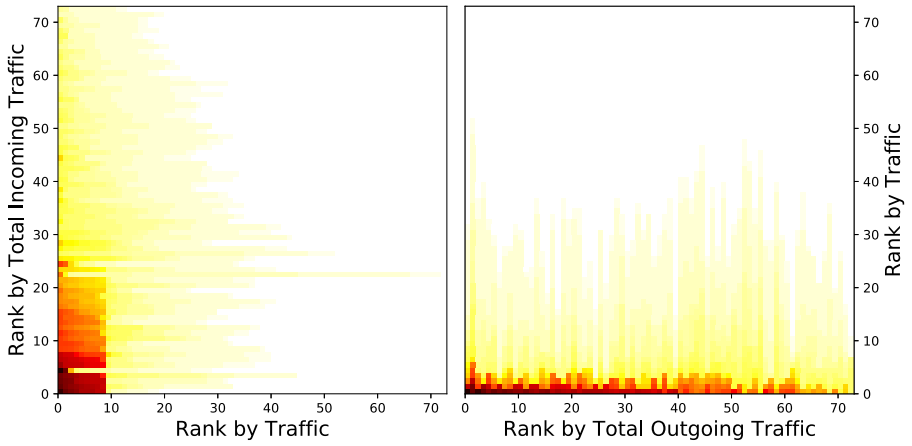

Fig. 13. The original sample Cosmos traffic pattern sorted by total incoming traffic (left) and total outgoing traffic (right).

for this value range are 0. Therefore, an exponent of 1.00 corresponds to $[0\ 0\ 0]_{\text{sRGB}}$ (black), 0.33 corresponds to $[255\ 0\ 0]_{\text{sRGB}}$ (red), and the values in between are linear interpolations of these two. Note that $0.33 = 1 - [255/255\ 0\ 0]_{\text{sRGB}} \cdot [0.67\ 0.24\ 0.09]^T$, therefore Eq. (4) holds.

$$\text{Exponent} = 1 - \begin{bmatrix} R/255 & G/255 & B/255 \end{bmatrix}_{\text{sRGB}} \cdot \begin{bmatrix} 0.67 \\ 0.24 \\ 0.09 \end{bmatrix} \tag{4}$$

Red component remains constant at 255 for all values below 0.33. Exponent values from 0.33 to 0.09 are represented by an increase in blue component from 0 to 255. Green component remains 0 for this value range. Therefore, an exponent of 0.33 corresponds to $[255\ 0\ 0]_{\text{sRGB}}$ (red), 0.09 corresponds to $[255\ 255\ 0]_{\text{sRGB}}$ (yellow). For the values below 0.09, red and blue components remain constant at 255 while green component increases from 0 to 255. An exponent of 0.00 is represented by $[255\ 255\ 255]_{\text{sRGB}}$ (white).

The next step is to find the amount of traffic in linear scale using the exponent. A value of 0 in linear space cannot be represented by an exponent; nevertheless, the original paper states that white corresponds to no traffic. Therefore, we subtract 1 from the traffic value raised to the exponent, as defined in Eq. (5). Remember that D is the maximum amount of traffic between two ToR switches in the data center. When exponent is 0, traffic value is 0; when exponent is 1, traffic value is D, as in the definition.

$$\text{Traffic} = (D+1)^{\text{Exponent}} - 1 \tag{5}$$

Eq. (6) defines the inverse transformation, i.e., the transformation from linear traffic amount to exponent value. A traffic value of 0 yields an exponent of 0, and a traffic value of D yields an exponent of 1, as in the definition.

$$\text{Exponent} = log_{D+1}(\text{Traffic} + 1) \tag{6}$$

The value of D is not given in the referenced literature, [11]. We assume that $D = 100$ in our simulations.

Now that we know the linear traffic values exchanged between ToR switches, we can analyze the traffic better by finding the total incoming traffic to or total outgoing traffic from a ToR switch.

Referenced work states two properties of the traffic: (1) most of the ToR pairs exchange low amount of traffic, (2) hot ToRs exchange high amount of traffic with only a few other ToRs.

Authors also state that the second property is apparent by horizontal and vertical streaks in Fig. 12. We note, however, that horizontal streaks are more dominant than vertical streaks. In order to compare them better, we sort the data by total incoming and outgoing traffic in Fig. 13. Each row and column is also sorted in descending order by the amount of traffic from left to right and bottom to top respectively.

The most important observation is the difference between incoming and outgoing traffic patterns. While almost all nodes send a high amount of traffic to 1–4 other nodes (seen on the right-side plot), this traffic is received by only around 25 nodes (seen on the left-side plot). It is also notable that almost all of these 25 nodes receive their traffic from around 9 other nodes.

5.2. Traffic generation

Based on our analysis of Cosmos data center traffic, we propose a 2-step procedure to randomly generate data center traffic: (1) randomly generate lighter-colored base traffic, and (2) randomly generate darker-colored hotspot traffic. Our purpose is to *simulate* a data center traffic from common probability distributions, so that randomly generated traffic patterns have similar characteristics with some variation that depends on the chosen distributions, parameters of which can also be modified to obtain custom traffic patterns.

Fig. 14 shows the distribution of the original Cosmos pairwise traffic between nodes, except hotspot traffic (top 2%). Bar plots and curved line plot are associated with left y-axis and show the density of the distribution. Cumulative step plots are associated with right y-axis and show the cumulative distribution. Red lines show an exponential distribution fitted to the original traffic data using *expon.fit()* function in *stats* module of *SciPy* python library [30]. The result of *expon.fit()* is $loc = 0.0000$ and $scale = 0.0749$. The largest difference between the original traffic data and the randomly generated one is around the smallest traffic values. (Note that the plot is cropped on the right to better focus on the large difference area. The difference between red and blue lines get smaller in higher values.) Exponential distribution was the best fitting distribution among others we tried, lognormal, gamma, and power law.

In a data center traffic it is not realistic to have a very small amount of traffic flowing between every pair of nodes. Therefore, there is a gap between no traffic and the smallest traffic value in the original traffic. This gap does not exist in a randomly generated traffic because it is drawn from a continuous probability distribution. In the original data, smallest non-zero traffic value is around 0.05. We set any randomly generated traffic value lower than this threshold to zero. Resulting randomly generated data can be seen on the plot in orange. Up to the threshold value, cumulative distribution (orange line) is much closer to the original traffic (blue line) than exponential distribution (red line). Orange and red lines are the same for values greater than the threshold.

To generate hotspot traffic, we use total traffic incoming to each node (sum of rows in Fig. 12). Fig. 15 shows distribution of total incoming traffic of the original data. Brown lines show a lognormal distribution fitted to the original data using *lognorm.fit()* function. The result of *lognorm.fit()* is $s = 1.4456$, $loc = 1.1435$, and $scale = 5.0357$. Orange bar plot and cumulative step plot is a sample drawn from this distribution. As seen from the plot, lognormal distribution fits the

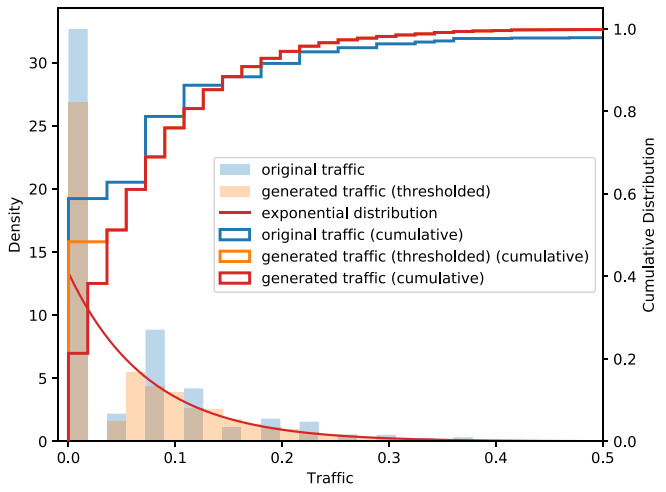

Fig. 14. Pairwise traffic distribution, except hotspots, between nodes. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

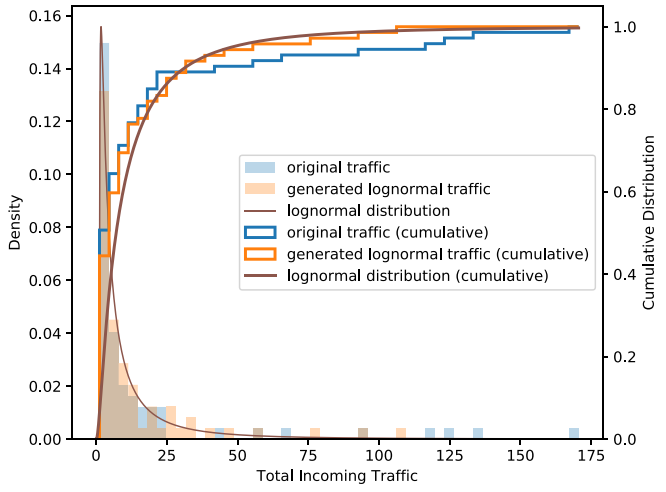

Fig. 15. Total incoming traffic distribution for each node. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

original traffic well, especially at lower values, and fits better than the other distributions we tried, gamma, exponential, and power law.

Details of how individual hotspot traffic is determined is presented in Algorithm 6. Base traffic is generated in Lines 1 – 5. Total incoming traffic for each node is generated in Line 6. Total incoming traffic value is a target that will be reached by adding hotspot traffic over the base traffic.

For each node (Line 7), if target incoming traffic is greater than total incoming base traffic (Line 9), h nodes are selected randomly as hotspots for that node (Line 10). This random selection ensures that hotspots are distributed uniformly over all nodes for outgoing traffic as depicted in Fig. 13 (right). We make sure that hotspot traffic cannot be self-traffic in Line 11. We choose the amount of hotspot traffic from a normal distribution with mean μ' and $\sigma = \mu'/3$ (Lines 12, 13). This hotspot traffic is added to base traffic in Line 14. These last two steps introduce variance in hotspot traffic values to different nodes from this node.

It is possible that target incoming traffic is lower than total incoming base traffic of a node (Line 15). In that case, we clear traffic to randomly selected nodes in Lines 17 – 19, until base traffic is lower than the target incoming traffic.

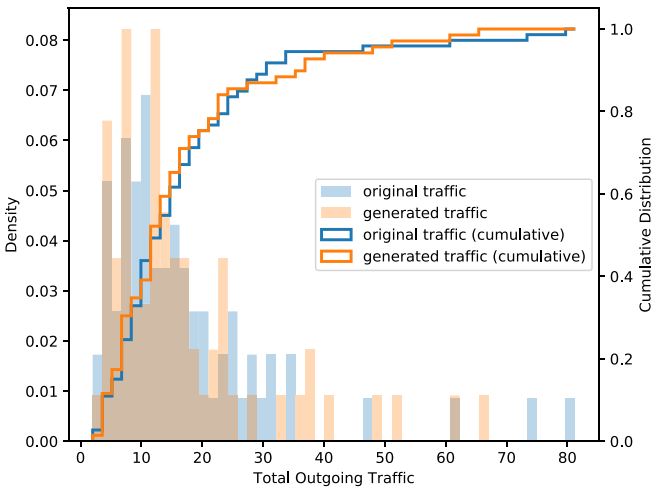

Fig. 16. Comparison of total outgoing traffic distribution.

Algorithm 6 Generate random data center traffic

```
1:  GENERATE-TRAFFIC(n, h, min−traffic, λ, μ, σ)
2:  base−traffic ← EXP-DIST(λ, n × n)
3:  base−traffic[base−traffic < min−traffic] ← 0
4:  for i ∈ {0, 1, ... n − 1} do
5:      base−traffic[i, i] ← 0                          ▷ Clear self-traffic
6:  end for
7:  incoming−traffic ← LOGNORMAL-DIST(μ, σ, n)
8:  for i ∈ {0, 1, ... n − 1} do
9:      base−incoming ← SUM(base−traffic[i])
10:     if base−incoming < incoming−traffic[i] then
11:         hotspots ← UNIFORM-DIST(0, n − 1, h)
12:         hotspots ← hotspots − {i}
13:         μ' ← (incoming−traffic[i] − base−incoming) / h
14:         hotspot−traffic ← NORMAL-DIST(μ', μ'/3, h)
15:         base−traffic[i, hotspots] += hotspot−traffic
16:     else
17:         repeat
18:             rand−node ← UNIFORM-DIST(0, n − 1, 1)
19:             base−traffic[i, rand−node] ← 0
20:         until base−incoming < incoming−traffic[i]
21:     end if
22: end for
23: return CLIP-MIN(base−traffic, 0)
```

5.3. Generated traffic verification

We verify our proposed traffic generation algorithm by comparing the traffic it generates to the original traffic. Fig. 16 compares total outgoing traffic of nodes in a randomly generated traffic and the original traffic. Total outgoing traffic is a good comparison for validation, because it is not sampled from a random distribution in the traffic generation algorithm; rather, it emerges from the whole procedure. Kolmogorov–Smirnov goodness-of-fit test statistic is 0.11 with p-value 0.78, therefore we accept the null hypothesis that the original and generated samples are drawn from the same distribution.

Overall looks of the original and generated traffic could be compared in Fig. 17. Properties of the original traffic that are explained in Section 5.1 could be observed in generated traffic as well. Horizontal

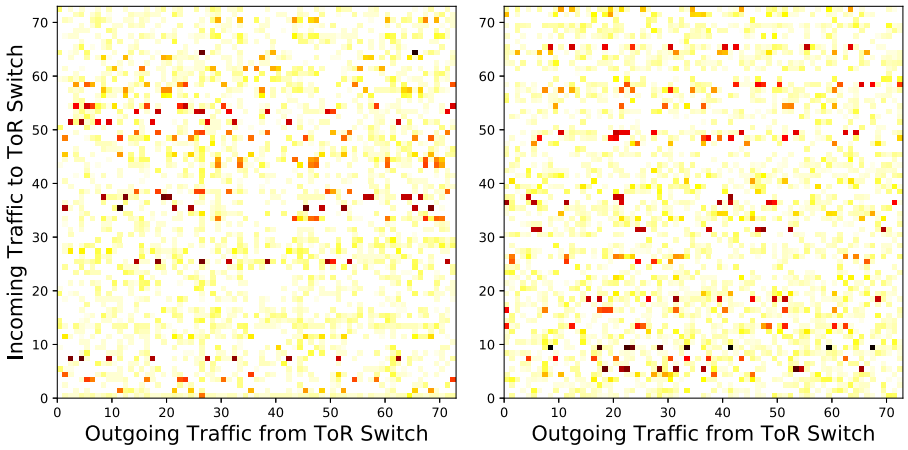

Fig. 17. Comparison of the original (left) and generated traffic (right).

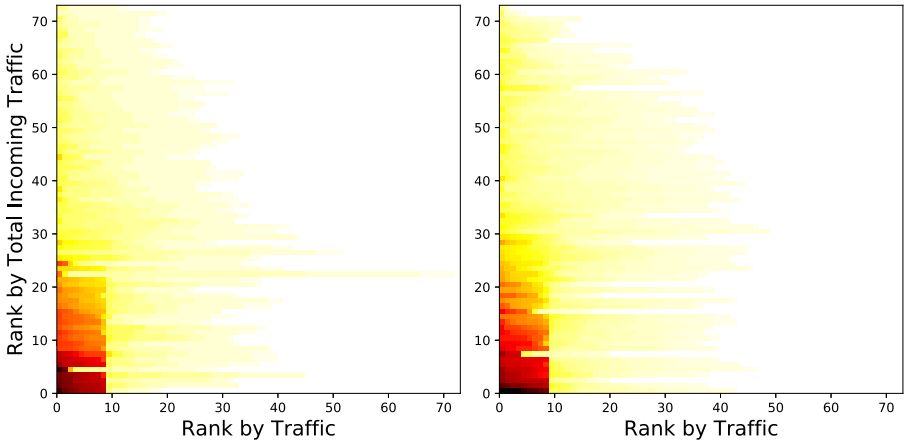

Fig. 18. Comparison of the original (left) and generated traffic (right) sorted by total incoming traffic.

stripe pattern indicates that distribution of incoming traffic is more skewed than outgoing traffic, which is rather uniform among nodes.

Comparison of incoming traffic is given in Fig. 18. Note that generated distribution of hotspots has similar patterns to that of the original traffic. Not every node has exactly h hotspots as in the original traffic, because the amount of hotspot traffic for each node is drawn from a normal distribution. The variation in distribution sometimes yields a very low amount of traffic that looks like base traffic. Another similarity is apparent in low-traffic nodes. In the original data, nodes with the least amount of traffic communicate with fewer other nodes compared to higher-traffic nodes. This pattern is visible in generated data as well, because of the logic in Lines 17 – 19.

Comparison of outgoing traffic is presented in Fig. 19.

Finally, we verify that random traffic generation can scale to different network sizes. When the number of nodes is different than the original (73), we scale all parameters, except min-traffic threshold, in proportion. Characteristic traffic patterns can be seen in different network sizes in Fig. 20.

6. Simulations and results

We developed a custom simulation environment in python. Our simulations are computational, rather than event-based. A simulation takes as input a traffic assignment algorithm and a traffic flow between nodes. It outputs a wireless link configuration and the set of flows each link carries. Reported measurements are calculated from output traces.

Because we do not run an event-based simulation, we neglect effects due to various kinds of delays, wireless link quality, network congestion, etc. We report path length as a proxy for overall delay. Our system model aims to minimize wireless transmission quality concerns. In our simulation model, we assume the availability of wireless links between only direct neighbors which are physically close on the data center floor. Therefore, we expect that the results can be generalized to more realistic scenarios.

Our simulation uses variable base units of measurement, rather than definite units, e.g., time is measured not in seconds but in unit time, traffic is measured not in bytes but in unit traffic. Each wireless link has one unit of bandwidth, defined as units of traffic carried per unit time.

We report results that are averaged over 30 different randomly generated traffic for each network size. In order to compare different network sizes better, we use only square grid networks.

Fig. 21 shows that the mean traffic per source–destination pair is the same over different network sizes. From this result we conclude that the method of scaling traffic generation parameters to different network sizes is successful. Error bars show one standard deviation over the generated traffic patterns. Because smaller networks have fewer number of pairs of nodes (smaller sample size), their standard deviation is higher as a statistical phenomenon.

Fig. 22 shows the amount of allocated traffic for different network sizes. For the smallest two network sizes of 9 and 16 nodes, Shortest-path (SP) and longer-path (LP) algorithms perform closely, with LP version performing 7 and 11% better than the corresponding SP version, respectively. As the network gets larger, LP allocates 24 – 48% more traffic than SP. The increase in allocation is explained by larger networks having an abundance of alternative paths longer than the shortest paths. LP exploits these longer paths to allocate more traffic,

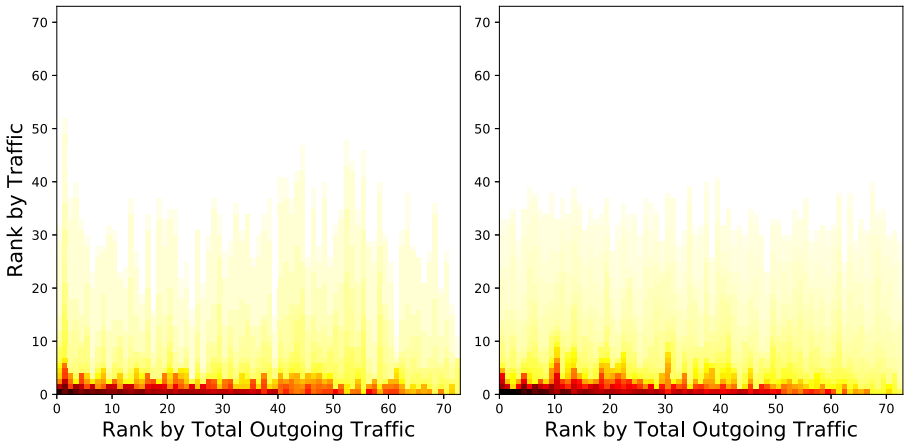

Fig. 19. Comparison of the original (left) and generated traffic (right) sorted by total outgoing traffic.

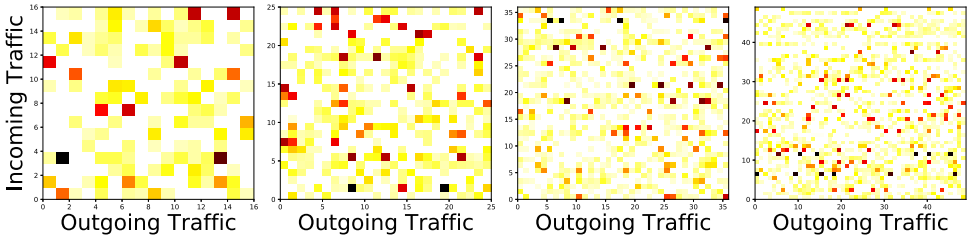

Fig. 20. Randomly generated traffic between ToR switches for network sizes of 16, 25, 36, and 49 nodes.

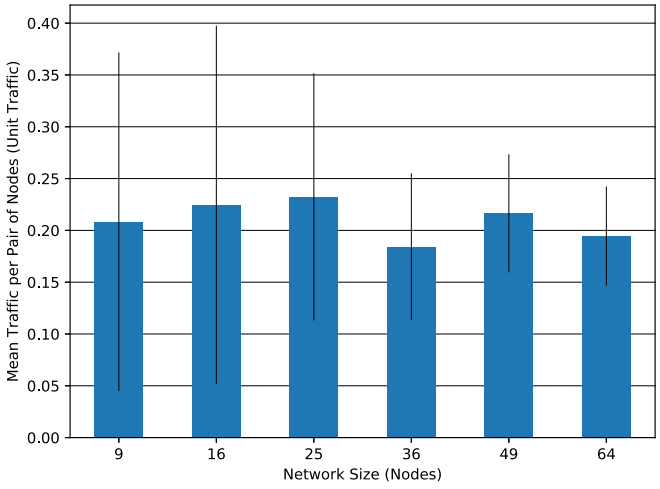

Fig. 21. Mean traffic amount (in unit traffic) for different network sizes.

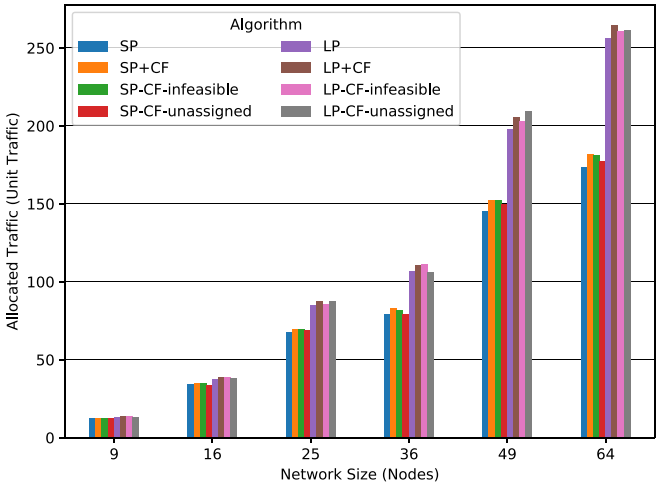

Fig. 22. Allocated traffic.

while SP is restricted to using shortest paths. Lack of alternative routes causes more conflicts between flows, therefore, allocation of less traffic.

Table 2 presents performance improvement of cost-free algorithms over the original ones. CF and CF-infeasible versions perform consistently better (3.22% on the average) than the original versions of SP and LP in all network sizes. Both versions perform similarly, though CF has a slight advantage of 0.51% over CF-infeasible overall. Considering that CF is also simpler to implement and computationally cheaper than CF-infeasible, it is practically the best alternative.

Performance of CF-unassigned has a more complicated profile. Compared to the original version, it performs 1.13% better for SP and 1.72% better for LP, on the average. However, it does not perform better consistently across all network sizes. SP-CF-unassigned performs 0.40% worse than the original SP for 16- and 36-node networks. LP-CF-unassigned performs 0.45% worse than the original LP for 9- and 36-node networks.

Compared to CF and CF-infeasible versions, CF-unassigned performs 1.73% worse on the average. SP-CF-unassigned performs consistently worse (2.47% on the average) than SP+CF and SP-CF-infeasible across different network sizes. Unlike its SP counterpart, LP-CF-unassigned performs better than (a) LP+CF (1.07% on the average) for 25- and 49-node networks and (b) LP-CF-infeasible (1.84% on the average) for 25-, 49-, and 64-node networks.

Compared to shorter routes that SP allocates, longer routes that LP allocates have higher chance of connecting yet-to-be-considered (*unassigned*) source–destination pairs. Therefore, LP version of CF-unassigned is more successful than its SP version.

Table 2
Performance improvement of cost-free versions over the original algorithms.

Network size	Performance improvement over SP			Performance improvement over LP		
	SP+CF	SP-CF-infeasible	SP-CF-unassigned	LP+CF	LP-CF-infeasible	LP-CF-unassigned
9	3.57%	3.59%	0.51%	3.01%	2.53%	−0.27%
16	2.49%	2.55%	−0.75%	2.98%	2.88%	0.60%
25	2.57%	2.42%	1.83%	2.78%	0.84%	2.95%
36	4.95%	3.65%	−0.05%	3.23%	3.81%	−0.64%
49	4.61%	4.87%	2.96%	3.69%	2.56%	5.73%
64	4.81%	4.22%	2.28%	3.05%	1.61%	1.94%
Average	**3.83%**	**3.55%**	**1.13%**	**3.12%**	**2.37%**	**1.72%**

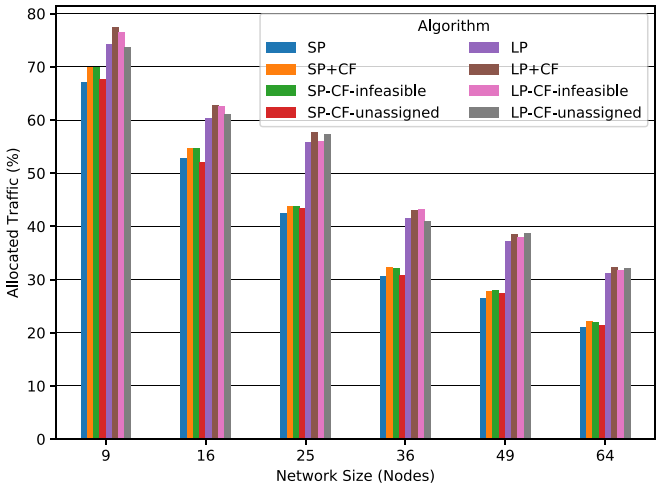

Fig. 23. Allocated traffic as a percentage of total data center traffic.

Because the performance of CF-unassigned is not consistent across different variables, we conclude that it is not a good alternative for reliability purposes. This variance can be attributed to the order in which CF-unassigned allocates flows. The original algorithms and their CF versions allocate flows strictly in the sorted order. CF-infeasible may allocate cost-free flows out of order, but only the infeasible ones. Because flows are added to the set of infeasible flows when they are evaluated for allocation in the sorted order, the set of infeasible flows grow in the sorted order. Therefore, the number and scope of out-of-order allocations are limited. CF-unassigned, however, may allocate cost-free flows in any order. As each allocation affects subsequent allocations, variation in early allocations result in more variation in performance.

As a measure of how much traffic can be offloaded from wired to wireless network, Fig. 23 shows allocated traffic (wireless) as a percentage of total data center traffic (wired and wireless). For the smallest size cloud, wireless network is able to carry a large portion, 65 – 75%, of the traffic. The ratio of allocated traffic falls as the network grows, because both the number of source–destination pairs and the distance between them increase with the network size, causing more conflicts. In the largest network, wireless network is able to carry at least 20% of the traffic.

Fig. 24 shows mean path length of allocated flows. Remember that SP allocates flows to a shortest path between source and destination, whereas LP can use longer paths. The longest path SP can assign is bounded by the diameter of the network, i.e., the distance between most distant two nodes. Length of an LP-allocated path is bounded by the network size minus one, because in the worst case LP may traverse all other nodes. In practice, though, LP-assigned paths are at most half of the network size.

Results show that path length increases quickly with network size for LP algorithms. Path length of SP algorithms remain very small,

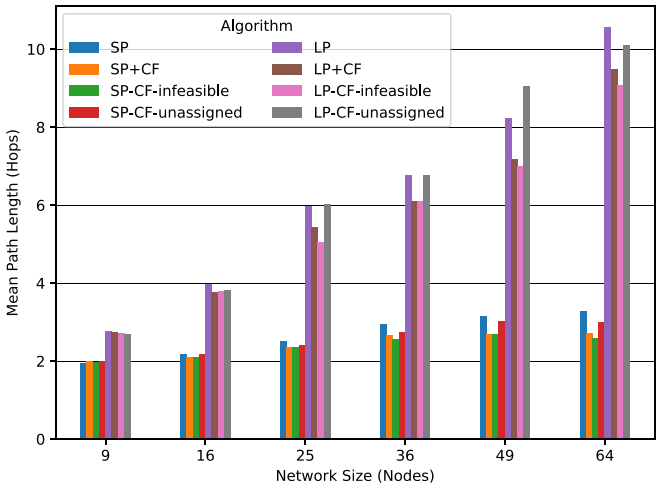

Fig. 24. Mean path length of allocated flows.

increasing by less than 1 hop from 1.97 hops at 9 nodes to 2.88 hops at 64 nodes. Path length of LP algorithms increase by more than 7 hops from 2.72 hops at 9 nodes to 9.81 nodes at 64 nodes. Because the path length determines latency, LP algorithms lose their advantage of allocating more traffic by trading it with higher latency.

Variations of SP have similar path length properties. Contrary to the original SP algorithm, CF versions are not restricted to allocating a flow over a shortest path. A flow can be assigned to a longer path as long as that path connects the flow's source and destination. Despite the fact that CF versions can assign longer paths, in practice assigned path length is at most 2 hops more than the shortest path, because existing paths are all chosen among shortest paths. Mean path length of CF versions (2.45 hops on the average) is even lower than that of the original SP algorithm (2.66 hops on the average), because most of the cost-free allocations are for close source–destination pairs.

Fig. 25 shows completion time of allocated traffic. Before defining completion time of allocated traffic, we explain how completion time of a single flow is calculated.

Completion time of a flow is determined by its share of bandwidth at the bottleneck link along its route. Flows share bandwidth of a bottleneck link proportional to the traffic they carry. Consider that a link is a bottleneck for two flows that carry 2 and 3 units of traffic respectively. The first flow is allocated $2/5$ of, and the second flow is allocated $3/5$ of the bandwidth of the link. If the first flow carries 10 units of traffic, then it will finish in $10/(2/5) = 25$ units of time, because each link has one unit of bandwidth, defined as units of traffic carried per unit time.

Completion time of allocated traffic is the completion time of the latest finishing flow, which is determined by the link that carries the most traffic, i.e., bottleneck of the network.

The traffic allocated by LP algorithms takes 87% longer to finish on the average, because LP algorithms allocate more traffic than SP algorithms. However, the benefit of allocating 35% more traffic on the average is offset by the increased completion time. In order to examine the trade-off between allocated traffic and completion time in more detail, Fig. 26 shows achieved network-wide throughput, calculated by dividing the amount of allocated traffic by its completion time.

The increase in throughput with increased network size shows that algorithms are able to allocate more simultaneous traffic distributed over the network by utilizing local wireless network resources. SP algorithms achieve 35% higher throughput on the average than LP algorithms. The increase in throughput for SP algorithms from 1.62 at 9 nodes to 3.79 at 64 nodes (2.34-fold) is also higher than the increase for LP algorithms from 1.56 at 9 nodes to 2.58 at 64 nodes (1.65-fold). Results show that even though LP algorithms allocate more traffic,

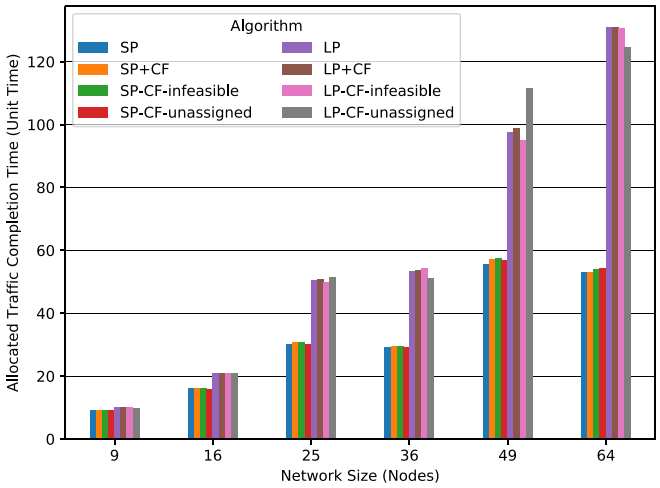

Fig. 25. Completion time of allocated traffic.

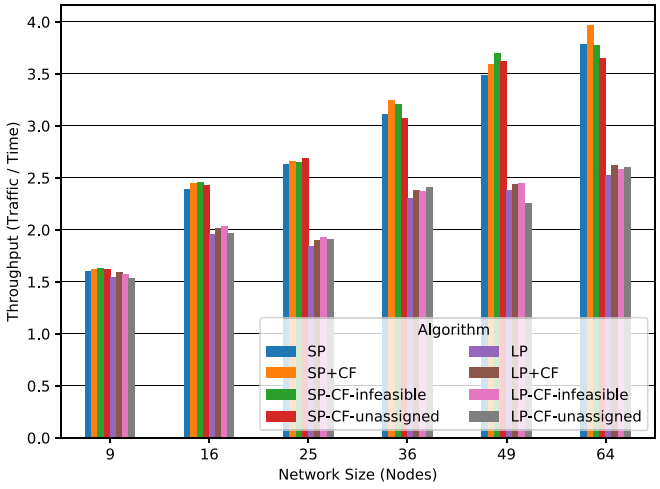

Fig. 26. Effective bandwidth.

they do so at the cost of completion time, compared to SP algorithms. The reason LP algorithms achieve lower throughput is because longer routes cause more flows to share the same link, therefore increasing the number of flows restricted by bottlenecks.

The difference between algorithm variations is also important. Remember that for both SP and LP, CF and CF-infeasible variations allocate 3.22% more traffic on the average than the original versions. They also achieve higher throughput of 2.89% on the average. We conclude that these algorithm versions are preferable over the original versions.

Finally, Fig. 27 shows average bandwidth utilization per wireless link established. As discussed earlier, completion time of a flow is determined by its share of bandwidth at the bottleneck link along its route. There is no benefit for the flow to consume more bandwidth at non-bottleneck links. To the contrary, doing so would increase buffering cost at each hop. Therefore, we consider that a flow consumes the same bandwidth at each link along its route. By definition, bottleneck links are utilized 100%. In an ideal allocation, all wireless links are fully utilized.

Results show that average bandwidth utilization is not affected by the network size. Algorithms are able to utilize expanding wireless networking resources. SP algorithms achieve 77% bandwidth utilization on the average, while LP algorithms achieve 69%. SP algorithms perform better than LP algorithms for the same reason that SP algorithms

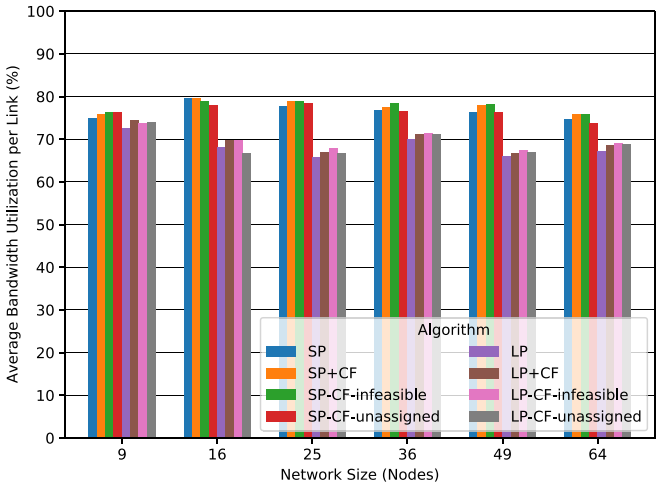

Fig. 27. Average bandwidth utilization per link.

achieve more bandwidth: longer routes allocated by LP algorithms force flows through the same link, therefore reducing bandwidth utilization at non-bottleneck links.

Similar to previous results, CF and CF-infeasible versions are more favorable by performing 1.76% better on the average than the others in terms of bandwidth utilization per link.

7. Conclusion

In this study, we first propose a practical system model for hybrid wireless data centers. Each top-of-the-rack (ToR) switch is equipped with two radios communicating in 60 GHz band using 802.11ad. We show that three non-overlapping channels of 802.11ad (available worldwide) is enough to achieve an interference-free assignment of channels to wireless links. We then propose multi-hop routing algorithms that offload traffic to wireless network under this system model. Our SP family of allocation algorithms route traffic over a shortest-path, hence the name, between source–destination pairs. Our LP family of allocation algorithms are not restricted to a shortest-path; they use longer paths as well. Each family contains an original algorithm and three modified versions. Modified versions run the original algorithm, and also perform a conflict or cost-free allocation at different steps. A cost-free allocation allocates flows for which there is already an established path between their source and destination. Because there are no changes to the wireless link configuration, no other traffic flow is deallocated — there are no costs paid in terms of allocated traffic.

In order to test the performance of our proposed traffic assignment algorithms, we propose a method to randomly generate data center traffic based on a real-world data center traffic pattern. First, we analyze the properties of the real-world traffic pattern. Then, we find probability distributions that fit well onto some of those properties. Finally, we propose a procedure that uses these distributions to randomly generate a traffic pattern. Our proposed traffic generation is able to generate traffic patterns for different network sizes. We verify the results by comparing the properties of the real-world and randomly generated traffic patterns.

We run our proposed multi-hop allocation algorithms on randomly generated traffic and evaluate results in terms of different metrics. LP algorithms allocate 35% more traffic than SP algorithms, but SP algorithms achieve 58% lower latency between source–destination pairs by keeping the routes short. The percentage of allocated traffic to total traffic decreases from 69 and 75% to 22 and 32% with increased network size for SP and LP algorithms respectively. The mean latency increases by only 1 hop for SP, and by 7 hops for LP from smallest to largest network size. SP-allocated traffic finishes in 46% less time

than LP-allocated traffic, but because the amount of allocated traffic is different for each family, we measure throughput to have a fair comparison. SP algorithms achieve 35% higher throughput, i.e., carry more traffic per unit time than LP algorithms. Finally, we measure bandwidth utilization of wireless links. SP algorithms use 77% of the available bandwidth at each link for all network sizes, while LP algorithms use 69%.

Among cost-free versions, CF and CF-infeasible perform better than the original versions in almost every metric and network size. CF and CF-infeasible perform similarly, but CF is simpler in terms of implementation and faster in running time than CF-infeasible, therefore is more practical. CF-unassigned is not preferred for reliability purposes because its performance is inconsistent across different variables.

CRediT authorship contribution statement

Cem Mergenci: Conceptualization, Methodology, Software, Validation, Data curation, Writing - original draft, Writing - review & editing, Visualization, Funding acquisition. **Ibrahim Korpeoglu:** Conceptualization, Writing - review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

Funding: This work was supported by The Scientific and Technological Research Council of Turkey (TÜBİTAK) [grant number 116E048].

References

[1] P. Smulders, Exploiting the 60 GHz band for local wireless multimedia access: prospects and future directions, IEEE Commun. Mag. 40 (1) (2002) 140–147.
[2] K. Ramach, R. Kokku, R. Mahindra, 60 GHz Data-Center Networking: Wireless => Worry less?, NEC Research Paper, 2008.
[3] IEEE, IEEE Standard for information technology–telecommunications and information exchange between systems–local and metropolitan area networks–specific requirements-part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications amendment 3: Enhancements for very high throughput in the 60 GHz band, in: IEEE Std 802.11ad-2012 (Amendment to IEEE Std 802.11-2012, as amended by IEEE Std 802.11ae-2012 and IEEE Std 802.11aa-2012), 2012, pp. 1–628.
[4] T. Nitsche, C. Cordeiro, A.B. Flores, E.W. Knightly, E. Perahia, J.C. Widmer, IEEE 802.11ad: directional 60 ghz communication for multi-gigabit-per-second wi-fi [invited paper], IEEE Commun. Mag. 52 (12) (2014) 132–141.
[5] J.-Y. Shin, E.G. Sirer, H. Weatherspoon, D. Kirovski, On the feasibility of completely wireless datacenters, in: Proceedings of the Eighth ACM/IEEE Symposium on Architectures for Networking and Communications Systems, in: ANCS '12, Association for Computing Machinery, New York, NY, USA, 2012, pp. 3–14, http://dx.doi.org/10.1145/2396556.2396560.
[6] H. Vardhan, R. Prakash, Concurrency in polygonally arranged wireless data centers with all line-of-sight links, in: 2014 International Conference on Computing, Networking and Communications (ICNC), 2014, pp. 716–720, http://dx.doi.org/10.1109/ICCNC.2014.6785424.
[7] Y. Cui, H. Wang, X. Cheng, Channel allocation in wireless data center networks, in: IEEE INFOCOM '11, 2011, pp. 1395–1403.
[8] Y. Cui, H. Wang, X. Cheng, B. Chen, Wireless data center networking, IEEE Wirel. Commun. 18 (6) (2011) 46–53.
[9] S. Kandula, J. Padhye, V. Bahl, Flyways To De-Congest Data Center Networks, Tech. Rep., MSR-TR-2009-109, Microsoft, 2009, URL https://www.microsoft.com/en-us/research/publication/flyways-to-de-congest-data-center-networks/.
[10] Y. Cui, H. Wang, X. Cheng, Wireless link scheduling for data center networks, in: Proceedings of the 5th International Conference on Ubiquitous Information Management and Communication, in: ICUIMC '11, Association for Computing Machinery, New York, NY, USA, 2011, http://dx.doi.org/10.1145/1968613.1968667.
[11] D. Halperin, S. Kandula, J. Padhye, P. Bahl, D. Wetherall, Augmenting data center networks with multi-gigabit wireless links, SIGCOMM Comput. Commun. Rev. 41 (4) (2011) 38–49, http://dx.doi.org/10.1145/2043164.2018442, URL http://doi.acm.org/10.1145/2043164.2018442.
[12] M. Al-Fares, A. Loukissas, A. Vahdat, A scalable, commodity data center network architecture, in: Proceedings of the ACM SIGCOMM 2008 Conference on Data Communication, in: SIGCOMM '08, Association for Computing Machinery, New York, NY, USA, 2008, pp. 63–74, http://dx.doi.org/10.1145/1402958.1402967.
[13] A. Greenberg, J.R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D.A. Maltz, P. Patel, S. Sengupta, VL2: A scalable and flexible data center network, Commun. ACM 54 (3) (2011) 95–104, http://dx.doi.org/10.1145/1897852.1897877.
[14] X. Zhou, Z. Zhang, Y. Zhu, Y. Li, S. Kumar, A. Vahdat, B.Y. Zhao, H. Zheng, Mirror mirror on the ceiling: Flexible wireless links for data centers, in: Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, in: SIGCOMM '12, Association for Computing Machinery, New York, NY, USA, 2012, pp. 443–454, http://dx.doi.org/10.1145/2342356.2342440.
[15] T. Benson, A. Anand, A. Akella, M. Zhang, Understanding data center traffic characteristics, in: Proceedings of the 1st ACM Workshop on Research on Enterprise Networking, in: WREN '09, Association for Computing Machinery, New York, NY, USA, 2009, pp. 65–72, http://dx.doi.org/10.1145/1592681.1592692.
[16] T. Benson, A. Akella, D.A. Maltz, Network traffic characteristics of data centers in the wild, in: Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement, in: IMC '10, Association for Computing Machinery, New York, NY, USA, 2010, pp. 267–280, http://dx.doi.org/10.1145/1879141.1879175.
[17] W. Zhang, X. Zhou, L. Yang, Z. Zhang, B.Y. Zhao, H. Zheng, 3D Beamforming for wireless data centers, in: Proceedings of the 10th ACM Workshop on Hot Topics in Networks, in: HotNets-X, Association for Computing Machinery, New York, NY, USA, 2011, http://dx.doi.org/10.1145/2070562.2070566.
[18] Y. Zhu, X. Zhou, Z. Zhang, L. Zhou, A. Vahdat, B.Y. Zhao, H. Zheng, Cutting the cord: A robust wireless facilities network for data centers, in: Proceedings of the 20th Annual International Conference on Mobile Computing and Networking, in: MobiCom '14, Association for Computing Machinery, New York, NY, USA, 2014, pp. 581–592, http://dx.doi.org/10.1145/2639108.2639140.
[19] M. Casado, M.J. Freedman, J. Pettit, J. Luo, N. McKeown, S. Shenker, Ethane: Taking control of the enterprise, in: Proceedings of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, in: SIGCOMM '07, Association for Computing Machinery, New York, NY, USA, 2007, pp. 1–12, http://dx.doi.org/10.1145/1282380.1282382.
[20] A. Greenberg, G. Hjalmtysson, D.A. Maltz, A. Myers, J. Rexford, G. Xie, H. Yan, J. Zhan, H. Zhang, A clean slate 4D approach to network control and management, SIGCOMM Comput. Commun. Rev. 35 (5) (2005) 41–54, http://dx.doi.org/10.1145/1096536.1096541.
[21] Y. Katayama, T. Yamane, Y. Kohda, K. Takano, D. Nakano, N. Ohba, MIMO Link design strategy for wireless data center applications, in: 2012 IEEE Wireless Communications and Networking Conference (WCNC), 2012, pp. 3302–3306.
[22] T. Yamane, Y. Katayama, An effective initialization of interference cancellation algorithms for distributed mimo systems in wireless datacenters, in: 2012 IEEE Global Communications Conference (GLOBECOM), 2012, pp. 4249–4254.
[23] Y.-J. Yu, C.-C. Chuang, H.-P. Lin, A.-C. Pang, Efficient multicast delivery for wireless data center networks, in: Proceedings - Conference on Local Computer Networks, LCN, 2013, pp. 228–235, http://dx.doi.org/10.1109/LCN.2013.6761238, URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-84898033934&doi=10.1109%2fLCN.2013.6761238&partnerID=40&md5=c9c87aff68a6fd588d01ca236e6cd372.
[24] L. Zhu, J. Wu, G. Jiang, L. Chen, S.-K. Lam, Efficient hybrid multicast approach in wireless data center network, Future Gener. Comput. Syst. 83 (2018) 27–36, http://dx.doi.org/10.1016/j.future.2018.01.012, URL http://www.sciencedirect.com/science/article/pii/S0167739X17311160.
[25] E. Baccour, S. Foufou, R. Hamila, M. Hamdi, A survey of wireless data center networks, in: 2015 49th Annual Conference on Information Sciences and Systems (CISS), 2015, pp. 1–6.
[26] A. Hamza, J.S. Deogun, D.R. Alexander, Wireless communication in data centers: A survey, IEEE Commun. Surv. Tutor. 18 (3) (2016) 1572–1595.
[27] V. Vizing, On an estimate of the chromatic class of a p-graph, Diskret Anal. (3) (1964) 23–30.
[28] I. Holyer, The NP-completeness of edge-coloring, SIAM J. Comput. 10 (4) (1981) 718–720, http://dx.doi.org/10.1137/0210055, arXiv:https://doi.org/10.1137/0210055.
[29] J. Dean, S. Ghemawat, Mapreduce: Simplified data processing on large clusters, Commun. ACM 51 (1) (2008) 107–113, http://dx.doi.org/10.1145/1327452.1327492.
[30] NumFOCUS Foundation, Scipy: Scientific computing tools for python, 2001, https://www.scipy.org/ (Online, accessed 04 May 2020).