IE-490 Final Report

Özgün Barış Kır 21803466

Fall 2021

Abstract

Due to the fast advancements in the field of cloud technologies, data science, and artificial intelligence applications, data workloads in the data centers grow significantly. Hence, efficient traffic routing in those centers becomes more and more necessary. Moreover, new optical transfer technologies allow us to establish non-segregated reconfigurable networks that can improve data transfer cost and time in such a network. In this work, we pursued an extension to [Fenz et al., 2019] whose work provides different heuristic algorithms and a model for one reconfigurable switch networks. As the complexity of the model is NP-Hard and could not work efficiently in the large instances, we improved their model in terms of running time. Furthermore, we provided a more general model that is also valid for networks with multiple switches.

1 Introduction

Digital society has been creating enormous amount of data over the last decades. Thanks to the ever growing digitization, advancements in artificial intelligence and machine learning, this trend tends to enlarge ever more in the coming years [Foerster et al., 2019]. As this increasing data load is handled in data centers around the world, they have become a critical infrastructure of our society [Yang et al., 2019]. Hence, designing those data centers and routing the data flow inside them efficiently gained a lot of attention from both academia and business. Traditional data centers have mostly utilized electrical connections, also called static topology, to send the data, whereas recent developments now allow us to benefit from the optical connections, also called the optic topology [Bienkowski et al., 2021][Pal and Kant, 2015]. Optic connections are more reliable and faster in terms of data transfer compare to the static connections [Fenz et al., 2019]. However, they are expensive to employ in the data center's every connection and most data centers have already had a static topology which is not impractical to build again from the scratch [Luo et al., 2020]. Hence, hybrid topologies comes forward to replace obsolete static data centers that has only static topologies [Foerster and Schmid, 2019]. However, first hybrid topologies are not made used of by employing reconfigurable connections between data points which means that limited connections are selected initially to be made optical and they were not changed later. This method obviously brought inefficient use of optical connections as demand changes constantly. One step forward comes with the help of devices called reconfigurable switches. These switches establish optical connections, also called matchings, between two data points. Since these matchings are not permanent and can be changed according to the incoming data demand, the data networks that has them are called reconfigurable and demand aware networks [Foerster and Schmid, 2019]. [Luo et al., 2019] show that reconfigurable networks increase the throughput rate of the network and number of accepted requests up to 70%. This efficiency increase mainly comes from the fact that 1% of the node pairs in the network account for 80% percent of the total traffic[Foerster and Schmid, 2019].

1.1 Our Contribution

In this report, first we will review the recent developments in this field particularly about the demand-aware reconfigurable data center design. We will provide a better mathematical model for the networks with one reconfigurable switch than the [Fenz et al., 2019]'s model and show that our model has a much lower running time. Then, we will provide general model for multiple reconfigurable switches. After that we will discuss the shortcomings of the heuristic algorithms proposed by [Fenz et al., 2019].

2 Literature Review

The authors in [Foerster and Schmid, 2019] summarizes the history of reconfigurable switch technologies in the data centers starting from 2009 such as Optical Network Switches, 60 GHz Wireless, and Free-Space Optics. They also cite that the matchings in the data center occur mainly in the specific regions, %80 percent of the traffic comes from %1 percent of the nodes. This regionality of the traffic suggests that reconfigurable switches can be implemented to lower the cost without changing the performance. They also point out that in some settings, it can also be interesting to study the removal of edges. Finally, they also note that reconfigurable networks are not only arising in data centers, but for example also in wide-area networks suggesting it might be worth to use this model in other fields as well. The authors in [Luo et al., 2019] focused on Wide Area Networks and optimized them without rescheduling or preemption. They provided a model as well. In another work of the [Foerster et al., 2019], they developed an extension with focusing on two matters in the model. Namely, they are the maximum simultaneous connections a node has to the reconfigurable switch and the number of alternations. The analytical results of this paper also revealed that while reconfigurable interconnects make data center networks more flexible, exploitation of this greater flexibility to route traffic demands more effectively faces steep computational challenges which we will hopefully ease of with better modelling. The analysis also suggests a need for a better understanding on algorithmic complexity of approximate demand-aware routing with provable approximation guarantees which could be an interesting research idea in the future research. In the another paper of the same authors [Luo et al., 2020], they worked on the spittable multicast matchings which is another approach to solve the matching problem of the model. Another paper [Yang et al., 2019] deal with ever growing traffic in data centers by comparing different routing strategies and scheduling large flows. They also developed a decomposition technique with performance guarantee. Secondly, they come about with two flow scheduling solutions, WiRo and OFS, to achieve efficient routing in reconfigurable DCNs with different workloads. One difference with our work is that they expanded the routing policy by introducing 60GHz Wireless flow to the network whereas we focus on implementing optical wires in addition to the existing topology. They used one hour of university's data traffic (containing

about one million flow entries) to lay the foundation of their work. Furthermore, they provided a model which they also proved that NP-Hard. Another novel work [Bienkowski et al., 2021] concentrates on B-matchings, a generalization of maximum weight matching where each node has at most b1 adjacent matching edges in a similar network. Their main contribution happens to be an online algorithm which is also O(b) competitive. They have made simulations with real-life data as well. Even though [Fenz et al., 2019] proposed their model and heuristics for networks with one reconfigurable switch, [Xue et al., 2018] argued that networks with multiple switches offers different combinations of routing and their algorithms and complexities are yet to be explored.

3 Problem Definition

This is mainly a design and routing problem. While the demand is routed along the network, the matching decisions are also made. Hence, we have mainly two set of decision variables, one of them is for matchings and the other one for routing. In [Fenz et al., 2019]' model the latter is taken as finding the shortest path between demand source and destination. Whereas in our model, they are taken as flow variables and the aim is to minimize the total cost of the summation of flow variables. Another part of the problem is network. We did not consider any specific network not to lose generality but some of the authors have used specialized tree networks [Foerster and Schmid, 2019]. The first layer of the network is static topology, it is connected to all nodes, thus, it is possible to go from any node to any node just by using static topology. When we created the random data to represent that feature, we have used a connected tree. Although it is a connected graph, it does not have to be a spanning tree, it may contain circles, as there is no restriction not to lose generality. However, the optic topology does not have to be all connected. In fact, each reconfigurable switch has a set of nodes that it is connected to that is its own optic topology. In the multiple reconfigurable switch setting, a node might connected to more than one node. For a node, connecting to a switch does not mean this connection is open always, it is only a potential optical connection and unless the model decides to put one of its matchings on that arcs it is not an open connection. A node can make at most one matching on a reconfigurable switch. Hence, if a node is connected to more than one switch it might match more than one node on separate switches. One of the significant element of this problem is that when there is a relatively large traffic between two nodes, this is called elephant flow, the models tries to put a matching there initially. Differently, when there is not much of a demand between two nodes, it is called mouse flows, and the models beware to put matchings there. Fortunately, these elephant-mouse flows are prevalent [Foerster and Schmid, 2019]. Hence, they are the main framework for heuristic algorithms. As proven by the [Foerster et al., 2018], the problem is NP-hard. We provided the following figure to show the multiple reconfigurable switch setting in a small network with 3 switches.

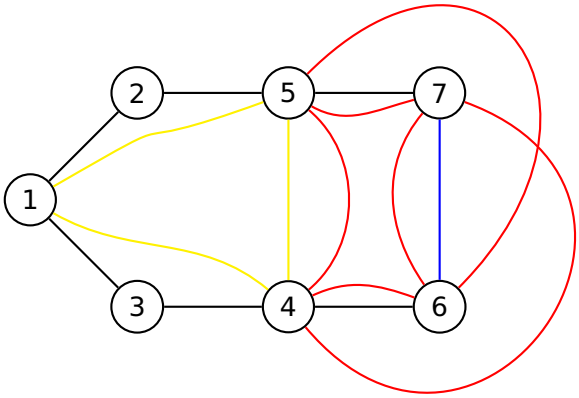

Figure 1: A Network with 3 Switches

First reconfigurable switch represented by red arcs is connected to nodes 5, 7, 4, 6; second switch is connected to 6 and 7 referred by blue arc; third switch is connected to 1, 4, 5 alluded by yellow arcs. For instance, there are two matchings options between node 6 and 7, one on first switch and the other is on second.

4 Mathematical Models

4.1 Parameters, Sets, and Decision Variables

Sets and Parameters:

N:Set of nodes
O:Set of optic arcs
A:Set of all arcs(union of static and optic arcs)
s_{ij}:represents the weight of the static link from i to j ($s_{ij} \geq o_{ij}$)
o_{ij}:represents the weight of the optic link from i to j
D_{ij}: size of the demand value from i to j

Decision Variables:

x_{ij}^{st}:is set to 1 if a link from i to j is used in the shortest path from s to t
y_{ij}^{st}:is set to 1 if an optic link from i to j is used in the shortest path from s to t and
m_{ij}:equals to 1 when there is an optic matching between node i and j, 0 otherwise
$dist_{st}$: the resulting shortest distance between s and t

4.2 [Fenz et al., 2019]'s Model for One Reconfigurable Switch

$$\text{minimize} \quad \sum_{s}\sum_{t} D_{st} dist_{st} \tag{1a}$$

$$\text{subject to} \quad \sum_{j=1}^{n} m_{ij} \leq 1 \qquad \forall i \in N, \tag{1b}$$

$$\sum_{j=1}^{n} m_{ji} \leq 1 \qquad \forall i \in N, \tag{1c}$$

$$m_{ij} = m_{ji} \qquad \forall (i,j) \in O, \tag{1d}$$

$$\sum_{j:(i,j)\in A} x_{ij}^{st} - \sum_{j:(j,i)\in A} x_{ji}^{st} = \begin{cases} 1 & if \quad i = s \\ -1 & if \quad i = t \\ 0 & else \end{cases} \qquad \forall i,s,t \in N, \tag{1e}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n} s_{ij}(x_{ij}^{st} - y_{ij}^{st}) + o_{ij} y_{ij}^{st} = dist_{st} \qquad \forall s,t \in N, \tag{1f}$$

$$y_{ij}^{st} \leq m_{ij} \qquad \forall (i,j) \in O \quad \forall s,t \in N, \tag{1g}$$

$$y_{ij}^{st} \leq x_{ij}^{st} \qquad \forall (i,j) \in A \quad \forall s,t \in N, \tag{1h}$$

$$x_{ij}^{st}, y_{ij}^{st} \geq 0 \qquad \forall i,j,s,t \in N, \tag{1i}$$

$$m_{ij} \in \{0,1\} \qquad \forall (i,j) \in O \tag{1j}$$

4.3 Our Model for One Reconfigurable Switch

New Decision Variables:

f_{ij}^s:static flow value from i to j when the seed is s

g_{ij}^s:optic flow value from i to j when the seed is s

$$\text{minimize} \quad \sum_{s\in N}\sum_{(i,j)\in A} s_{ij} f_{ij}^s + o_{ij} g_{ij}^s \tag{2a}$$

$$\text{subject to} \quad \sum_{j:(j,i)\in A} f_{ji}^s + g_{ji}^s - \sum_{j:(i,j)\in A} f_{ij}^s + g_{ij}^s = \begin{cases} -\sum_{t:t\neq s; t\in N} D_{st} & i = s \\ D_{si} & i \neq s \end{cases} \quad \forall s \in N, \tag{2b}$$

$$\sum_s g_{ij}^s \leq M m_{ij} \qquad \forall (i,j) \in O, \tag{2c}$$

$$\sum_{j=1}^{n} m_{ij} \leq 1 \qquad \forall i \in N, \tag{2d}$$

$$\sum_{j=1}^{n} m_{ji} \leq 1 \qquad \forall i \in N, \tag{2e}$$

$$m_{ij} = m_{ji} \qquad \forall (i,j) \in O, \tag{2f}$$

$$f_{ij}^s, g_{ij}^s \geq 0 \qquad \forall i,j,s \in N, \tag{2g}$$

$$m_{ij} \in \{0,1\} \qquad \forall (i,j) \in O$$

4.4 Our Model for K Reconfigurable Switch

New Sets:

A^S: All the static links in the static topology

A^O: All optic links connected to all reconfigurable switches (Union of O_t's)

O_t: All optic links connected to reconfigurable switch t

$$\text{minimize} \quad \sum_{s \in N} \Big(\sum_{(i,j) \in A^S} s_{ij} f_{ij}^s + \sum_{(i,j) \in A^O} o_{ij} g_{ij}^s \Big) \tag{3a}$$

$$\text{subject to} \quad \sum_{j=1}^{n} m_{ij}^t \le 1 \qquad \forall i \in N \quad \forall t, \tag{3b}$$

$$\sum_{j=1}^{n} m_{ji}^t \le 1 \qquad \forall i \in N \quad \forall t, \tag{3c}$$

$$m_{ij}^t = m_{ji}^t \qquad \forall (i,j) \in O_t \quad \forall t, \tag{3d}$$

$$\sum_{j:(j,i) \in A^S} f_{ji}^s + \sum_{j:(j,i) \in A^O} g_{ji}^s - \sum_{j:(i,j) \in A^S} f_{ij}^s - \sum_{j:(i,j) \in A^O} g_{ij}^s = \begin{cases} -\sum_{t:t \ne s; t \in N} D_{st} & i = s \\ D_{si} & i \ne s \end{cases} \quad \forall s \in N, \tag{3e}$$

$$\sum_{s} g_{ij}^s \le M \sum_{t=1}^{k} m_{ij}^t \qquad \forall (i,j) \in O_t, \tag{3f}$$

$$f_{ij}^s, g_{ij}^s \ge 0 \qquad \forall i,j,s \in N, \tag{3g}$$

$$m_{ij}^t \in \{0,1\} \qquad \forall (i,j) \in O_t \quad \forall t$$

5 Results

In this section, we will present the computational results. We had done our computations by using Python's Gurobi interface. In the first set of results, our primary aim was to show that our computation times are better than [Fenz et al., 2019]'s times in every instance. Hence, we did not hold any parameters constant, and we displayed results in different number of nodes to exhibit how the CPU times change with respect to the increase in number of nodes, and each case our times are significantly lower. Since we limited the computation time to one and a half an hour, maximum number of nodes used is 15. We had done other calculations with 12 and 14 nodes. To display the effect of different densities in demand matrix, static and optic topology, we presented three instances with different densities for each number of nodes. In the second set of runs, we did not varied the number of nodes, whereas we had changed static topology density, optic topology density, and demand matrix density in the preceding order while holding the other parameters constant. In that section we aimed to show the effect of only the changing densities. Similarly, we presented three instances for each varied parameter. Colored columns are varied parameters while different colors refers to the change in constant variables.

In total, we had made the calculations 27 times, 1 for each density combinations of 0.3, 0.6, and 0.9, for each number of nodes in each set of runs. However, we did not include all the results in this reports not to overwhelm the reader, nevertheless, we could provide all set of results to the reader upon request. The results we provided below are the most extreme ones, in terms of both good and bad. Also, first GPU column is for the times of our model, whereas second column is for [Fenz et al., 2019]' model. We run both of the models with the same computer which runs on Apple M1 Chip with 8-Core CPU and 8 GB unified memory. Computation times are calculated within Gurobi solver.

5.1 First Run Results

In the first set of runs, a new set of data is created by the random generator for each run with different densities. GPU times are in seconds.

Table 1: Run Results

Static	Optic	Demand	# of nodes	Our CPU	CPU [Fenz et al., 2019]
.3	.3	.3	15	32.18	632.85
.9	.3	.9	15	2.37	112.74
.9	.3	.6	15	30.99	4876.51
.6	.3	.6	14	5.92	226.74
.9	.3	.3	14	5.02	241.55
.6	.3	.9	14	3.16	62.85
.6	.3	.3	12	2.54	44.02
.3	.3	.6	12	0.03	0.27
.9	.9	.3	12	3.16	62.85

5.2 Second Run Results

In the first part of the second set of runs, the same data set is used for optic topology and demand matrix, only the static topology is changed.

Table 2: Second run results with static density is the independent variable

Static	Optic	Demand	# of nodes	Our CPU	CPU [Fenz et al., 2019]
.3	.6	.6	12	0.3	14.51
.6	.6	.6	12	0.31	3.05
.9	.6	.6	12	0.31	7.87
.3	.3	.3	12	1.93	53.29
.6	.3	.3	12	1.96	47.3
.9	.3	.3	12	1.53	43.45
.3	.9	.3	12	0.04	0.35
.6	.9	.3	12	0.03	0.38
.9	.9	.3	12	0.02	0.4

In the second part of the second set of runs, the same data set is used for static topology and demand matrix, only the optic topology is changed.

Table 3: Second run results with optic density is the independent variable

Static	Optic	Demand	# of nodes	Our CPU	CPU [Fenz et al., 2019]
.3	.3	.6	12	1.61	24.44
.3	.6	.6	12	0.19	0.71
.3	.9	.6	12	0.03	0.14
.6	.3	.9	12	0.27	2.41
.6	.6	.9	12	0.08	0.36
.6	.9	.9	12	0.03	0.1
.9	.3	.3	12	1.36	46.11
.9	.6	.3	12	0.3	6.17
.9	.9	.3	12	0.03	0.31

In the third part of the second set of runs, the same data set is used for static and optic topologies, only the demand matrix is changed.

Table 4: Second run results with demand density is the independent variable

Static	Optic	Demand	# of nodes	Our CPU	CPU [Fenz et al., 2019]
.3	.3	.3	12	2.4	56.69
.3	.3	.6	12	1.82	30.83
.3	.3	.9	12	0.22	1.94
.6	.9	.3	12	0.03	0.69
.6	.9	.6	12	0.02	0.11
.6	.9	.9	12	0.02	0.1
.9	.3	.3	12	2.25	53.88
.9	.3	.6	12	1.31	42.03
.9	.3	.9	12	0.87	5.77

6 Heuristic Algorithms

In this section we provided two of the heuristic algorithms that we are planning to improve in the future research. Moreover, to compare them with different approaches, we coded these algorithms by using Python's Gurobi Interface.

6.1 Demand First Algorithm

Algorithm 1 Demand First Algorithm

Data: Graph N, Demand Matrix D
Result: A graph N with the newly created links.
1. Sort the demand entries in D by size in descending order.
while D *is not empty OR all matchings have not been created*
do
(a) Pick the first entry in D.
(b) Run ReconfigDijkstra to route this demand.
(c) Update N by making used matchings permanent, delete the current demand from D.
end

6.2 Gain Demand Algorithm

Algorithm 2 Gain Demand Algorithm

Data: Graph N, Demand Matrix D
Result: A graph N with the newly created links.
1. Initialize an empty list called $gains$.
for *Every entry in* D
do
(a) Run ReconfigDijkstra on N, add the value of the objective function in the updated N to value.
(b) Reset N to its initial state.
end
2. Sort $gains$ by size in increasing order.
while D *is not empty OR all matchings have not been created*
do
(a) Pick the first entry in $gains$.
(b) Run ReconfigDijkstra to route this demand.
(c) Update N by making used matchings permanent, delete the current demand from $gains$.
end

7 Conclusion

In this paper, we first provided two novel mathematical models for data center design and routing. The first one was an extension of the [Fenz et al., 2019]'s model which we were aiming to bring about a less complexity and lower run time comparing to the current model. Our computations show that our attempt is successful as we have displayed in the Results section. Secondly, [Fenz et al., 2019]'s model was for one reconfigurable switches, in our second model, we provided a more general model for multiple reconfigurable switches which is also valid for one switch.

As we have shown in the Results part, our model runs with a much lower computation time in every instance we have run the model. We run it with different configurations where demand matrix, static and optic topology densities has been taken with different values such as 0.3, 0.6, 0.9. Since the complexity of the model is NP-Hard as shown by [Foerster et al., 2018], solving the exact model in a considerably shorter time gives us the advantage to solve large instances of the model. In that aspect, our model is much more practical to solve large networks, though we could not find the chance to run our model in large instances yet. We are planning to run our model by using one of the real life data center data.

Moreover, since the complexity of the problem does not allow us to find the optimal solutions in reasonable times for large instances, we are bound to use heuristic approaches. We first review the four heuristic approaches proposed by [Fenz et al., 2019], the first two of which we inquired in details and came up with the counter examples where these heuristic approaches does not yield the optimal solutions even with 7 nodes instances. Having refuted the already proposed heuristics in terms of their accuracy, we are planning to emerge a new heuristic approach that is better than the current ones in terms of accuracy and computation time.

When it comes to the design part, how many reconfigurable switches should be used or how dense the static or optic topology should be are crucial questions. Having gotten the answers of these questions could help data center design enormously in terms of cost and flow time. Our computation results provided us a small perspective on the matter. According to our results, when the demand matrix is too dense or too light, the computation time decreases. Although the latter one is not a surprise, the former is because when demand goes to nearly everywhere from everywhere, the model tries to put optical connections to the bottleneck traffic points. Whereas when the demand density is moderate, in our instance it corresponds to 0.6, the computation time increases significantly. On the other hand, the density of the static topology makes the smallest change in the CPU times, and expected CPU times decreases scarcely with the increasing density. There is a greater change in the computation times when the optic topology is increased as can be seen in our second part of the

second run results. When the network has scarce number of optic connections, the decision of where to put the matchings becomes more vital, however, when virtually every connection can be made optical, the decision depends only on the demand amounts. Thus, dense optic topology results in a considerably lower CPU time. As said, these are only initial remarks and they may lack the complete accuracy. Therefore, we are planning to make sensitivity analysis to find the optimal densities for demand matrix, and static and optic densities. We believe this knowledge would benefit the design of the future data centers as it would yield the most effective routing when completed with the models we provided.

References

[Bienkowski et al., 2021] Bienkowski, M., Fuchssteiner, D., Marcinkowski, J., and Schmid, S. (2021). Online dynamic b-matching: With applications to reconfigurable datacenter networks. *SIGMETRICS Perform. Eval. Rev.*, 48(3):99–108.

[Fenz et al., 2019] Fenz, T., Foerster, K.-T., Schmid, S., and Villedieu, A. (2019). Efficient non-segregated routing for reconfigurable demand-aware networks. In *2019 IFIP Networking Conference (IFIP Networking)*, pages 1–9.

[Foerster et al., 2018] Foerster, K.-T., Ghobadi, M., and Schmid, S. (2018). Characterizing the algorithmic complexity of reconfigurable data center architectures. In *Proceedings of the 2018 Symposium on Architectures for Networking and Communications Systems*, ANCS '18, page 89–96, New York, NY, USA. Association for Computing Machinery.

[Foerster et al., 2019] Foerster, K.-T., Pacut, M., and Schmid, S. (2019). On the complexity of non-segregated routing in reconfigurable data center architectures. *SIGCOMM Comput. Commun. Rev.*, 49(2):2–8.

[Foerster and Schmid, 2019] Foerster, K.-T. and Schmid, S. (2019). Survey of reconfigurable data center networks: Enablers, algorithms, complexity. volume 50, page 62–79, New York, NY, USA. Association for Computing Machinery.

[Luo et al., 2019] Luo, L., Foerster, K.-T., Schmid, S., and Yu, H. (2019). Dartree: Deadline-aware multicast transfers in reconfigurable wide-area networks. In *2019 IEEE/ACM 27th International Symposium on Quality of Service (IWQoS)*, pages 1–10.

[Luo et al., 2020] Luo, L., Foerster, K.-T., Schmid, S., and Yu, H. (2020). Splitcast: Optimizing multicast flows in reconfigurable datacenter networks. In *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, pages 2559–2568.

[Pal and Kant, 2015] Pal, A. and Kant, K. (2015). Roda: A reconfigurable optical data center network architecture. In *2015 IEEE 40th Conference on Local Computer Networks (LCN)*, pages 561–569.

[Xue et al., 2018] Xue, X., Yan, F., Pan, B., and Calabretta, N. (2018). Flexibility assessment of the reconfigurable opsquare for virtualized data center networks under realistic traffics. In *2018 European Conference on Optical Communication (ECOC)*, pages 1–3.

[Yang et al., 2019] Yang, Z., Cui, Y., Xiao, S., Wang, X., Li, M., Li, C., and Liu, Y. (2019). Achieving efficient routing in reconfigurable dcns. *Proc. ACM Meas. Anal. Comput. Syst.*, 3(3).