On the Complexity of Non-Segregated Routing in Reconfigurable Data Center Architectures

Klaus-Tycho Foerster
University of Vienna, Austria
klaus-tycho.foerster@univie.ac.at

Maciej Pacut
University of Wroclaw, Poland
pct@cs.uni.wroc.pl

Stefan Schmid
University of Vienna, Austria
stefan_schmid@univie.ac.at

ABSTRACT

By enhancing the traditional static network (e.g., based on electric switches) with a dynamic topology (e.g., based on reconfigurable optical switches), emerging reconfigurable data centers introduce unprecedented flexibilities in how networks can be optimized toward the workload they serve. However, such hybrid data centers are currently limited by a restrictive routing policy enforcing artificial *segregation*: each network flow can only use either the static or the flexible topology, but not a combination of the two.

This paper explores the algorithmic problem of supporting more general routing policies, which are not limited by segregation. While the potential benefits of non-segregated routing have been demonstrated in recent work, the underlying algorithmic complexity is not well-understood.

We present a range of novel results on the algorithmic complexity of non-segregated routing. In particular, we show that in certain specific scenarios, optimal data center topologies with non-segregated routing policies can be computed in polynomial-time. In many variants of the problem, however, introducing a more flexible routing comes at a price of complexity: we prove several important variants to be NP-hard.

CCS CONCEPTS

• **Networks → Network architectures**; • **Theory of computation → Design and analysis of algorithms**;

KEYWORDS

Algorithms, Complexity, Routing, Optical Circuit Switches, Free-Space Optics, Reconfigurable Topologies

1 INTRODUCTION

With the increasing popularity of data-centric applications, the design of efficient and cost-effective data center networks has received much attention over the last years. While traditionally, data center topologies are optimized to provide performance guarantees under *arbitrary* workloads (e.g., [2, 17, 18, 25, 30, 38]), emerging *reconfigurable* topologies (e.g., [8, 10, 13, 15, 19, 21, 28, 39, 45]) allow to dynamically adjust the topology, enabling *demand-aware* ("workload-aware"), self-adjusting networks [7]. It has been shown that demand-aware networks can achieve a performance similar to demand-oblivious networks at lower cost [8, 15], depending on the workload.

However, while reconfigurable topologies introduce a new dimension of flexibility to the data center design problem, it typically impossible to fully exploit these flexibilities due to restrictive routing policies. Reconfigurable data center networks are typically *hybrid* and combine two types of topologies: a static topology which consists of electric switches, and a flexible topology which consists of optical (or wireless) switches providing the reconfigurable links. But while the topology is hybrid, routing is not: routing policies enforce an artificial *segregation*. In segregated routing, a network flow can either only use the static topology (e.g., mice flows) or only the flexible topology (e.g., elephant flows), but not a combination of the two; this can lead to a suboptimal resource allocation [14].

This paper is motivated by the desire to unlock the full flexibility of reconfigurable networks by supporting *non-segregated routing*. In particular, we are interested in the algorithmic complexity of supporting such general routing policies as in Fig. 1: essentially a *joint optimization problem*, involving both topology design and routing.

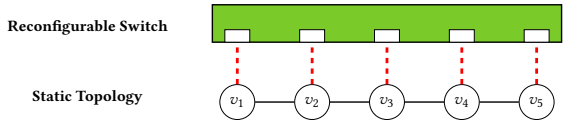

Figure 1: Line topology with five nodes, where each node is connected to a reconfigurable switch (dashed): the choice of the matching inside the switch depends on the current communication demands. For example, if v_1 wants to speed up its connection to v_5, one would create a matching link between both nodes via the reconfigurable switch. However, a problem arises if there is also a demand between *e.g.,* v_4 and v_1, as v_1 can only match to one other node in this example. In the *segregated* case, one flow has to suffer from a poor connection. However, in the *non-segregated* case, v_4 can significantly improve the flow routing to v_1, by first routing to v_5, and then taking the matching shortcut to v_1.

1.1 Contributions

We explore the algorithmic complexity of supporting more general routing policies, which are not limited by segregation. We classify demand-aware routing in reconfigurable networks along two dimensions, (1) the number of connections to the reconfigurable switch per node and (2) the number of allowed alternations between the static reconfigurable network parts. We also investigate the effect of allowing at most one reconfigurable hop per route. A tabular overview of our results is presented in Table 1.

- *Segregated routing:* We first show that when each route is limited to at most one reconfigurable link, then an optimal routing can be found efficiently (§3). However, if we remove this restriction, then already allowing $b = 2$ connections to the reconfigurable switch turns the problem NP-hard (§4.1), as well as for every larger $b \in \mathbb{N}$.
- *Non-segregated routing:* When one can mix reconfigurable and static links, routing is more efficient, but computationally harder to optimize. We show that already allowing $k = 1$ alternations between network parts is NP-hard (§4.3), even if the reconfigurable degree is just $b = 1$. The same hardness results also apply to any larger reconfigurable degree $b \in \mathbb{N}$, see §4.1 and §4.2.

Reconfig. degree $\Delta_R \in \mathbb{N}$	$h_\# \leq 1, k = 0$ **Alternations**	$k = 0$ **Alternations**	$k = 1$	$\forall k > 1, k \in \mathbb{N}$
$\Delta_R = 1$	**P** [14]	**P** [14]	**NPC** (§4.3)	**NPC** [14]
$\Delta_R = 2$	**P** (§3)	**NPC** (§4.1)	**NPC** (§4.2)	**NPC** (§4.1)
$\forall \Delta_R > 2$	**P** (§3)	**NPC** (§4.2)	**NPC** (§4.2)	**NPC** (§4.2)

Table 1: Overview of the complexity of demand-aware routing, depending on the reconfigurable degree b (how many connections to the reconfigurable switch) and the number of allowed alternations k between the static and the reconfigurable topology. Most problem classes are NP-hard to optimize, except when adding the restriction that at most $h_\# = 1$ reconfigurable links may be used on a route—a choice that simplifies calculating the routing, but at the cost of routing efficiency.

While these results are presented for the popular model of connecting one reconfigurable switch (*e.g.,* an optical circuit switch) to the nodes, as we point out in §6, many results transfer to the case of multiple switches. Our results further apply to both unidirectional and bidirectional reconfigurable links.

2 MODEL

We study the problem of computing a data center topology to optimally serve a given communication pattern, where the topology combines static (fixed) and reconfigurable links. Our notation follows the model of [14] for most parts.

Network model. Let $N = (V, E, w)$ be a weighted *hybrid* network [29, 41] connecting the nodes $V = \{v_1, \ldots, v_n\}$ (*e.g.,* top-of-the-rack switches), using (1) static links $E = \{e_1, \ldots, e_m\}$ and (2) reconfigurable links implemented through a reconfigurable (optical circuit) switch. A reconfigurable switch connects the set of nodes V by choosing a matching M on V, where two matched nodes are connected by a bidirectional link. For the sake of generality, we assume each link, whether electrical or optical, comes with a positive weight w (a cost, *e.g.,* latency).

Traffic demands. The resulting network should serve a certain communication pattern, represented as a $|V| \times |V|$ communication matrix D (the *demand matrix*) with positive real-valued entries. An entry (i, j) in D represents the communication frequency from the node v_i to the node v_j.

Optimization objective. We say that the hybrid network N is *configured* by the reconfigurable switch, where the links contained in the matching M are referred to as the *configuration* of N. For ease of notation, we will simply write $N(\mathcal{M})$ to denote the concrete topology resulting from configuration $\mathcal{M}$ and define $dist_{N(\mathcal{M})}(i, j)$ to be the shortest (weighted) distance from node v_i to node v_j on the network $N(\mathcal{M})$. Given a hybrid network N and a communication demand D, our goal is to compute a network $N(\mathcal{M})$ which minimizes the (weighted) average path length for serving D in N by providing a set of matchings $\mathcal{M}$ accordingly. Succinctly stated:

$$\min \sum_{(i,j)\in D} D[i, j] \cdot dist_{N(\mathcal{M})}(i, j) \tag{1}$$

That is, we aim to minimize the sum of the weighted (*i.e.,* by flow size and link costs) path lengths: for each ordered pair of nodes $v_i, v_j \in V$, we multiply the (weighted) length of the shortest path $dist_{N(\mathcal{M})}(i, j)$ from v_i to v_j on $N(\mathcal{M})$ with their entry (i, j) in D. We denote this optical routing problem by ORP.

Problem dimensions. The work in [14] already showed a performance gap between networks with segregated and non-segregated routing, i.e., whether or not the routing may use a combination of static and reconfigurable links. We analyze this distinction from a more fine-grained perspective, namely:

- We introduce a parameter k that defines how often a route may switch between static and reconfigurable links, with $k = 0$ and $k = \infty$ representing the extremes of (non-)segregation.
- We allow nodes to connect more than once to a reconfigurable switch. The number of connections is limited by a hardware available to the node (*i.e.,* the number of optical transmitters and receivers). For a node v, by $\delta_R(v)$ we denote the maximum number of reconfigurable links that v may utilize, and we set $\Delta_R(N) = \max_{u \in V(N)} \delta_R(u)$.
- We also study unidirectional reconfigurable links, where each node v has $\delta_R^{\text{in}}(v)$ incoming and $\delta_R^{\text{out}}(v)$ outgoing reconfigurable links, setting $\delta_R^{\text{in}}(v) + \delta_R^{\text{out}}(v) = \delta_R(v)$.

3 OPTIMALITY FOR SEGREGATED ROUTING

We begin our study with the segregated case (*i.e.,* $k = 0$) and study the parameter Δ_R that defines how often a node may be connected to the reconfigurable switch at most. We can show this variant of ORP to be efficiently solvable if $h_\# = 1$, *i.e.,* the case where one must choose to route each demand between either solely along the static network or a single reconfigurable link (*e.g.,* for elephant flows).

Our result will make use of weighted u-capacitated b-matching algorithms [26], which compute a maximum weight matching for the case where each node v may match $b_v \leq n$ times, with each link e being allowed to be used at most $u_e \leq u$ times. b-matching algorithms were already proposed for reconfigurable networks, *e.g.,* in [39]: however, there the b-matching is used to assign elephant flows to links, without regards to the static network or providing optimality proofs. Conceptually, our proof is inspired by [14, Theorem 1], where the case of $\Delta_R = b = 1$ was considered.

Theorem 3.1. *Let $\Delta_R \in \mathbb{N}$. The resulting reconfigurable routing problem* ORP *with $k = 0$ alternations and $h_\# = 1$ is in P.*

Proof of Theorem 3.1. For each pair of demand entries $d_{i,j}, d_{j,i}$ (possibly of size 0) we compute the non-negative gain $g_{i,j}$ obtained by connecting the nodes i, j in the matching, *i.e.,* the potential route improvement which results from using the reconfigurable link from i to j, multiplied by the combined size of $d_{i,j}, d_{j,i}$. If a reconfigurable link from i to j may not exist, we set $g_{i,j}$ as 0.

We next consider the complete graph G' for the node set V, where the link weights are defined by $g_{i,j}$, and compute a maximum weighted 1-capacitated (each link may only be used once) Δ_R-matching (with $b_v = \delta_R(v)$) on it in polynomial time [26],

where we set the respective matching as the configuration $\mathcal{M}$ of N, ignoring links with $g_{i,j} = 0$. Assume that a better configuration $\mathcal{M}'$ for N, D were to exist: then, we could translate $\mathcal{M}'$ to an improved solution for G', a contradiction. □

Remarks on directed routing. We can extend Theorem 3.1 to apply to unidirectional links as well. To this end, we split each node $v \in V$ into two nodes $v^{\text{in}}, v^{\text{out}}$, where v^{in} takes care of all outgoing demands and reconfigurable links of v, analogously for v^{out}. Matching links that may not exist are assigned a weight of 0, *i.e.,* they provide no benefit.

4 HARDNESS OF NON-SEGREGATION

We continue our study with the non-segregated case. It is known from previous work [14] that ORP is NP-hard for a reconfigurable degree of 1 and multiple alternations.[1] We will now show that for any combination of 1) alternations $k \geq 1$ and 2) reconfigurable degree $\delta_R \geq 1$, the optical routing problem ORP remains NP-hard as well. We start with $\delta_R = 2$ in Section 4.1 and $\delta_R > 2$ in Section 4.2, followed by the more complicated case of $\delta_R = 1$ in Section 4.3.

4.1 Reconfigurable Degree of Two

In this section, we start with the scenario where all nodes have a reconfigurable degree of 2 and then extend it to higher degree combinations in Section 4.2. As our construction will just consist of reconfigurable links, the NP-hardness is independent of the number of allowed alternations k. Still, if a connected static network is desired, we can add it s.t. the problem nature is unchanged, by setting the weights prohibitively high.

Theorem 4.1. *For every number of allowed alternations $k \in \mathbb{N}$ holds: the reconfigurable routing problem* ORP *with reconfigurable degree of $\delta_R(v) = 2$ at every node $v \in V$ is NP-complete.*

Proof. Our proof will by reduction from the NP-hard problem Circular Arrangement [27, §2], which can be equivalently defined as follows, adapted to our model setting: given a set of n nodes V with a demand matrix D with non-negative entries, arrange the n nodes in a graph cycle with unweighted edges, s.t. the weighted average path length is minimized. If every entry in D is positive in an instance I of Circular Arrangement, then the reduction to ORP with $\delta_R(v) = 2$ is immediate: we construct an instance of ORP with the same set of nodes and demands. As any ORP solution that does not form a single Hamiltonian cycle has infinite cost[2], only single cycle solutions are possible, where finding the optimal one is equivalent to solving the Circular Arrangement problem instance I. It remains to cover the case where D has entries of value 0, allowing ORP solutions that differ from forming just one cycle. To this end, we augment those entries to be of size ε, where ε is small enough s.t. it will not affect the optimal matching solution, *e.g.,* by setting it to be the smallest positive entry of D divided by $100n^3$. Lastly, note that for the case of all entries in D being 0, any solution is optimal, and that the corresponding decision problem is clearly in NP. □

[1] A careful analysis of [14, §3.2] reveals that ORP is NP-hard for $k = 2$ (or more) alternations with $\delta_R = 1$, even though it is only stated for $k = \infty$ in [14].

[2] Note that it would also be possible to form a Hamiltonian path, but the only option for the last remaining link is to close the cycle, omitting it does not improve solutions.

Unidirectional links. As Directed Circular Arrangement is also NP-hard [27, §3], the above proof can be directly modified to hold for the unidirectional case with $\delta_R^{\text{in}}(v) = \delta_R^{\text{out}}(v) = 1, \forall v \in V$.

4.2 Beyond a Reconfigurable Degree of Two

We now introduce multiple techniques that allow us to extend the proofs from Section 4.1 to higher reconfigurable degrees. We believe these techniques also to be of independent interest for future work.

Link enforcement. If we want to force two nodes v, v' to match with each other, we can create an arbitrarily high fake demand between them, s.t. any optimal solution must match v and v'. With respect to optimal solutions, the link (v, v') must be created, for both the uni- and bidirectional case.

Cloning technique. We next consider the case where $b = 1$, where we want to either 1) connect some v with $\delta_R(v) = 0$ to the reconfigurable switch (without changing the matching in the optimal solutions), or 2) make the reconfigurable connection of v useless.

To this end, we create ("*clone*") a node v', connected to the reconfigurable switch once, and set arbitrarily high demand between v and v', i.e., we enforce the reconfigurable link. Observe that both v, v' have a reconfigurable degree of $\delta_R(v) = \delta_R(v') = 1$. As v' is not connected to any other nodes, this created (v, v')-link is useless for the original demands, for both uni- and bidirectional links.

2-Extension technique. Consider a network where every node has identical reconfigurable degree of at exactly two, i.e., $\forall v \in V : \delta_R(v) = \Delta_R = 2$. We will now show, first for the bidirectional case, that if ORP is, *e.g.,* NP-hard in such a specific setting, then it is also NP-hard when the reconfigurable degree is increased to some larger $b \in \mathbb{N}$. Similarly, we can also use this extension technique to extend the reconfigurable degree of some subset of nodes from 2 to b for algorithmic purposes, *i.e.,* that it leaves the matching of an optimal solution untouched and all newly created nodes will have a reconfigurable degree of b. To increase the reconfigurable degree from 2 to 3, we create a complete binary tree of depth 3 by enforcing links, where we enforce to connect the root to a node v with a connectivity deficit of one, and two links between the leaves of this tree T_v^1 s.t. all 7 nodes $v_1^1, v_2^1, \ldots v_7^1$ in T_v^1 have a reconfigurable degree of 3. For the unidirectional case, we orient the link (v, v_1^1) towards respectively away from v, analogously for the other links, the reconfigurable degree sum δ_R remains unchanged.

We now show how to directly jump from 2 to b: we create $b - 2$ trees $T_3, \ldots, T_b$, where we enforce 7 cliques, one for each of the seven node groups $v_1^i, \ldots, v_7^i$—each of them thus having a reconfigurable degree of $3 + b - 3 = b$, except for the v_1^is, which have $2 + b - 3 = b - 1$. We then enforce to connect those $b - 2$ v_1^is to v. Again, for the unidirectional case, we orient those links arbitrarily.

By applying the 2-extension technique, Theorem 4.1 can be extended to any fixed reconfigurable degree in $\mathbb{N}$.

Corollary 4.2. *For every number of allowed alternations $k \in \mathbb{N}$ and for every reconfigurable degree $\delta_R(v) = b, b \in \mathbb{N}, b \geq 2, \forall v \in V$ holds: the reconfigurable routing problem* ORP *is NP-complete.*

Furthermore, as the 2-extension technique only increased the number of nodes by a factor of $O(b)$, the reconfigurable degree b can be raised even higher as a function of n, *i.e.,* $b = \lceil f(n) \rceil \geq 2$. As long as this function f remains polynomial, NP-hardness holds.

4.3 Reconfigurable Degree of One

In this section, we show that ORP is NP-complete even in the restricted variant, where all nodes have a reconfigurable degree of 1, and with at most 1 alternation for routing of any demand. Our construction unfolds in two stages. First, we introduce an auxilary variant of ORP problem called ℓ-ORP: for any integer ℓ, by ℓ-ORP we denote the variant of ORP, where the reconfigurable network $\mathcal{M}$ consists of at most ℓ edges. In Lemma 4.3, we present a polynomial time reduction from ORP to ℓ-ORP. Then, in Lemma 4.4, we reduce the classic Vertex Cover problem to ℓ-ORP. By $A \prec_P B$ we denote the existence of a polynomial-time reduction from the problem A to the problem B.

Lemma 4.3. *For any positive integer ℓ, we have ℓ-ORP $\prec_P$ ORP.*

Proof. Consider any ℓ-ORP instance I with the static network G. We assume that G is normalized, i.e. the minimum weight of an edge is 1. We construct an instance I' of ORP that simulates I. Precisely, we prove that I has a solution of cost at most Thr iff I' has a solution of cost at most

$$Thr' := Thr + (2 \cdot (\lfloor n/2 \rfloor - \ell)) \cdot ((n-1) \cdot (\mathcal{D}+1) + \mathcal{D}) \cdot (Thr + 1) \ ,$$

where $\mathcal{D}$ is the maximum weight of the shortest weighted path between any two nodes in G.

We preserve the static links weights, the reconfigurable links weights, and the demands between every pair of nodes from G. We introduce an additional set of nodes $\mathbb{A}$ of size $2 \cdot (\lfloor n/2 \rfloor - \ell)$. We connect every node from $\mathbb{A}$ with every node from G by a static link with weight $\mathcal{D}+1$. Every reconfigurable link between $\mathbb{A}$ and G has weight $\mathcal{D}$. We produce additional demands of volume $Thr+1$ from every node from $\mathbb{A}$ to every node from G.

Consider a demand between a pair of nodes $a \in \mathbb{A}$ $b \in V(G)$. The optimal routing of a demand from a to b costs $\mathcal{D}$ if a reconfigurable link (a, b) is present, and costs $\mathcal{D}+1$ otherwise. If a reconfigurable link is not present, every non-direct route costs at least $\mathcal{D}+1$: the cost at least $\mathcal{D}$ is incurred between a and any node $c \in V(G)$, and the cost at least 1 is incurred between b and c (the static network is normalized). Complementary, the optimal route between a and b costs at most $\mathcal{D}+1$, as a direct static link of such weight exists.

Note that providing $\mathbb{A}$ with less than $|\mathbb{A}|$ reconfigurable links results in surpassing the threshold Thr'. As at most one reconfigurable link can be adjacent to any node, each node from $\mathbb{A}$ incurs the cost of at least $((n-1) \cdot (\mathcal{D}+1) + \mathcal{D}) \cdot (Thr+1)$ for its demands. Every node from $\mathbb{A}$ with no adjacent reconfigurable link incurs the cost at least $(n \cdot (\mathcal{D}+1)) \cdot (Thr+1)$, which incurs additional cost at least $Thr+1$, which cannot be compensated by savings in routing demands among nodes in G. As the maximum reconfigurable degree (Δ_R) is 1, in every solution to I' with cost at most Thr', every node from $\mathbb{A}$ has a reconfigurable link to some node in G.

To reconstruct the solution to I, we take the reconfigurable links among nodes from G from the solution to I'. Now, we claim that the reconstructed solution has exactly ℓ reconfigurable links. In any graph with n vertices, the maximum size of any matching is $\lfloor n/2 \rfloor$. To restrict it to ℓ edges, we need to remove $\lfloor n/2 \rfloor - \ell$ matching edges. To prevent one edge from appearing, we need to reduce the number of matchable nodes by 2. Each node from $\mathbb{A}$ matches to one node from G, and $|\mathbb{A}| = 2 \cdot (\lfloor n/2 \rfloor - \ell)$.

As every node of $\mathbb{A}$ has exactly one reconfigurable link to a node from G, the cost of routing demands between $\mathbb{A}$ and G is exactly $|\mathbb{A}| \cdot ((n-1) \cdot (\mathcal{D}+1) + \mathcal{D}) \cdot (Thr+1)$. By the definition of the threshold Thr', the remaining budget for routing demands inside G is Thr. Note that we preserve the shortest paths among nodes from G: by the weight of static and reconfigurable links between $\mathbb{A}$ and G, the routes through $\mathbb{A}$ weigh more than any path in the original network. Hence, the cost of reconstructed solution to I is at most Thr. □

Lemma 4.4. *It holds that* Vertex Cover $\prec_P \bigcup_\ell \ell$-ORP.

Proof. For an integer t, the decision version of a Vertex Cover is a problem of determining an existence of a vertex cover of size at most t. Consider any decision Vertex Cover instance $\langle G, t \rangle$, where $G = \langle V, E \rangle$. We produce a ℓ-ORP instance (where $\ell = |E| + t$) that has a feasible solution that satisfies a threshold $Thr := 5 \cdot |E|$ iff there exist a vertex cover of G of size at most t.

The construction unfolds as follows. For each vertex $v \in V$ we produce a Vertex Gadget that consists of two nodes: a_v and b_v. For each edge $e \in E$ we produce an Edge Gadget that consists of three nodes: l_e, m_e and r_e, and two edges of weight 3: (l_e, m_e) and (r_e, m_e). For each edge $e = (u, v) \in E$ we produce two edges of weight 2: (m_e, b_u) and (m_e, b_v) and two edges of weight 1: (a_u, l_e) and (a_v, r_e). For each edge $e \in E$, reconfigurable links (l_e, m_e) and (r_e, m_e) have weight 1 and for each vertex $v \in V$, a reconfigurable link (a_v, b_v) has weight 1. Remaining reconfigurable links $(x, y) \in V \times V$ have weight equal to the shortest path (via static links only) between x and y in graph G, and an appearance of such a reconfigurable link does not improve routing of any demand. For each edge $e = (u, v) \in E$ we produce two unitary demands: (m_e, a_u) and (m_e, a_v), and we call those the *cover demands* of e. The construction is depicted in Figure 2.

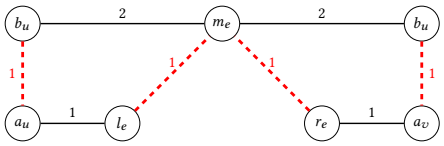

Figure 2: Construction for an edge e adjacent to vertices u and v. Static links are drawn solid, with their weight denoted next to them. Only reconfigurable links that can possibly improve the routing are shown (dashed). We omit the other links and the reconfigurable switch in this figure for better visibility.

Consider a demand (m_e, a_u). We distinguish among three ways of routing the demand: In presence of a reconfigurable link (m_e, l_e), the *short route* of weight 2 consists of nodes $m_e \to l_e \to a_u$. In presence of a reconfigurable link (a_v, b_v), the *medium route* of weight 3 consists of nodes $m_e \to b_u \to a_u$. We classify every other route (of weight at least 4) as the *long route*. Symmetrically, for a demand (m_e, a_v), analogous short routes through vertex r_e (instead of l_e) exist.

We say that a vertex $v \in V$ is *active* if a reconfigurable link (a_v, b_v) appears. Now, we argue that at most t vertices are active. Assume that more than t vertices are active. As we have at most $t + |E|$ reconfigurable links, there exists $f \in E$ such that none of reconfigurable links $\{(m_f, l_f), (m_f, r_f)\}$ exists. In this case, no short route for the cover demands of f exists, and the cost incurred for

them is at least 6. For the remaining cover demands $e \in E \setminus \{f\}$: as $\Delta_R = 1$, at most one reconfigurable link from $\{(m_e, l_e), (m_e, r_e)\}$ exists. Hence, at most one of the cover demands of e can be routed by the short route of cost 2, and the minimum cost of routing of both cover demands of e is 5. Summing up, the total cost is $6 + 5 \cdot (|E| - 1) > Thr$, a contradiction.

To reconstruct the solution to the Vertex Cover, we take active vertices. Now, we argue that such solution covers all edges. Note that any solution that routes any demand by a long route exceeds the threshold. We stated previously that for each $e = (u, v) \in E$, at most one of the cover demands of e is routed by the short route. Hence, exactly one of the cover demands of e is routed by a path of cost 3, and by construction the only path of such weight is the medium route to either u or v. The existence of a medium path implies that either u or v is active, and hence e is covered.

Finally, we show how to reconstruct the ℓ-ORP solution from the vertex cover. Consider an edge $e = (u, v) \in E$, and assume that it is covered by u. We route the demand (m_e, a_u) by the medium route, and we route the demand (m_e, a_v) by the short route, placing reconfigurable links to allow the existance of such routes. □

Remarks on directed routing. Note that we can modify Lemmas 4.3 and 4.4 to show hardness in the directed routing model. Instead of setting $\Delta_R = 1$, we set $\delta_R^{\text{in}} = 1$ and $\delta_R^{\text{out}} = 1$ (note that those values are minimal for any reconfigurable links to appear). To show that we can reduce the number of reconfigurable links to ℓ, we modify Lemma 4.3 in the following way: we direct the reconfigurable and static links, and demands between A and G towards nodes of A. As the maximum number of reconfigurable links in the directed routing problem is n (rather than $\lfloor n/2 \rfloor$), we adjust the size of set A to $n - \ell$, and we adjust the threshold value accordingly: $Thr' := Thr + (n - \ell)) \cdot ((n - 1) \cdot (\mathcal{D} + 1) + \mathcal{D}) \cdot (Thr + 1)$. In any solution of the cost at most Thr, each node from A has an incoming links, and the number of links inside G is $n - (n - \ell) = \ell$. Finally, we modify Lemma 4.4 by directing every reconfigurable and static link, and every demand from m_e towards a_v. Note that although the model allows for multiple hops through reconfigurable links, in our construction we used paths with at most one reconfigurable link.

Conclusions. By combining Lemma 4.3, Lemma 4.4, and the transitivity of relation $\prec_P$, we obtain that ORP is NP-complete. The problem remains NP-complete even if we allow at most one alternation, and at most one hop through reconfigurable network in routing of any demand.

5 RELATED WORK

Most existing literature on data center network design deals with demand-oblivious topologies, see [33] for a recent survey. In contrast, we in this paper are interested in demand-aware network designs, which not only arise in data centers but also in wide area networks, e.g., [21, 22, 37].

We are not the first to explore non-segregated routing in hybrid networks. In particular, Xia *et al.* [44] leverage converter switches to dynamically convert between a Clos network and approximate random graphs of different sizes. Venkatakrishnan *et al.* [41] show that routing policies restricted to direct or single-hop routing are inefficient and present near optimal scheduling algorithms, however, only for the segregated case; the general case is stated as an open problem. An orthogonal approach is taken by Mellette *et al.* [31] who consider switches which rotate through a set of pre-defined matchings, also leveraging Valiant-style [40] multi-hop optical connections.

We in this paper are particularly interested in network design and routing algorithms which come with *formal (approximation or optimality) guarantees*. Most prior algorithmic works usually assume segregated routing models and rely on heuristics based on matchings [8, 13, 28, 29, 42], edge-coloring [11], or stable-marriage algorithms [15], see [23, 43]. Avin et al. [5] presented a constant-degree network design algorithm which achieves a constant approximation of the optimal expected route length, which is shown to be proportional to the conditional entropy of the workload. Avin et al. [4] also presented a resilient demand-aware network based on coding, but with unbounded degree. However, the above results concern fully reconfigurable networks, where all links are reconfigurable. Closer to our work (and reality) are the results by Foerster et al. [14] who provide polynomial-time *exact* (i.e., optimal) algorithms, for specific demands and models, and also derive first hardness results. We in this paper extend [14] by investigating the complexity of more general non-segregated routing.

The problem of enhancing a given static network with a reconfigurable topology is related to classic combinatorial problems arising in graph theory. For example, Manos et al. [34] presented algorithms to augment a given graph which "ghost edges" to provide small world properties and short path lengths, see also the recent paper by Gozzard for a good overview of the state-of-the-art [16]. The underlying problems are also related to the *k-median* problem [32] and known to be hard, even to approximate, in general [35]. Besides considering shortest paths, researchers have also investigated algorithms to reduce the network diameter [9, 12]. In contrast to these works, motivated by emerging optical switches, we consider the problem of adding entire matchings, hence introducing a new perspective on the b-matching literature [1, 24], typically arising in market situations where, e.g., users need to be matched to a cardinality-constrained set of items, e.g., matching children to schools. We in this paper are only interested in the route length between nodes which actually communicate.

Finally, we note that there also exist results on dynamic network design algorithms which aim to strike a balance between reconfiguration costs and providing shorter routes [3, 6, 20, 36], as well as for the case where links need to be removed for maintenance [46].

6 CONCLUSION

This paper showed that more flexible, non-segregated routing policies can introduce additional algorithmic complexities. In particular, we presented algorithms and charted a detailed complexity landscape of non-segregated routing. We hence hope that our results can be useful and provide a more complete picture of the benefits and costs when moving beyond segregated routing.

Even though we focused on the popular model of one reconfigurable switch in this paper [23, 43], the case of multiple such

switches is also of importance [31, 44]. Our hardness results naturally transfer to this extension, and in most non-segregated scenarios, there is not much difference between algorithms for one or multiple switches, as multiple reconfigurable switches can be emulated by one switch, combining 1) large weights for not permitted reconfigurable links and 2) fake child nodes for each node to enforce the inter-switch connectivity constraints.

There still remain several interesting open problems for future research. In particular, it will be interesting to shed light on the complexity of *specific* network topologies. Furthermore, while we have focused on exact algorithms, it remains to explore the complexity of (provably) *approximate* algorithms in more depth.

REFERENCES

[1] Faez Ahmed, John P. Dickerson, and Mark Fuge. 2017. Diverse Weighted Bipartite b-Matching. In *IJCAI*. ijcai.org, 35–41.

[2] Mohammad Al-Fares, Alexander Loukissas, and Amin Vahdat. 2008. A scalable, commodity data center network architecture. In *SIGCOMM*. ACM, 63–74.

[3] Chen Avin, Michael Borokhovich, Bernhard Haeupler, and Zvi Lotker. 2013. Self-adjusting Grid Networks to Minimize Expected Path Length. In *SIROCCO*. 36–54.

[4] Chen Avin, Alexandr Hercules, Andreas Loukas, and Stefan Schmid. 2018. *rDAN*: Toward robust demand-aware network designs. *Inf. Process. Lett.* 133 (2018), 5–9.

[5] Chen Avin, Kaushik Mondal, and Stefan Schmid. 2017. Demand-Aware Network Designs of Bounded Degree. In *DISC (LIPIcs)*, Vol. 91. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 5:1–5:16.

[6] Chen Avin, Kaushik Mondal, and Stefan Schmid. 2018. Push-Down Trees: Optimal Self-Adjusting Complete Trees. *CoRR* abs/1807.04613v1 (2018).

[7] Chen Avin and Stefan Schmid. 2018. Toward Demand-Aware Networking: A Theory for Self-Adjusting Networks. In *ACM SIGCOMM Computer Communication Review (CCR)*, Vol. 48.

[8] Navid Hamed Azimi, Zafar Ayyub Qazi, Himanshu Gupta, Vyas Sekar, Samir R. Das, Jon P. Longtin, Himanshu Shah, and Ashish Tanwer. 2014. FireFly: a reconfigurable wireless data center fabric using free-space optics. In *SIGCOMM*. ACM, 319–330.

[9] Davide Bilò, Luciano Gualà, and Guido Proietti. 2012. Improved approximability and non-approximability results for graph diameter decreasing problems. *Theoretical Computer Science* 417 (2012), 12–22.

[10] Kai Chen, Ankit Singla, Atul Singh, Kishore Ramachandran, Lei Xu, Yueping Zhang, Xitao Wen, and Yan Chen. 2014. OSA: An Optical Switching Architecture for Data Center Networks With Unprecedented Flexibility. *IEEE/ACM Trans. Netw.* 22, 2 (2014), 498–511.

[11] Li Chen, Kai Chen, Zhonghua Zhu, Minlan Yu, George Porter, Chunming Qiao, and Shan Zhong. 2017. Enabling Wide-Spread Communications on Optical Fabric with MegaSwitch. In *NSDI*. USENIX Association, 577–593.

[12] Erik D Demaine and Morteza Zadimoghaddam. 2010. Minimizing the diameter of a network using shortcut edges. In *Proc. Scandinavian Workshop on Algorithm Theory (SWAT)*. Springer, 420–431.

[13] Nathan Farrington, George Porter, Sivasankar Radhakrishnan, Hamid Hajabdolali Bazzaz, Vikram Subramanya, Yeshaiahu Fainman, George Papen, and Amin Vahdat. 2010. Helios: a hybrid electrical/optical switch architecture for modular data centers. In *SIGCOMM*. ACM, 339–350.

[14] Klaus-Tycho Foerster, Manya Ghobadi, and Stefan Schmid. 2018. Characterizing the algorithmic complexity of reconfigurable data center architectures. In *ANCS*. IEEE/ACM, 89–96.

[15] Monia Ghobadi, Ratul Mahajan, Amar Phanishayee, Nikhil R. Devanur, Janardhan Kulkarni, Gireeja Ranade, Pierre-Alexandre Blanche, Houman Rastegarfar, Madeleine Glick, and Daniel C. Kilper. 2016. ProjecToR: Agile Reconfigurable Data Center Interconnect. In *SIGCOMM*. ACM, 216–229.

[16] Andrew Gozzard, Max Ward, and Amitava Datta. 2018. Converting a network into a small-world network: Fast algorithms for minimizing average path length through link addition. *Information Sciences* 422 (2018), 282–289.

[17] Albert G. Greenberg, James R. Hamilton, Navendu Jain, Srikanth Kandula, Changhoon Kim, Parantap Lahiri, David A. Maltz, Parveen Patel, and Sudipta Sengupta. 2009. VL2: a scalable and flexible data center network. In *SIGCOMM*. ACM, 51–62.

[18] Chuanxiong Guo, Guohan Lu, Dan Li, Haitao Wu, Xuan Zhang, Yunfeng Shi, Chen Tian, Yongguang Zhang, and Songwu Lu. 2009. BCube: a high performance, server-centric network architecture for modular data centers. In *SIGCOMM*. ACM, 63–74.

[19] Daniel Halperin, Srikanth Kandula, Jitendra Padhye, Paramvir Bahl, and David Wetherall. 2011. Augmenting data center networks with multi-gigabit wireless links. In *SIGCOMM*. ACM, 38–49.

[20] Sikder Huq and Sukumar Ghosh. 2017. Locally Self-Adjusting Skip Graphs. In *ICDCS*. IEEE Computer Society, 805–815.

[21] Su Jia, Xin Jin, Golnaz Ghasemiesfeh, Jiaxin Ding, and Jie Gao. 2017. Competitive analysis for online scheduling in software-defined optical WAN. In *INFOCOM*. IEEE, 1–9.

[22] Xin Jin, Yiran Li, Da Wei, Siming Li, Jie Gao, Lei Xu, Guangzhi Li, Wei Xu, and Jennifer Rexford. 2016. Optimizing Bulk Transfers with Software-Defined Optical WAN. In *SIGCOMM*. ACM, 87–100.

[23] Christoforos Kachris and Ioannis Tomkos. 2012. A Survey on Optical Interconnects for Data Centers. *IEEE Communications Surveys and Tutorials* 14, 4 (2012), 1021–1036.

[24] Bala Kalyanasundaram and Kirk Pruhs. 2000. An optimal deterministic algorithm for online b-matching. *Theor. Comput. Sci.* 233, 1-2 (2000), 319–325.

[25] Simon Kassing, Asaf Valadarsky, Gal Shahaf, Michael Schapira, and Ankit Singla. 2017. Beyond fat-trees without antennae, mirrors, and disco-balls. In *SIGCOMM*. ACM, 281–294.

[26] Adam N. Letchford, Gerhard Reinelt, and Dirk Oliver Theis. 2008. Odd Minimum Cut Sets and b-Matchings Revisited. *SIAM J. Discrete Math.* 22, 4 (2008), 1480–1487.

[27] Vincenzo Liberatore. 2004. Circular arrangements and cyclic broadcast scheduling. *J. Algorithms* 51, 2 (2004), 185–215.

[28] He Liu, Feng Lu, Alex Forencich, Rishi Kapoor, Malveeka Tewari, Geoffrey M. Voelker, George Papen, Alex C. Snoeren, and George Porter. 2014. Circuit Switching Under the Radar with REACToR. In *NSDI*. USENIX Association, 1–15.

[29] He Liu, Matthew K. Mukerjee, Conglong Li, Nicolas Feltman, George Papen, Stefan Savage, Srinivasan Seshan, Geoffrey M. Voelker, David G. Andersen, Michael Kaminsky, George Porter, and Alex C. Snoeren. 2015. Scheduling techniques for hybrid circuit/packet networks. In *CoNEXT*. ACM, 41:1–41:13.

[30] Vincent Liu, Daniel Halperin, Arvind Krishnamurthy, and Thomas E. Anderson. 2013. F10: A Fault-Tolerant Engineered Network. In *NSDI*. USENIX, 399–412.

[31] William M. Mellette, Rob McGuinness, Arjun Roy, Alex Forencich, George Papen, Alex C. Snoeren, and George Porter. 2017. RotorNet: A Scalable, Low-complexity, Optical Datacenter Network. In *SIGCOMM*. ACM, 267–280.

[32] Adam Meyerson and Brian Tagiku. 2009. Minimizing Average Shortest Path Distances via Shortcut Edge Addition. In *APPROX-RANDOM*. 272–285.

[33] Mohammad Noormohammadpour and Cauligi S Raghavendra. 2017. Datacenter Traffic Control: Understanding Techniques and Tradeoffs. *IEEE Communications Surveys & Tutorials* 20, 2 (2017), 1492–1525.

[34] Manos Papagelis, Francesco Bonchi, and Aristides Gionis. 2011. Suggesting Ghost Edges for a Smaller World. In *Proc. 20th ACM International Conference on Information and Knowledge Management (CIKM)*. 2305–2308.

[35] Nikos Parotsidis, Evaggelia Pitoura, and Panayiotis Tsaparas. 2015. Selecting shortcuts for a smaller world. In *Proc. SIAM International Conference on Data Mining*. SIAM, 28–36.

[36] Stefan Schmid, Chen Avin, Christian Scheideler, Michael Borokhovich, Bernhard Haeupler, and Zvi Lotker. 2016. SplayNet: Towards Locally Self-Adjusting Networks. *IEEE/ACM Trans. Netw.* 24, 3 (2016), 1421–1433.

[37] Rachee Singh, Manya Ghobadi, Klaus-Tycho Foerster, Mark Filer, and Phillipa Gill. 2018. RADWAN: Rate Adaptive Wide Area Network. In *SIGCOMM*. ACM.

[38] Ankit Singla, Chi-Yao Hong, Lucian Popa, and Philip Brighten Godfrey. 2012. Jellyfish: Networking Data Centers Randomly. In *NSDI*. USENIX Association, 225–238.

[39] Ankit Singla, Atul Singh, Kishore Ramachandran, Lei Xu, and Yueping Zhang. 2010. Proteus: a topology malleable data center network. In *HotNets*. ACM, 8.

[40] Leslie G. Valiant. 1982. A Scheme for Fast Parallel Communication. *SIAM J. Comput.* 11, 2 (1982), 350–361.

[41] Shaileshh Bojja Venkatakrishnan, Mohammad Alizadeh, and Pramod Viswanath. 2016. Costly Circuits, Submodular Schedules and Approximate Carathéodory Theorems. In *SIGMETRICS*. ACM, 75–88.

[42] Guohui Wang, David G. Andersen, Michael Kaminsky, Konstantina Papagiannaki, T. S. Eugene Ng, Michael Kozuch, and Michael P. Ryan. 2010. c-Through: part-time optics in data centers. In *SIGCOMM*. ACM, 327–338.

[43] Wenfeng Xia, Peng Zhao, Yonggang Wen, and Haiyong Xie. 2017. A Survey on Data Center Networking (DCN): Infrastructure and Operations. *IEEE Communications Surveys and Tutorials* 19, 1 (2017), 640–656.

[44] Yiting Xia, Xiaoye Steven Sun, Simbarashe Dzinamarira, Dingming Wu, Xin Sunny Huang, and T. S. Eugene Ng. 2017. A Tale of Two Topologies: Exploring Convertible Data Center Network Architectures with Flat-tree. In *SIGCOMM*. ACM, 295–308.

[45] Xia Zhou, Zengbin Zhang, Yibo Zhu, Yubo Li, Saipriya Kumar, Amin Vahdat, Ben Y. Zhao, and Haitao Zheng. 2012. Mirror mirror on the ceiling: flexible wireless links for data centers. In *SIGCOMM*. ACM, 443–454.

[46] Danyang Zhuo, Monia Ghobadi, Ratul Mahajan, Klaus-Tycho Foerster, Arvind Krishnamurthy, and Thomas E. Anderson. 2017. Understanding and Mitigating Packet Corruption in Data Center Networks. In *SIGCOMM*. ACM, 362–375.