Routing in Data-Centers: Theoretical Limits and Algorithms

by

Miguel Alves Ferreira
maferrei@andrew.cmu.edu

A DISSERTATION PROPOSAL
Submitted in partial fulfillment of
the requirements for the degree of

DOCTOR OF PHILOSOPHY
Electrical and Computer Engineering
Carnegie Mellon University

Thesis Committee:
Phillip Gibbons (CMU)
Justine Sherry (CMU)
Rodrigo Rodrigues (IST)
João Luís Sobrinho (IST)
João Xavier (IST)
Michael Dinitz (JHU)

CARNEGIE MELLON UNIVERSITY
Pittsburgh, Pennsylvania
September 2025

Abstract

Clos networks are the building block of modern data-centers. With arbitrarily splittable flows, or sources and destinations restricted to at most one flow, Clos networks have been shown to emulate a macro-switch connecting all sources to all destinations, such that flow rates are limited only by the link capacities at the ingress and egress points. Therefore, Clos networks are an attractive design for a range of use cases, and have been widely deployed. However, today's data-centers operate on different traffic premises: flows are unsplittable, and sources and destinations are not restricted to at most one flow, with congestion control imposing that link capacities are shared among flows according to max-min fair allocations. Given this mismatch, cloud providers rely on heuristics for routing flows in Clos networks. How good are these heuristics? How much can they be improved upon?

This thesis presents the first theoretical investigation of routing in data-centers. In this context, we first argue that no routing algorithm can guarantee that Clos networks emulate a macro-switch. Then, we argue that new routing algorithms operating on additional degrees of freedom available in the technological forefront of data-centers can support a macro-switch abstraction. We divide this argument into three parts, the first two of which have been completed:

- First, assuming unsplittable flows, we study the discrepancies between Clos networks and a macro-switch with congestion control. In our main result, we prove that, when optimizing routing for max-min fairness, there are sets of flows for which the network rates of some flows fall short of their macro-switch rates by a factor of N, where N is the number of middle switches in the network.
- Second, maintaining unsplittable flows, we waive congestion control and investigate again these discrepancies. In our main result, we establish tight bounds on the scale-down factor by which Clos networks approximate a macro-switch. On the one hand, we show that there are sets of flows for which the network rates of some flows fall short of their macro-switch rates by a factor of $3/2$. On the other hand, we give a new algorithm that improves upon know heuristics to ensure that for all sets of flows no flow falls short of its macro-switch rate by a factor greater than $9/5$.
- Third, we plan to design new routing algorithms that make use of the possibility of *either* placing the endpoints of flows in arbitrary servers *or* splitting each flow over a small number of paths to ensure that for all sets of flows the network rate of every flow nearly matches its macro-switch rate.

Overall, this thesis calls into question several common assumptions about the design options for data-centers, and fosters the design of data-centers from algorithmic principles.

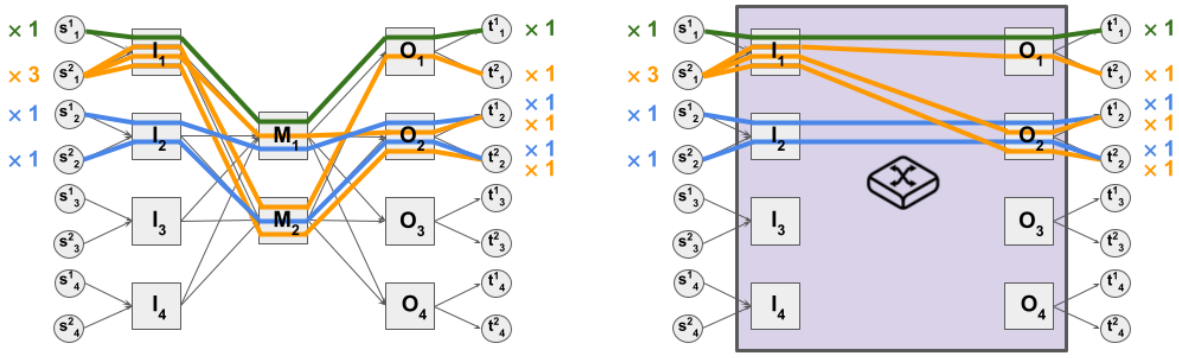
1 Introduction

Hyper-scale data-centers are today's primary computing infrastructure. These infrastructures are operated by large cloud providers, like Google, Microsoft, and Amazon, and co-locate a massive number of hosts, or *servers*, to support a wide range of applications, including search engines, streaming services, social media platforms, and machine learning training. A large cloud provider has typically dozens of data-centers, with each data-center hosting hundreds of thousands of servers [1]. In the United States, data-centers are currently estimated to account for roughly 5% of the electrical power demand, 0.2% of the gross domestic product, and 3 times more traffic than the entire Internet [2].

At a high level of abstraction, a data-center consists of a set of servers interconnected by a network of switches. Applications are distributed over multiple servers, and network connections, or *flows*, ensure the transmission of data-packets between servers. Critically, application performance depends on the rates at which flows transmit data-packets between servers; in fact, for most applications, the network is the main bottleneck for application performance [3–5]. Therefore, an important task for cloud providers is to decide how to *route* flows through the data-center, that is, assign each flow to a path between its servers.

Typically, flows have a desired rate in a data-center, or *demand*, limited by the capacities of links leaving and entering servers, and the routing of the flows seeks to ensure that the network emulates a single switch connecting all servers, or a *macro-switch* [4,6–9] (see Figure 1b.) A data-center network emulates a macro-switch if the aggregate demand routed on any internal link is at most the link capacity, in which case every flow upholds its demand [7, 10–12]. This way, cloud providers aim to abstract away the network from a performance modeling standpoint while guaranteeing predictable rates to applications [7, 9].

Since it is impossible to build a macro-switch, most of today's data-centers are architected after (folded) Clos networks [4, 5, 13–16]. In a *Clos network*, each source (destination) server is linked to a single input (output) top-of-rack (ToR) switch, and each ToR switch is linked to multiple middle switches, as many as there are servers per ToR switch [17] (see Figure 1a). Links have uniform capacities. A key property of Clos networks is that they have *full bisection bandwidth*, which means that the capacity of any cut separating all sources from all destinations is at least the aggregate capacity over all links between sources and input switches (destinations and output switches) [18,19]. This property implies that, under the following traffic premises, Clos networks emulate a macro-switch, and justifies their widespread adoption in range of use cases, including telephone networks and data-centers:



(a) *A Clos network.* ***(b)*** *The macro-switch.*

Figure 1: *A set of flows in a Clos network and in a macro-switch. Circles symbolize servers, and squares symbolize switches. The large colored box symbolizes the single switch. Lines symbolize flows from source to destination, with numbers next to sources and destinations (e.g., ×3) indicating the number of flows leaving those sources or entering those destinations.*

- **Splittable flows:** When flows are *arbitrarily splittable*, meaning that each flow can be routed concurrently over all available paths (as in classic network flow problems), then uniformly dividing the demand of each flow over all source-destination paths satisfies all internal link capacities [20].
- **Single flow per source and destination:** When at most a single input flow per source and per destination is accepted (which can be accomplished with *admission control*, as in early telephone networks, the original use case for Clos networks), then a link-disjoint routing of the flows exists and satisfies all internal link capacities [21]. Furthermore, if these flows are transmitted at link capacity, then the aggregate rate over all flows, or *throughput*, is maximized [22].

However, these are not the traffic premises under which today's data-centers operate. Instead:

- **Unsplittable flows:** Flows are *unsplittable*, for arbitrary flow splitting has significant deployability challenges. While there are several proposals implementing arbitrary flow splitting, including flow-cell [23] and packet-spraying [24], none of them has been adopted in practice, since they require extensive modifications to transport protocols and remain unverified in large-scale [14–16]. (In contrast, *constrained flow splitting* is likely to be deployed in the near future, and is consider in our future work [25, 26].)
- **Multiple flows per source and destination:** All input flows must be accepted, possibly multiple flows per source and per destination. In this context, cloud providers rely on *congestion control* to share the link capacities among the flows, with congestion control imposing a max-min fair allocation at each routing [27]; a *max-min fair allocation* maximizes in lexicographic order the vector whose components are the flow rates sorted from lowest to highest rate [28].

Given this mismatch between the traffic premises of modern data-centers and those of earlier problems, cloud providers cannot simply apply the routing algorithms used in network flows problems and early telephone networks, and rather rely on heuristics for routing flows in Clos networks. In practice, these heuristics perform reasonably well, in the sense that the rates of most flows are close to their demands [14, 25]. But how do cloud providers know if their heuristics are satisfactory for all sets of flows, or if they should look for new heuristics? In other words, what is an optimal heuristic, and how closely does it approximate a macro-switch abstraction?

1.1 Thesis objectives

In this thesis, we present the first formal investigation of routing in data-centers. Our investigation is predicated on the following objectives:

- Characterize how Clos-based data-center networks subject to the realistic constraints of congestion control and unsplittable flows fails to provide a macro-switch abstraction to applications, considering routing in the network as the main design variable.
- Design new routing algorithms that close this gap by operating on additional degrees of freedom available in the technological forefront of data-centers.

This thesis is divided into three parts addressing these objectives. Overall, our goal is to derive bounds on the scale-down factor by which Clos networks approximates a macro-switch, as routing is coupled with further degrees of freedom. The *scale-down factor* for a set of flows denotes the maximum factor by which the network rate of some flow falls short of its macro-switch rate.

• **Part 1: Impossibility Results for Data-Center Routing with Congestion Control and Unsplittable Flows.** In the first part of the thesis, assuming unsplittable flows, we analyze the effect of

congestion control on the discrepancies between Clos networks and a macro-switch. Under congestion control, cloud providers seek to replicate in a Clos network the max-min fair rates in its macro-switch and, with that purpose, they optimize the routing in the network for max-min fairness []. Our key finding proves that there are set of flows for which the max-min fair rates in the macro-switch cannot be replicated; in fact, it proves that, surprisingly, optimizing the routing for max-min fairness reduces the max-min fair rates in the macro-switch of some flows by a factor of N, where N is the number of middle switches. In other words, routing for max-min fairness leads to a worse-case scale-down factor of N, and fails to closely replicate a macro-switch abstraction for fairness.

• **Part 2: Minimum Congestion Routing of Unsplittable Flows in Data-Centers.** In the second part of the thesis, maintaining unsplittable flows, we waive congestion control, and analyze again the discrepancies between Clos networks and a macro-switch. Since data-centers are proprietary, cloud providers are not bound to congestion control, and, instead, they may support arbitrary rates at each routing [29,30]. In this context, our key finding shows that the worst-case scale-down factor improves to between $3/2$ and $9/5$. Notably, showing that this factor is at most $9/5$ entails designing a new routing algorithm breaks through the factor of 2 obtained with existing heuristics [6,31]. These bounds hold for arbitrary demands, not just max-min fair rates in the macro-switch. The main takeaway is that no routing algorithm can uphold arbitrary macro-switch rates under the restriction of a single path per flow.

• **Part 3: Towards Congestion-Free Data-Center Networks.** The previous parts reveals that, if Clos networks wish to emulate a macro-switch, then further design variables beyond single-path routing and arbitrary rate allocation at each routing are required. In the third part of the thesis, which is under development, we are investigating two possible extra design variables. The first is *variable virtual machine placement*. Most applications are hosted in virtual machines, and cloud providers can choose in which servers to place each virtual machine [7–9,32]. Accordingly, an idea is to jointly assign these virtual machines to servers and route the flows between them. The second is *multi-path routing over a few paths*. While arbitrary flow splitting is unrealistic, future transport protocols may tolerate a limited degree of splitability [25,26]. Therefore, another idea is to jointly route flows and divide their demands over a small number of paths. We hope that success in showing that at least one of these extra design variables yields a worst-case scale down factor of nearly 1 leads to the conclusion of the thesis.

In summary, this thesis aims to support the following statement:

Thesis statement: *In today's data-centers, routing in Clos networks falls short of providing a macro-switch abstraction to applications due to restrictions imposed by congestion control and unsplittable flows. However, coupling routing with a variable placement of virtual machines, or a measured form of multi-path routing, allows to bypass these restrictions and reclaim this important abstraction.*

1.2 Roadmap

The remainder of this prospectus presents an overview of the results obtained concerning the gap between Clos networks and a macro-switch in the context of modern data-centers, and of the preliminary ideas for closing this gap by the introduction of new degrees of freedom. Assuming unsplittable flows, in §2 and §3 we summarize, respectively, our results with and without congestion control. In §4, we discuss preliminary ideas for joint virtual-machine placement and routing, and for multi-path routing with a few paths. In §5, we propose a timeline until the thesis defense.

2 Impossibility Results for Data-Center Routing with Congestion Control and Unsplittable Flows

In this section, we overview our impossibility results concerning the discrepancy between a Clos network and a macro-switch subject to congestion control and unsplittable flows in terms of throughput and fairness, which are the two key performance metrics in data-center networking. In §2.1, we discuss the impossibility of maximizing throughput assuming that Clos networks can emulate macro-switches, in §2.2, the impossibility of Clos networks emulating macro-switches when routing for max-min fairness, and, in §2.3, the impossibility of Clos networks emulating macro-switches when routing for throughput. A detailed account of these results can be found in a paper accepted for publication at PODC 2024 [22].

2.1 Imposing max-min fairness up to halves throughput

Cloud providers seek to uphold in a Clos network the rates obtained if congestion control acted only on links leaving and entering servers, that is, the rates in a macro-switch subject to congestion control [6]. Our first result concerns throughput loss due to congestion control assuming that Clos networks can emulate a macro-switch. With admission control, there is at most a single input flow per source and per destination, and flows are transmitted at link capacities, such that the throughput across a macro-switch is maximized. In contrast, with congestion control, there are possibly multiple input flows per source and per destination, and flows are transmitted at max-min fair rates. Therefore, we ask:

Q2.1 *What is the throughput loss, if any, by the max-min fair allocation in the macro-switch relative to the maximum throughput across the switch?*

While it is well-known that a max-min fair allocation does not maximize throughput [22], the simplicity of the macro-switch suggests that this throughput loss should be small. In contrast, we prove that imposing max-min fairness up to halves the maximum throughput, meaning that congestion control introduces a significant throughput loss.

R2.1 *For all sets of flows, the throughput of the max-min fair allocation in a macro-switch is at least half the maximum throughput across the switch. This bound is tight.*

The proof of the bound follows from the known characterization of a maximum throughput allocation in a macro-switch in terms of a bipartite maximum matching [22], and of a max-min fair allocation in terms of flow bottlenecks [28]. More interestingly, the adversarial flows behind its tightness correspond to the *cross gadget* shown in Figure 2. In the maximum throughput allocation, both orange flows are transmitted at link capacity while all blue flows are zeroed. In contrast, in the max-min fair allocation, the capacity of the each link traversed by an orange and all blue flows is shared evenly between that orange and all blue flows; hence, the blue flows as a whole are (effectively) transmitted at link capacity while both orange flows are zeroed despite being unrestricted elsewhere in the network, thus halving the throughput.

2.2 Routing for max-min fairness reduces some macro-switch rates by factor of N

Our second and main result concerns replicating in Clos networks the max-min fair rates in a macro-switch when routing for max-min fairness. While in a macro-switch there is a unique max-min fair allocation for each set of flows, since there is single routing choice, in a Clos network there are many different max-min fair allocations for each set, since there are multiple routings choices. In fact, there may be a different allocation for each routing, since changing the path assigned to some flow may affect the rates of all other flows, even the rates of those that do not share common links with that flow.

(a) *In the maximum throughput allocation, both orange flows have rate* 1*, and all blue flows have rate* 0*, leading to throughput* 2*.*

(b) *In the max-min fair allocation, all flows have rate* $1/k+1$*, leading to throughput* $1 + 1/(k+1)$*, which approaches* 1 *as* k *grows large.*

Figure 2: *The cross gadget underlying the impossibility of maximizing throughput in a macro-switch. In the cross gadget, there are two orange flows, and* k *blue flows, for some large positive integer* k*.*

Cloud providers seek to replicate the max-min fair rates in the macro-switch by (tacitly) routing for max-min fairness, that is, by finding the fairest allocation in the Clos network (in a max-min sense). The fairest allocation in a Clos network is called a *lex-max-min fair allocation*, and maximizes in lexicographic order over all routings the sorted vectors corresponding to max-min fair allocations at each routing [33].

Q2.2 *Does a lex-max-min fair allocation in a Clos network match the max-min fair allocation in a macro-switch for all sets of flows? If not, how close are they?*

In general networks, there is an approximation algorithm that for each set of flows having a common source and possibly different destinations returns a feasible allocation that is approximately max-min fair, and assigns each flow at least a $1/2$-fraction of its lex max-min fair rate [33]. In Clos networks, it is known that there are macro-switch rates (which are not max-min fair) that cannot be replicated in the network [34]. However, no prior work has analyzed the closeness between the lex-max-min fair rates in a Clos network and the max-min fair rates in its macro-switch.

Since both the lex-max-min fair rates and the max-min fair rates in a macro-switch optimize rates in lexicographic order, we would expect them to be close. We show that there are max-min fair rates in a macro-switch that cannot be replicated in a Clos network. In fact, we show that there are lex-max-min fair rates that are significantly worse than the max-min fair rates in the macro-switch. That is, routing for max-min fairness strongly fails to approximate a macro-switch abstraction for max-min fairness.

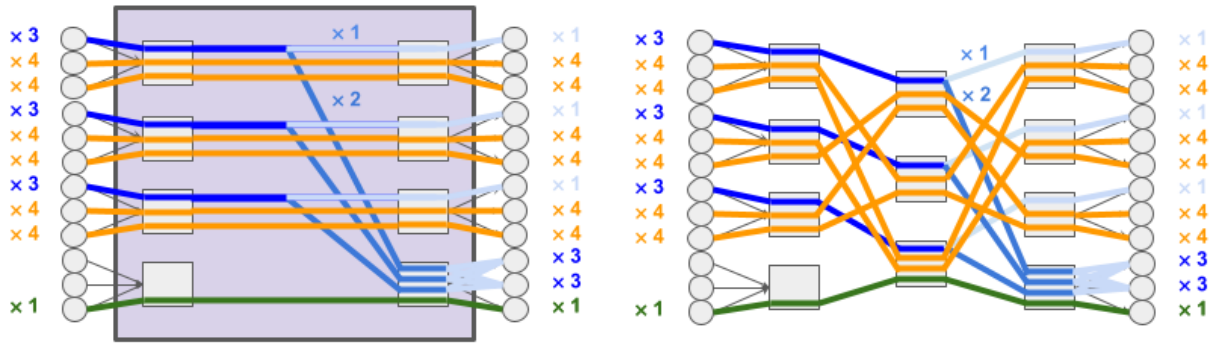
R2.2 *There are sets of flows for which the lex-max-min fair rates of some flows are smaller by a factor of* N *than the max-min fair rates in the macro-switch, where* N *is the number of middle switches.*

The adversarial flows underlying this result are shown in Figure 3a. The key property of this set is that not all flows can replicate their macro-switch rates; hence, lex-max-min fairness prioritizes upholding the lower macro-switch rates of the orange and the blue flows at the expense of decreasing the higher macro-switch rate of the green flow. The crux of the result is to show that all routings that uphold the macro-switch rates of the orange and the blue flows decrease that of the green from 1 to $1/N$. In particular, all such routings reduce to the routing of Figure 3b.

While existing heuristics fail to find a lex-max-min fair allocation, we can show that they also suffer from the same starvation bound posited in the previous result.

2.3 Routing for throughput doubles throughput but zeros most flows

Our third result concerns replicating in Clos networks the max-min fair rates in a macro-switch now when routing for throughput. Since routing for max-min fairness strongly fails to replicate a macro-switch abstraction for max-min fairness, another idea is to replicate it by routing for throughput.



(a) *In the max-min fair allocation in the macro-switch, the orange flows have rate* $1/4$*, the blue flows have rate* $1/3$*, and the green flow has rate* 1*.*

(b) *In a lex-max-min fair allocation in the Clos network, the orange flows have rate* $1/4$*, the blue flows have rate* $1/3$*, but the green flow has rate* $1/4$*.*

Figure 3: *The adversarial flows underlying the impossibility of Clos networks emulating a macro-switch when routing for max-min fairness in the case that there are* 3 *middle switches in the network.*

Q2.3 *If routing optimizes for throughput, then how does the max-min fair allocation in a Clos network compare to that in its macro-switch?*

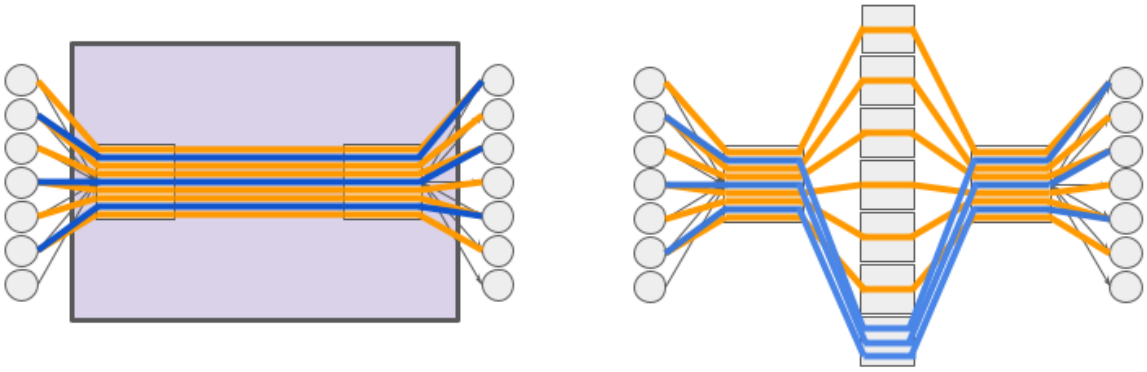
Since congestion control imposes max-min fair rates at each routing, we would expect the maximum throughput over all max-min fair rates in a Clos network to be at most the throughput of the max-min fair rates in its macro-switch, with these maximum throughput rates providing some degree of fairness. However, we establish that routing for throughput up to doubles the throughput relative to the macro-switch, in which case it zeros the max-min fair rates of most flows. Thus, routing for throughput also strongly fails to approximate a macro-switch abstraction for max-min fairness.

R2.3 *For all sets of flows, the maximum throughput over all max-min fair rates in a Clos network is at most twice the throughput of the max-min fair rates in its macro-switch. This bound is tight (as the network grows large), with the max-min fair rates of most flows approaching* 0*.*

The proof of the bound introduces a simple new algorithm for approximating maximum throughput with max-min fair rates at each routing. Of greater significance, the adversarial flows behind its tightness correspond to stacking several copies of the cross gadget in a common ToR switch, as shown in Figure 4. In the macro-switch, for each copy of the gadget, the blue flows as a whole are (effectively) transmitted at link capacity while both orange flows are zeroed, thus yielding a single unit of throughput per gadget. However, in the Clos network, if each of the orange flows is assigned to a different middle switch, and all blue flows are assigned to a common middle switch, then the blue flows cancel each other out, which allows the orange flows to be transmitted at link capacity, thus yielding two units of throughput per gadget.

3 Minimum Congestion Routing of Unsplittable Flows in Data-Centers

While the previous section assumed that congestion control imposes max-min fair rates at each routing, the current section supposes that data-centers support arbitrary rates at each routing. Under this premise, the scale-down factor by which Clos networks approximates a macro-switch equates to the *congestion* of a routing, which is the maximum aggregate demand routed on any link. If the congestion of a routing is at most 1, then the demand of every flow is satisfied; otherwise, some flow falls short of its demand by a factor of at most the congestion, but no greater.



(a) *In the max-min fair allocation in the macro-switch, all flows have rate $1/x$, leading to throughput $1 + 1/(x+1)$ per gadget, which approaches 1 as x grows large.*

(b) *In the max-min fair allocation in the Clos network that maximizes throughput, all blue flows have rate $1/3x$ and all orange flows have rate $2/3$, leading to throughput $5/3$ per gadget, which approaches 2 as the number of middle switches grows large.*

Figure 4: *The stack of cross gadgets underlying the impossibility of Clos networks emulating macro-switches when routing for throughput in the case that there are 7 middle switches in the network.*

With splittable flows, minimizing congestion in a Clos network is easy: for all sets of flows, uniformly splitting the demand of each flow over all source-destination paths yields a minimum congestion routing with congestion at most 1 [20]. However, in today's data-centers, flows are unsplittable. We ask:

Q3.1 *Is the congestion of a minimum congestion routing at most 1 for all sets of flows? If not, how close to 1 is it?*

Q3.2 *Is a minimum congestion routing computable in polynomial-time? If not, how well can it be approximated?*

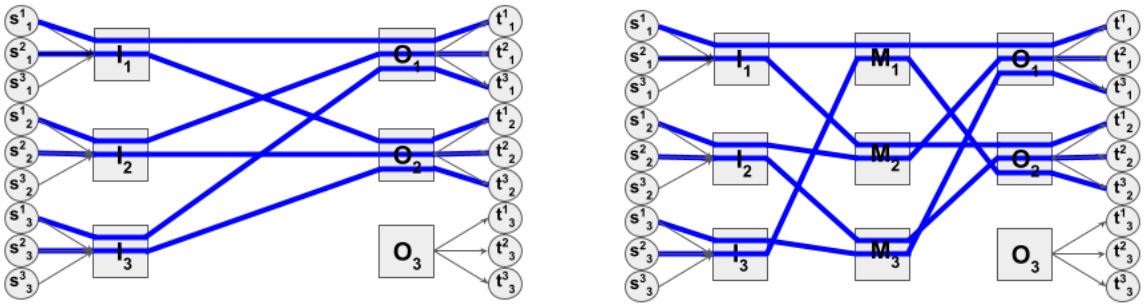
In general networks, there are approximation algorithms that ensure worst-case congestion and approximation-factor logarithmic in the size of the network [35–37]. However, Clos networks have a symmetric structure that allows for better bounds. In Clos networks, there are several heuristics, which offer empirical congestion close to 1, but worst-case congestion distant from 1. The predominant heuristic, random ECMP [13], assigns each flow to a random path, and ensures worst-case congestion and approximation-factor also poly-logarithmic in the number of middle switches. State-of-the-art heuristics [6, 20, 31, 38–40] are based on combinatorial algorithms, and ensure worst-case congestion and approximation-factor of 2. Thus, prior work shows that the worst-case congestion and approximation-factor is between 1 and 2.

In this section, we overview our lower and upper bounds on the existence and efficient computation of minimum congestion routings of unsplittable flows in Clos networks. In §3.1, we discuss improvements to the known lower bounds, in §3.2, a new routing algorithm yielding improvements to the known upper bounds, and, in §3.3, further lower bounds now in an online setting. A detailed account of these results can be found in a paper currently submitted to ITCS 2025, and available on arXiv [41].

3.1 Worst-case congestion and approximation-factor are at least 3/2

Our first result is to establish that, despite the special structure of Clos networks, a minimum congestion routing cannot have worst-case congestion less than $3/2$, nor be approximated by a factor less than $3/2$.

R3.1 *There is a set of flows for which the congestion of a minimum congestion routing is $3/2$.*



(a) *The funnel gadget without routing.* ***(b)*** *A routing for the gadget with congestion* 1.

Figure 5: *The funnel gadget (in blue) underlying the lower bounds in the offline setting in the case that there are* 3 *middle switches in the network. We omit the large colored box representing the macro-switch henceforth.*

R3.2 *It is NP-hard to approximate a minimum congestion routing by a factor strictly less than* 3/2*.*

A basic property of Clos networks is that, if all flows have demand 1 (meaning that there is at most one flow per source and per destination), then for all sets of flows there is a link-disjoint routing of the flows with congestion 1, and one such routing can be found in polynomial-time [21]. In contrast, we show that as soon as flows have demand 1 or $1/2$, the previous property no longer holds. Specifically, we prove that for some set of flows with demand 1 or $1/2$ the minimum congestion is $3/2$. Furthermore, we establish that distinguishing between the sets with congestion at most 1 and those with congestion $3/2$ is NP-complete. The latter implies that no polynomial-time algorithm can approximate a minimum congestion routing by a factor less than $3/2$ unless $P = NP$.

The key idea behind both results is a set of demand 1 flows called a *funnel gadget* and illustrated in Figure 5. The funnel gadget funnels flows from N input switches to $N - 1$ output switches, where N is the number of middle switches; in particular, it maps each of $N - 1$ flows leaving each input switch to a different output switch. Consequently, the funnel gadget has the property that, in a routing with congestion 1, the middle switch to which no flow is assigned is different for each input switch. In the first result, the funnel gadget is used to construct a new set of flows for which no routing with congestion 1 exists by adding demand 1/2 flows from each of the input switches to a extra output switch such that all middle switches are blocked. In the second result, the funnel gadget is used to establish a reduction from the 3-edge coloring problem [42].

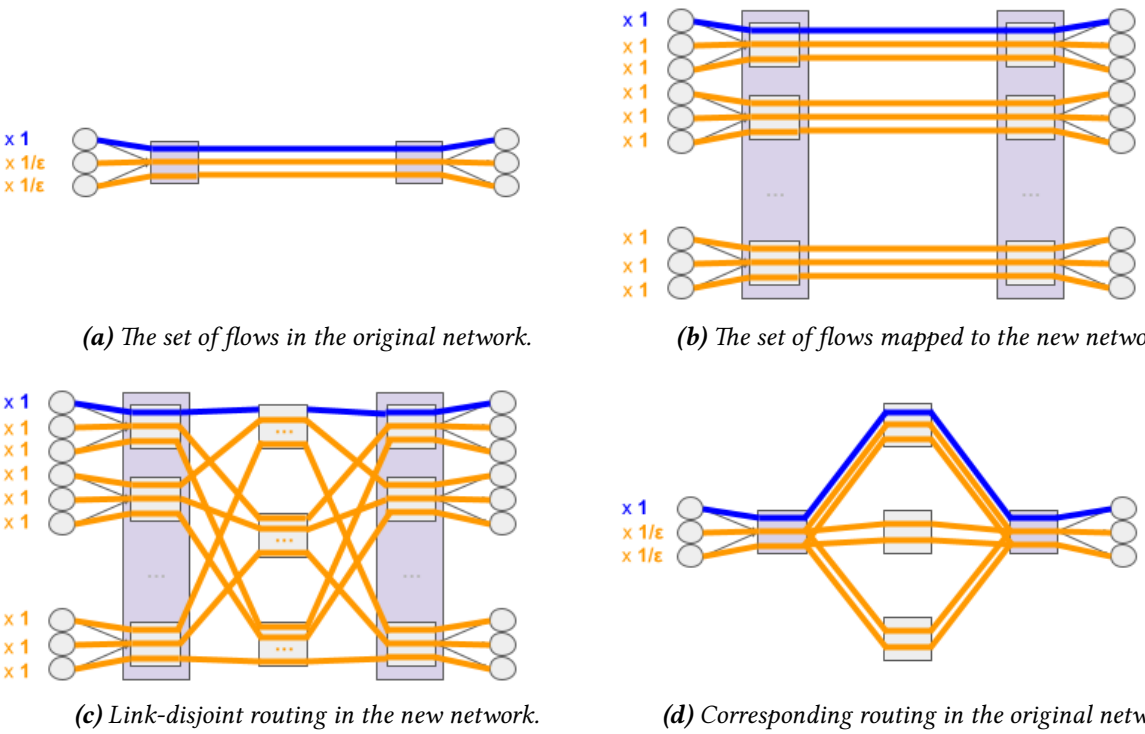
3.2 Worst-case congestion and approximation-factor are at most 9/5

Our second and main result is to design a polynomial-time algorithm that yields worst-case congestion and approximation-factor at most 9/5, thus showing that existing heuristics are not optimal.

R3.3 *There is a polynomial-time algorithm that returns a routing that has congestion at most 9/5 and approximates a minimum congestion routing by a factor at most 9/5 for all sets of flows .*

A natural approach to design such an algorithm would be to analyze the linear program pertaining to the fractional relaxation of the minimum congestion routing problem, as proposed by prior work in the context of general networks. However, in Clos networks, this program has a trivial uniform solution, and thus this approach is not promising. Instead, our algorithm builds upon two known combinatorial algorithms, both of which have worst-case congestion and approximation-factor of 2:

- The *Melen-Turner algorithm* [31] makes use of the basic property that, if there is at most one flow per source and per destination, then there is a link-disjoint routing for the flows. (See Figure 6.) First, the



(a) *The set of flows in the original network.*

(b) *The set of flows mapped to the new network.*

(c) *Link-disjoint routing in the new network.*

(d) *Corresponding routing in the original network.*

Figure 6: *Worst-case flows for the Melen-Turner algorithm. The blue flow has demand* 1*, while the orange flows have demand* ϵ*, for some small positive* ϵ*. The algorithm spreads the low demand orange flows evenly, failing to steer them away from the middle switch traversed by the high demand blue flows.*

algorithm constructs a new Clos network from the original one, and the maps the flows to the new network. In the new network, there are multiple copies of each original ToR switch, with flows assigned to copies of its ToR switches such that there are at most as many flows per copy as there are middle switches and demands decrease with the indexing of the copies. Then, the algorithm finds a link-disjoint routing for the flows in the new network, thus obtaining a routing for them in the original one.

- The *Sorted-Greedy algorithm* [6] sorts the flows in decreasing order of demands, and assigns each flow to a path of minimum congestion; the congestion of a path is the maximum aggregate demand routed on any of its links.

While the pitfall of the Melen-Turner algorithm is that it evenly splits low demand flows over all middle switches without accounting for the presence of high demand flows, the pitfall of the Sorted-Greedy algorithm is that it fails to route high demand flows according to a link-disjoint routing. The essential point of our algorithm is that, by carefully interpolating between the Melen-Turner and the Sorted-Greedy algorithms, it mitigates their pitfalls, and thus arrives at a better bound than they yield individually.

Our algorithm routes a set of flows in two phases bridged by a threshold on the congestion of each phase. In the first phase, a subset of the flows is routed via the Melen-Turner algorithm with congestion at most the threshold. In the second phase, the remaining flows are routed via the Sorted-Greedy algorithm without increasing the congestion beyond the threshold. By setting the threshold to $9/5$, the algorithm returns a routing with congestion at most $9/5$. However, since the minimum congestion may be less than 1, the algorithm fails to approximate a minimum congestion routing by a factor of at most $9/5$. To achieve the latter, a more careful bridging of the two phases is required.

3.3 In online setting, worst-case congestion and approximation-factor are at least 2

In practice, flows arrive one at a time, and cloud providers route each flow before receiving the next flow and without re-routing the previously routed flows. Shifting to this online setting, our third result proves that no online algorithm, even if randomized, has worst-case congestion or approximation-factor less than 2. Thus, the online setting offers strictly more pessimistic guarantees than the offline setting.

R3.4 *No online algorithm (even randomized) approximates minimum congestion by a factor strictly less* 2.

The proof of this result is divided into two parts, the former showing the result for deterministic algorithms, and the latter generalizing it for randomized algorithms. The first part designs two sequences of flows with demand 1 that agree in their prefixes but disagree in their suffixes. The main property of the sequences is that, in a routing with congestion 1, the routing of their common prefix differs; hence, any deterministic algorithm that returns a routing with congestion 1 for one sequence returns a routing with congestion at least 2 for the other. The second part designs exponentially many supersequences, each supersequence consisting of linearly many independent sequences, each of which corresponds to one of the previous sequences. From the main property of the sequences, any deterministic algorithm fails to return a routing with congestion 1 for at least all but one of the sequences; hence, the expected congestion of an optimal deterministic algorithm for the uniform random sequence over the supersequences approaches 2. Then, the application of Yao's Minimax Principle [43, 44] yields the result.

4 Towards Congestion-Free Data-Center Networks

The above section revealed that, even in the absence of congestion control, in order to ensure that every flow satisfies with demand, routing must be coupled with additional degrees of freedom. In this section, we introduce two degrees of freedom: in §4.1, we consider variable placement of virtual machines, and, in §4.2, multi-path routing over a few paths. For each degree of freedom, we discuss a motivating example and mention preliminary ideas for showing that they allow for attaining minimum congestion close to 1.

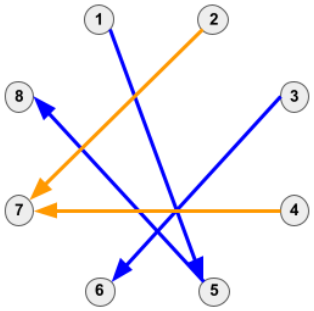
4.1 Joint virtual machine placement and routing

Thus far, we have assumed that the endpoints of flows are physical servers. However, the endpoints of flows are actually virtual machines, which cloud providers can choose in which physical servers to place [7, 8]. With this degree of freedom, flows whose virtual machines are placed in servers linked to a common ToR switch can be routed *internally*, that is, on the path not traversing some middle switch, with the remaining flows routed *externally*, meaning on a path traversed by some middle switch. We ask:

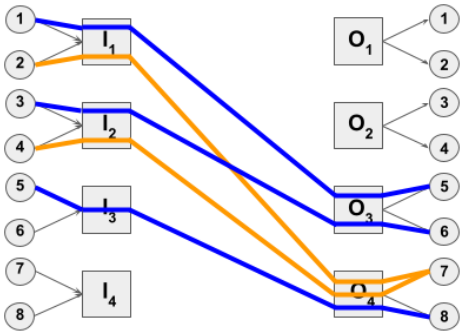
Q4.1 *To what extent does the variable placement of virtual machines allow to decrease the minimum congestion achievable?*

The next example shows that having virtual machine placement as an extra design variable may allow to not only obtain low congestion routings, but doing so using a smaller number of middle switches. Consider the set of flows of Figure 7. With the placement of Figure 7b, the resulting set of flows corresponds to the adversarial flows underlying the lower bound of the minimum congestion achievable introduced in §3.1, for which no routing has congestion less than 3/2. In contrast, with the placement of Figure 7c, there is a routing for the flows with congestion 1, and it uses only a single middle switch.

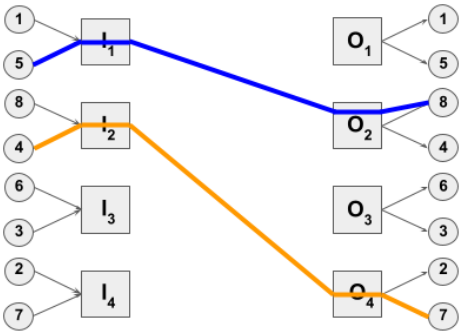
The joint virtual machine placement and routing problem, or the virtual network embedding problem, has received significant attention in the networking community within the deployment of virtualization technologies in the Internet and in data-centers. In general networks, it is known that approximating a



(a) The input flows represented in terms of a directed graph whose vertices are the virtual machines and whose edges are the flows between the respective machines.



(b) A virtual machine placement yielding worst-case congestion. All flows are routed externally.



(c) A virtual machine placement yielding minimum congestion. Only flows routed externally are shown.

Figure 7: An instance of the virtual network embedding problem in Clos networks. We assume that each server can host one virtual machine. Orange flows have demand $1/2$ while blue flows have demand 1.

minimum congestion virtual network embedding with any finite factor is NP-hard [45]. However, there are multiple heuristics for approximating a minimum congestion virtual network embedding based on simulated annealing [46], linear programming rounding [47], and greedy principles [48]. In Clos networks, there are heuristics seeking to minimize the aggregate demand routed between ToR switches based on graph-partitioning algorithms [32]. Furthermore, in the online setting where a sequence of virtual networks arrives one at a time, there are heuristics in the online setting seeking to maximize the aggregate number of accepted virtual networks while satisfying the link capacities based on first-fit algorithms [7].

Despite the existence of this body of work, none offers a guideline to a formal approach to the minimum congestion virtual network embedding problem in Clos networks. We point out a starting point:

- Place the virtual machines in servers in such a way that the flows routed externally can be routed with low congestion by the new routing algorithm introduced in §3.2. While it is possible to route most high demand flows internally, this becomes impossible as flow demands decrease. However, if flows routed externally have low demand, then our algorithm returns a low congestion routing, since its performance depends mainly on the demand of the heaviest flow. Would it help to assume that all strongly connected components in the input graph are small?

4.2 Multi-path routing over a few paths

If the demand of each flow can be split over all available paths, then the minimum congestion is less than 1. While arbitrarily flow splitting seems ultimately impractical, there are several proposals, including MP-TCP [25] and MP-RDMA [26], that already support flow splitting over a few paths. We ask:

Q4.2 *What is the trade-off between the degree of splittability and the minimum congestion achievable? In particular, is it possible to obtain minimum congestion close to* 1 *when splitting each flow over only a constant number of paths?*

Going back to the previous example, it is easy to show that at most two paths per flow suffice to route the adversarial flows of Figure 7b with congestion 1. The basic property of Clos networks asserting the existence of a link-disjoint routing for all sets of flows with at most one flow per server generalizes to assert the existence of a routing where each link is traversed by at most k flows for all sets of flows with at most k flows per server, for any positive integer k. Therefore, by dividing each flow with demand 1 into two flows with demand $1/2$, we obtain a new set of flow with demand $1/2$ for which the previous property guarantees the existence of a routing with congestion 1.

Prior work suggests that a measured form of multi-path routing is a promising idea [25]. Suppose that each flow is routed concurrently on a given number of paths chosen at random, and that flow rates are specified by MP-TCP, which imposes an extension of a max-min fair allocation from single to multi-path routing of the link capacities among the flow rates. In Clos networks with hundreds of middle switches, it has been reported empirically that, while with a single path per flow the average congestion is at least 2, less than ten paths per flow suffice to achieve average congestion is almost 1.

Similarly to the minimum congestion virtual network embedding problem, prior work does not offer a guideline to a formal approach to the minimum congestion routing problem in Clos networks with a few paths per flow. We point out two starting points:

- Analyze a simpler form of the previous randomized algorithm within the balls-and-bins framework [49]. If each flow is divided into a given number of sub-flows with even demand, and each sub-flow is assigned to a random path, then each sub-flow can be cast as a weighted ball and each link as a bin, with each throw placing two copies of a ball in two different bins.
- Investigate the structural properties of the linear program corresponding to the fractional relaxation of the minimum congestion routing problem. While this linear program has trivial uniform solutions, these solutions are not extreme points. Do all extreme points of the linear program correspond to nearly sparse solutions?

5 Tentative Timeline

The table below proposes a timeline for the remaining thesis work. Our main task is to continue the research of the joint virtual machine placement and routing problem, which began last Spring and proceeded throughout this Summer. We have already arrived at a few results showing worst-case congestion of nearly 1 when flows satisfy particular restrictions, but are still far from setting the question in the general case. If the previous problem reveals overly demanding, we consider shifting our research towards the multi-path routing problem with a few paths.

Semester	Plan
Fall 2025	• Continue the investigation of the joint virtual machine placement and routing problem.
Spring 2026	• Work on and submit a paper concerning the joint virtual machine placement and routing problem.
Summer 2026	• Write the thesis.

References

[1] Meta. Reinventing Facebook's Data Center Network, 2019.

[2] McKinsey & Company. The Data Center Balance: How US states an Navigate the Opportunities and Challenges, 2025.

[3] Theophilus Benson, Aditya Akella, and David A Maltz. Network Traffic Characteristics of Data Centers in the Wild. In *Proceedings of the ACM Internet Measurement Conference*, pages 267–280, 2010.

[4] Albert Greenberg, James Hamilton, Navendu Jain, Srikanth Kandula, Changhoon Kim, Parantap Lahiri, David Maltz, Parveen Patel, and Sudipta Sengupta. VL2: A Scalable and Flexible Data Center Network. *ACM SIGCOMM Computer Communication Review*, 39(4):51–62, 2009.

[5] Radhika Niranjan Mysore, Andreas Pamboris, Nathan Farrington, Nelson Huang, Pardis Miri, Sivasankar Radhakrishnan, Vikram Subramanya, and Amin Vahdat. PortLand: A Scalable Fault-Tolerant Layer 2 Data Center Network Fabric. *ACM SIGCOMM Computer Communication Review*, 39(4):39–50, 2009.

[6] Mohammad Al-Fares, Sivasankar Radhakrishnan, Barath Raghavan, Nelson Huang, and Amin Vahdat. Hedera: Dynamic Flow Scheduling for Data Center Networks. In *Proceedings of the USENIX Conference on Networked Systems Design and Implementation*, pages 89–92, 2010.

[7] Hitesh Ballani, Paolo Costa, Thomas Karagiannis, and Ant Rowstron. Towards Predictable Datacenter Networks. *ACM SIGCOMM Computer Communication Review*, 41(4):242–253, 2011.

[8] Hitesh Ballani, Keon Jang, Thomas Karagiannis, Changhoon Kim, Dinan Gunawardena, and Greg O'Shea. Chatty Tenants and the Cloud Network Sharing Problem. In *Proceedings of the USENIX Conference on Networked Systems Design and Implementation*, pages 171–184, 2013.

[9] Jeongkeun Lee, Yoshio Turner, Myungjin Lee, Lucian Popa, Sujata Banerjee, Joon-Myung Kang, and Puneet Sharma. Application-Driven Bandwidth Guarantees in Datacenters. *ACM SIGCOMM Computer Communication Review*, 44(4):467–478, 2014.

[10] Nick Duffield, Pawan Goyal, Albert Greenberg, Partho Mishra, Kadangode Ramakrishnan, and Jacobus van der Merive. A Flexible Model for Resource Management in Virtual Private Networks. *ACM SIGCOMM Computer Communication Review*, 29(4):95–108, 1999.

[11] Mohammad Alizadeh, Shuang Yang, Milad Sharif, Sachin Katti, Nick McKeown, Balaji Prabhakar, and Scott Shenker. pFabric: Minimal Near-Optimal Catacenter Transport. *ACM SIGCOMM Computer Communication Review*, 43(4):435–446, 2013.

[12] Mosharaf Chowdhury, Yuan Zhong, and Ion Stoica. Efficient Coflow Scheduling with Varys. In *Proceedings of the ACM SIGCOMM Conference*, page 443–454, 2014.

[13] Mohammad Al-Fares, Alexander Loukissas, and Amin Vahdat. A Scalable, Commodity Data Center Network Architecture. *ACM SIGCOMM Computer Communication Review*, 38(4):63–74, 2008.

[14] Mubashir Qureshi, Yuchung Cheng, Qianwen Yin, Qiaobin Fu, Gautam Kumar, Masoud Moshref, Junhua Yan, Van Jacobson, David Wetherall, and Abdul Kabbani. PLB: Congestion Signals are Simple and Effective for Network Load Balancing. In *Proceedings of the ACM SIGCOMM Conference*, pages 207–218, 2022.

[15] Kun Qian, Yongqing Xi, Jiamin Cao, Jiaqi Gao, Yichi Xu, Yu Guan, Binzhang Fu, Xuemei Shi, Fangbo Zhu, and Rui Miao. Alibaba HPN: A Data Center Network for Large Language Model Training. In *Proceedings of the ACM SIGCOMM Conference*, pages 691–706, 2024.

[16] Adithya Gangidi, Rui Miao, Shengbao Zheng, Sai Jayesh Bondu, Guilherme Goes, Hany Morsy, Rohit Puri, Mohammad Riftadi, Ashmitha Jeevaraj Shetty, and Jingyi Yang. RDMA Over Ethernet for Distributed Training at Meta Scale. In *Proceedings of the ACM SIGCOMM Conference*, pages 57–70, 2024.

[17] Charles Clos. A Study of Non-Blocking Switching Networks. *The Bell System Technical Journal*, 32(2):406–424, 1953.

[18] Sangeetha Abdu Jyothi, Ankit Singla, P. Brighten Godfrey, and Alexandra Kolla. Measuring and Understanding Throughput of Network Topologies. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 761–772, 2016.

[19] Pooria Namyar, Sucha Supittayapornpong, Mingyang Zhang, Minlan Yu, and Ramesh Govindan. A Throughput-Centric View of the Performance of Datacenter Topologies. In *Proceedings of the ACM SIGCOMM Conference*, page 349–369, 2021.

[20] Marco Chiesa, Guy Kindler, and Michael Schapira. Traffic Engineering With Equal-Cost-MultiPath: An Algorithmic Perspective. *IEEE/ACM Transactions on Networking*, 25(2):779–792, 2017.

[21] Frank Hwang. Control Algorithms for Rearrangeable Clos Networks. *IEEE Transactions on Communications*, 31(8):952–954, 1983.

[22] Miguel Alves Ferreira, Nirav Atre, Justine Sherry, and João Luís Sobrinho. Impossibility Results for Data-Center Routing with Congestion Control and Unsplittable Flows. In *Proceedings of the ACM Symposium on Principles of Distributed Computing*, pages 358–368, 2024.

[23] Keqiang He, Eric Rozner, Kanak Agarwal, Wes Felter, John Carter, and Aditya Akella. Presto: Edge-Based Load Balancing for Fast Datacenter Networks. *ACM SIGCOMM Computer Communication Review*, 45(4):465–478, 2015.

[24] Advait Dixit, Pawan Prakash, Yu Hu, and Ramana Rao Kompella. On the Impact of Packet Spraying in Data Center Networks. In *Proceedings of the IEEE International Conference on Computer Communications*, pages 2130–2138, 2013.

[25] Costin Raiciu, Sebastien Barre, Christopher Pluntke, Adam Greenhalgh, Damon Wischik, and Mark Handley. Improving Datacenter Performance and Robustness with Multipath TCP. *ACM SIGCOMM Computer Communication Review*, 41(4):266–277, 2011.

[26] Yuanwei Lu, Guo Chen, Bojie Li, Kun Tan, Yongqiang Xiong, Peng Cheng, Jiansong Zhang, Enhong Chen, and Thomas Moscibroda. Multi-Path Transport for RDMA in Datacenters. In *Proceedings of the USENIX Conference on Networked Systems Design and Implementation*, pages 357–371, 2018.

[27] Chuanxiong Guo, Haitao Wu, Zhong Deng, Gaurav Soni, Jianxi Ye, Jitu Padhye, and Marina Lipshteyn. RDMA over Commodity Ethernet at Scale. In *Proceedings of the ACM SIGCOMM Conference*, page 202–215, 2016.

[28] Dimitri Bertsekas and Robert Gallager. *Data Networks, 2nd Edition.* Prentice-Hall, Inc., 1992.

[29] Jonathan Perry, Amy Ousterhout, Hari Balakrishnan, Devavrat Shah, and Hans Fugal. Fastpass: A Centralized "Zero-Queue" Datacenter Network. In *Proceedings of the ACM SIGCOMM Conference*, pages 307–318, 2014.

[30] Saksham Agarwal, Qizhe Cai, Rachit Agarwal, David Shmoys, and Amin Vahdat. Harmony: A Congestion-Free Datacenter Architecture. In *Proceedings of the USENIX Conference on Networked Systems Design and Implementation*, pages 329–343, 2024.

[31] Riccardo Melen and Jonathan Turner. Nonblocking Multirate Networks. *SIAM Journal on Computing*, 18(2):301–313, 1989.

[32] Xiaoqiao Meng, Vasileios Pappas, and Li Zhang. Improving the Scalability of Data Center Networks with Traffic-Aware Virtual Machine Placement. In *Proceedings of the IEEE INFOCOM Conference*, pages 1–9, 2010.

[33] Jon Kleinberg, Eva Tardos, and Yuval Rabani. Fairness in Routing and Load Balancing. In *Proceedings of the IEEE Symposium on Foundations of Computer Science*, pages 568–578, 1999.

[34] Shun-Ping Chung and Keith Ross. On Nonblocking Multirate Interconnection Networks. *SIAM Journal on Computing*, 20(4):726–736, 1991.

[35] Prabhakar Raghavan and Clark Tompson. Randomized Rounding: A Technique for Provably Good Algorithms and Algorithmic Proofs. *Combinatorica*, 7(4):365–374, 1987.

[36] Amit Chakrabarti, Chandra Chekuri, Anupam Gupta, and Amit Kumar. Approximation Algorithms for the Unsplittable Flow Problem. *Algorithmica*, 47(1):53–78, 2007.

[37] Julia Chuzhoy, Venkatesan Guruswami, Sanjeev Khanna, and Kunal Talwar. Hardness of Routing with Congestion in Directed Graphs. In *Proceedings of the ACM Symposium on Theory of Computing*, page 165–178, 2007.

[38] Andrew Curtis, Jeffrey Mogul, Jean Tourrilhes, Praveen Yalagandula, Puneet Sharma, and Sujata Banerjee. DevoFlow: scaling flow management for high-performance networks. *ACM SIGCOMM Computer Communication Review*, 41(4):254–265, 2011.

[39] Mohammad Alizadeh, Tom Edsall, Sarang Dharmapurikar, Ramanan Vaidyanathan, Kevin Chu, Andy Fingerhut, Vinh The Lam, Francis Matus, Rong Pan, Navindra Yadav, and George Varghese. CONGA: Distributed Congestion-Aware Load Balancing for Data Centers. *ACM SIGCOMM Computer Communication Review*, 44(4):503–514, 2014.

[40] Kuo-Feng Hsu, Ryan Beckett, Ang Chen, Jennifer Rexford, and David Walker. Contra: A Programmable System for Performance-Aware Routing. In *Proceedings of the USENIX Symposium on Networked Systems Design and Implementation*, pages 701–721, 2020.

[41] Miguel Ferreira, Nirav Atre, Justine Sherry, Michael Dinitz, and João Luís Sobrinho. Minimum Congestion Routing of Unsplittable Flows in Data-Center Networks. *arXiv preprint arXiv:2505.03908*, 2025.

[42] Richard Karp. Reducibility Among Combinatorial Problems. In *Proceedings of a Symposium on the Complexity of Computer Computations*, page 85–103, 1972.

[43] Andrew Yao. Probabilistic Computations: Towards a Unified Measure of Complexity. In *Proceedings of the IEEE Symposium on Foundations of Computer Science*, pages 222–227, 1977.

[44] Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[45] Matthias Rost and Stefan Schmid. On the Hardness and Inapproximability of Virtual Network Embeddings. *IEEE/ACM Transactions on Networking*, 28(2):791–803, 2020.

[46] Joe Wenjie Jiang, Tian Lan, Sangtae Ha, Minghua Chen, and Mung Chiang. Joint VM Placement and Routing for Data Center Traffic Engineering. In *Proceedings of the IEEE INFOCOM Conference*, pages 2876–2880, 2012.

[47] Mosharaf Chowdhury, Muntasir Rahman, and Raouf Boutaba. Virtual Network Embedding with Coordinated Node and Link Mapping. In *Proceedings of the IEEE INFOCOM Conference*, pages 783–791, 2009.

[48] Yong Zhu and Mostafa Ammar. Algorithms for Assigning Substrate Network Resources to Virtual Network Components. In *Proceedings of the IEEE INFOCOM Conference*, pages 1–12, 2006.

[49] Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomization and Probabilistic Techniques in Algorithms and Data Analysis*. Cambridge University Press, 2017.