2024 IEEE 25th International Conference on High Performance Switching and Routing (HPSR) | 979-8-3503-6385-2/24/$31.00 ©2024 IEEE | DOI: 10.1109/HPSR62440.2024.10635940

LASER: Learning Enhanced Segment Routing Using Link Telemetry

Murali Kodialam, T.V. Lakshman
Nokia Bell Labs

I. Abstract

This paper presents a novel approach to telemetry-based routing, aiming to minimize network congestion using multiple link load measurements collected at different points in time at a centralized management system. The objective is to determine a *fixed routing policy* that optimizes network performance by minimizing the maximum link utilization for any traffic matrix that could have generated any link load measurement. The key idea is to develop a routing mechanism that has the flexibility to handle a wide variety of traffic conditions without reconfiguration. We use a routing mechanism called *deflection routing* and develop a machine learning based gradient algorithm (LASER) to compute the deflection routing parameters. We use a combination of variable transformation and Lagrangian based techniques to transform the parameter optimization problem into an unconstrained loss minimization problem which is solved using a structured neural network in the PyTorch framework.

II. Introduction

Recently there has been renewed interest in finding in efficient routing solutions for traffic engineering data centers and WANS [1], [2]. This is due to the fact that networks are being run at higher utilizations and the traffic incident on the network changes in an unpredictable manner. The main thrust in [1], [2] is to use historical data to develop routing solutions. In general, networks continuously monitor the link loads and collect this information in a central location like a SDN controller [3], [4]. This process of measuring the link utilization and collecting it at a central location is commonly referred to as *network telemetry*. A set of all link loads at a given time will be referred to as a *snapshot.*. Since the number of links in a network is typically far less than the number of node pairs, there is no unique mapping from the link utilization to the point to point traffic matrix corresponding to a given snapshot. In fact, there are an infinite number of traffic matrices that could have potentially generated an observed snapshot. If the routing has to be modified to reduce network congestion, then this modification has to done without knowledge of the point to point traffic matrix. In this paper, we address the problem of designing routing techniques that use multiple historically generated snapshots or predicted snapshots to develop a routing scheme. Our objective is to develop

- A routing scheme that is flexible enough to handle traffic generated by any one of multiple snapshots without requiring any configuration changes.
- An algorithm to compute the configuration parameters to minimize network congestion for any traffic matrix that could have generated any one of the multiple snapshots.

In order to meet our objective, we have to develop a new routing that will reduce network congestion for *any* of the (infinite) traffic matrices that could have generated the observed link load. The fact that there are routing schemes that work well for a wide range of traffic matrices is implied by the vast literature in oblivious routing [5], [6], [7], which originated with the seminal work of Räcke [8]. There are however several practical problems with implementing these solutions that has not lead to widespread adoption of these schemes in practice. Moreover, oblivious routing guarantees relative worst case performance for any traffic matrix but what we want is an absolute guarantee for any traffic matrix that could have generated the observed link load. We refer to the routing mechanism that we use in this paper as *Deflection Routing* where a flow is routed from source to destination along at most two shortest path segments. This routing mechanism is very simple to implement using segment routing. The number of parameters to define a deflection routing scheme grows with the size of the network and the number of constraints increase with the number of snapshots. Traditional optimization approaches do not scale with the size of the networks. Deriving the optimal deflection routing parameters is very amenable to machine learning techniques. We develop a structured neural net based approach (LASER) to derive the optimal deflection routing parameters for a set of measured link loads.

A. Our Contributions

The LASER algorithm represents a novel approach to addressing routing problems by leveraging multiple snapshots. The developed scheme is simple to implement while offering computable performance guarantees. It is important to underscore that our routing scheme establishes an upper limit on link utilization, applicable to any traffic matrix within the observed set of link utilizations. To address the challenge of handling multiple snapshots, we propose a machine learning-driven approach. Our methodology leverages a gradient descent algorithm resembling a neural network, (Structured Neural Network (SNN)) customized to tackle the specific problem at hand. This approach allows for the derivation of optimal configuration parameters, a facet of our work that is of independent interest for other optimization problems.

III. Related Work

Network routing has been the subject of study for several decades and there is a large body of work that addresses different versions of network routing. Network routing can be broadly divided into centralized and distributed routing [6]. With the widespread introduction of the Software Defined Networking (SDN) paradigm, the SDN controller has a view of the whole network and it is possible to deploy powerful centralized algorithms for routing traffic [9]. With the development of machine learning based optimization techniques, there has been renewed interest in network routing using these newly developed techniques for routing traffic in a SDN based network [3]. The most commonly used machine learning based technique for routing traffic is reinforcement learning [10]. However, more recently, the problems with using reinforcement based learning techniques for routing has been discussed in [11]. The main criticism for using reinforcement learning based techniques for routing is that suffers from higher data-sample complexity, sensitivity to noisy training, and hyperparameter tuning problems [12]. Therefore, there has been recent effort [11] to develop prediction free methods for traffic engineering. In this line of work, a deep neural network is used find paths for each source destination pair based on training using the historical traffic matrix data. Our work differs from this line of research in two important ways. First we do not assume any explicit knowledge of the traffic matrix. The only information available to our algorithm are multiple snapshots of network link utilizations. This is a far more difficult problem then routing with historical traffic matrices. Secondly, we want to use a mechanism that is robust to traffic changes without constant reconfiguration. Papers in literature including [11] permit frequent network reconfguration. This makes existing approaches difficult to implement in practice. The routing mechanism that we use is *deflection routing*. Deflection routing has been studied in the literature as $\mathcal{SR}_2$ [13] and waypoint routing [14]. The approach that we use is to configure the deflection routing parameters that are matched to the given snapshots so that network congestion can be minimized for any of the traffic matrices that could have generated the snapshots. None of the papers in the literature can handle multiple snapshots due to the size of the optimization problems. The SNN based approach developed in this paper exploits the fast gradient descent framework to solve problems of practical size.

IV. Model and Assumptions

We consider a network with n nodes and m links, where periodic link load measurements are collected centrally. The capacity of link ℓ is denoted by $c(\ell)$ and the link weight metric is $w(\ell)$. The link weight metric is used to compute the shortest path. We refer to one complete set of link utilization measurement as a snapshot. In the multiple snapshot problem, we get T sets of link level measurement. We use $\rho(\ell, t)$ denote the measured utilization of link ℓ in snapshot t.

A. Congestion Measure

The metric that we use to measure network congestion is the maximum link utilization in the network. This is the most commonly used metric in literature but our method can be extended to other metrics like average link utilization.

B. Segment Routing

We assume that segment routing [15] is implemented in the network. Segment Routing is a routing protocol that simplifies and enhances the way packets are routed in a network. It introduces the concept of "segments," which are specific nodes or links in the network that act as waypoints for packet forwarding. The basic idea is to assign a unique identifier (segment ID) to each network segment. When a packet enters the network, it carries its destination and a list of segment IDs that represent the desired path. As the packet traverses the network, each router looks at the next segment ID and forwards the packet to the corresponding segment. This process continues until the packet reaches its final destination. The routing employed on each hop follows the Equal-Cost Multipath (ECMP) strategy, with $\beta_{ij}(\ell)$ representing the flow in link ℓ when one unit of flow is sent from node i to node j.

C. Deflection Routing

In deflection routing, packets are routed from source to destination using at most two segments. In other words, packets are routed directly from source to destination along the shortest path, or is routed from source to destination using one deflection through an intermediate node. Each segment is a minhop route with ECMP. Figure 1, shows a packet from node i to node j that is deflected through intermediate node k. Shortest path routing is a special case of deflection routing. If we route on the shortest path from source i to destination j, then this is equivalent to setting $\alpha_{ij}^{j} = 1$ or $\alpha_{ij}^{i} = 1$. If 60%

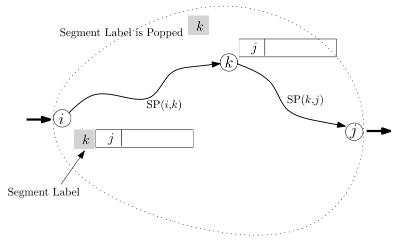

Fig. 1. Flow from node i to node j is deflected through intermediate node k.

of the traffic from i to j is deflected through node k_1 and 40% is deflected through node k_2 it is easy to implement this policy by hashing on the packet header and with pre-pending the appropriate intermediate node. We use

$$\delta_{ij}^{k}(\ell) = \beta_{ik}(\ell) + \beta_{kj}(\ell)$$

to represent the flow on link ℓ if traffic from source i to destination j is deflected through node k. The objective in the multiple snapshot problem, is to devise a deflection routing scheme that minimizes the maximum link utilization for *any* traffic matrix that could have generated *any* of the snapshots.

V. LASER: Learning to Segment Route with Multiple Snapshots

In the multiple snapshot problem, there are T snapshots where each snapshot gives a complete set of link utilization measurements. We index the snapshots with t and $\rho(\ell, t)$ is the utilization of link ℓ in snapshot t. The traffic matrix incident on the network during each snapshot could be different. The (unknown) traffic between nodes i and j in snapshot t is denoted by d_{ij}^t. We assume that deflection routing is used in each time slot (shortest path routing is a special case of deflection routing) and α_{ij}^{kt} represents the fraction of traffic from node i to node j that is deflected through node k during the snapshot t. We use $\mathcal{P}(\alpha, \mathbf{g}, \rho, t)$ to represent the polyhedron

$$\frac{1}{c(\ell)} \sum_i \sum_j \sum_k \delta_{ij}^k(\ell)\ \alpha_{ij}^k\ d_{ij}^t = \rho(\ell, t) \quad \forall\ \ell$$

which comprises of all traffic matrices that could have generated the link utilizations observed in snapshot t. The objective of the multiple snapshot problem is to determine a deflection routing scheme that minimizes the maximum link utilization for any traffic traffic matrix that could have generated the observed link load during any of the snapshots. This is pictorially shown in Figure 2. Note that each snapshot could have been generated by a potentially infinite number of traffic matrices.

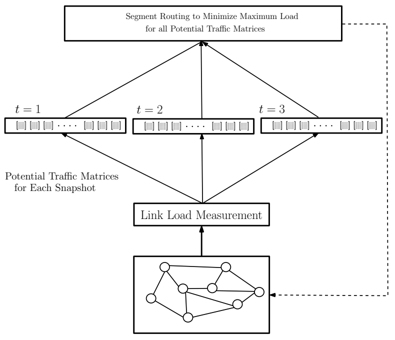
Segment Routing to Minimize Maximum Load
for all Potential Traffic Matrices
$t = 1$
$t = 2$
$t = 3$
Potential Traffic Matrices
for Each Snapshot
Link Load Measurement

Fig. 2. High Level View of LASER with Multiple Snapshots

A. Our Approach

The outline of our approach is as follows:

- Given a link ℓ', we first formulate the optimization problem of determining the traffic matrix in $\mathcal{P}(\alpha, \mathbf{g}, \rho, t)$ that maximizes the load on link ℓ'
- Different traffic matrices in $\mathcal{P}(\alpha, \mathbf{g}, \rho, t)$ might maximize the flow on link ℓ' for different links ℓ'
- Use the dual to these optimization problems to find the routing that minimizes the maximum link load for *any* traffic matrix in $\mathcal{P}(\alpha, \mathbf{g}, \rho, t)$
- Using strong duality, the dual problems corresponding to all links are consolidated into a single optimization problem.
- This optimization problem is re-formulated into a unconstrained problem which is solved by a structured neural network using stochastic gradient descent.

The optimization problem for determining the optimal deflection routing x_{ij}^k that minimizes the maximum link utilization for any traffic matrix that could have generated any of the snapshots is written using the dual approach. *Note that the optimal deflection routing x_{ij}^k does not have a t index, meaning that the deflection routing is not only independent of any traffic matrix in a given snapshot but also independent of any snapshot.* The problem $OPT(\ell', t)$ to determine the worst case traffic matrix for snapshot t that maximizes the utilization of a given link ℓ' can be written as

$$\max\ \frac{1}{c(\ell')} \sum_i \sum_j \sum_k \delta_{ij}^k(\ell') x_{ij}^k\ d_{ij}^t$$

subject to:

$$\frac{1}{c(\ell)} \sum_i \sum_j \sum_k \delta_{ij}^k(\ell)\ \alpha_{ij}^k\ d_{ij}^t = \rho(\ell, t) \quad \forall\ \ell$$

$$d_{ij}^t \geq 0 \quad \forall i, j, t$$

The variable is the traffic matrix d_{ij}^t, and note that x_{ij}^k is independent of the traffic matrix d_{ij}^t.

B. Dual to the Maximization Problem on Link ℓ'

Associating dual variables $\pi(\ell, \ell', t)$ with the link measurement constraint on link ℓ for the optimization problem $OPT(\ell', t)$, the dual can be written as:

$$\min \sum_\ell \rho(\ell, t)\pi(\ell, \ell', t)$$

subject to:

$$\sum_\ell \frac{1}{c(\ell)} \sum_k \delta_{ij}^k(\ell)\alpha_{ij}^{kt}\ \pi(\ell, \ell', t) \leq \frac{1}{c(\ell')} \sum_k \delta_{ij}^k(\ell')x_{ij}^k \quad \forall i, j, \ell', t$$

By strong duality [16], the optimal solution value of the primal and dual problems are equal. Note that there is one linear programming problem for each link ℓ'.

C. Putting All These Problems Together

The objective function of the dual is $\min \sum_\ell L(\ell)\pi(\ell, \ell', t)$ and this is equal to the primal objective function which is the maximum utilization of link ℓ'. We want to minimize the maximum link utilization. Therefore, the overall objective of the routing problem is to

$$\min \max_{\ell', t} \sum_\ell \rho(\ell, t)\pi(\ell, \ell', t)$$

$$\begin{aligned} \sum_\ell \frac{1}{c(\ell)} \sum_k \delta_{ij}^k(\ell)\alpha_{ij}^{kt}\ \pi(\ell, \ell', t) \quad &\leq \quad \frac{1}{c(\ell')} \sum_k \delta_{ij}^k(\ell')x_{ij}^k \quad \forall i, j, \ell', t \\ \sum_k x_{ij}^k \quad &= \quad 1 \quad \forall i, j \\ x_{ij}^k \quad &\geq \quad 0 \quad \forall i, j \end{aligned}$$

The last two constraints, ensure the deflection routing parameters are feasible.

D. LASER: A Structured Neural Net Based Approach

The min-max objective function can be re-formulated into a minimization problem. The number of variables in the problem is $O(n^3+m^2)$ and the number of constraints is $O(n^2m)$. This makes the problem quite large even for small networks. With the recent improvements in using gradient descent to solve optimization problems with billions of variables [17], [18], we leverage this framework to solve the problem of determining the optimal deflection routing scheme. The optimization problem has to be re-formulated to fit in this framework. We call the gradient descent based technique for solving this problem a Structured Neural Network (SNN) based approach. Instead of a standard multi-layered neural network, we use a neural network whose structure is tailored to the problem on hand. It is possible to include practical constraints like restricted splitting as a soft constraint in a SNN formulation. Moreover, in the case of multiple snapshot problem, it is easy to use batching to solve the problem for small batches of snapshot chosen at random (or cyclically) for each iteration. This approach leverages the recent improvements in the execution speed of gradient descent based approaches for problems involving millions to billions of variables. However, gradient descent based approaches work well on unconstrained optimization problems. Constraints can be handled in a gradient descent [19] by projecting the gradient into the feasible set but this slows the convergence process as well as increases the time required per iteration. Therefore, we use a combination of variable reformulation and Lagrangian based approach to reformulate this problem as an unconstrained optimization problem.

E. Reformulating as a SNN Computation

Corresponding to the deflection routing variable x_{ij}^k we define an unconstrained set of variables y_{ij}^k and set

$$x_{ij}^k = \frac{e^{\alpha y_{ij}^k}}{\sum_p e^{\alpha y_{ij}^p}} \tag{1}$$

where $\alpha > 0$ is a parameter. This function is the commonly used Softmax function in the machine learning literature. By this transformation, note that $x_{ij}^k \geq 0$ and $\sum_k x_{ij}^k = 1$ for all i, j. The value of α has to be chosen to ensure that network congestion is minimized without excessive traffic splitting. Note that as the value of α increases, the Softmax function becomes steep and there is a tendency for the values of x_{ij}^k to be close to zero or one. This is not guaranteed since the restricted splitting problem is NP-hard but larger values of α generally results in a fewer x_{ij}^k values being greater than zero. In all the experiments we set of value of $\alpha = 10$. Therefore, the optimization problem can be reformulated as

$$\min \max_{\ell',t} \sum_{\ell} \rho(\ell,t)\pi(\ell,\ell',t)$$

$$\sum_{\ell} \frac{1}{c(\ell)} \sum_k \delta_{ij}^k(\ell)\alpha_{ij}^{kt}\, \pi(\ell,\ell',t) \leq \frac{1}{c(\ell')} \sum_k \delta_{ij}^k(\ell') \left[\frac{e^{\alpha x_{ij}^k}}{\sum_p e^{\alpha x_{ij}^p}}\right] \quad \forall i,j,\ell'$$

In order to organize the computation as an SNN and use solve the problem using gradient descent in a PyTorch framework we organize the computation of the optimal solution as follows: We define

$$\Omega(\ell',t) = \sum_{\ell} \rho(\ell,t)\pi(\ell,\ell',t)$$

to denote the maximum link utilization of a given link ℓ' and the objective is to find a deflection routing that minimizes

$$M = \max_{\ell',t} \Omega(\ell',t).$$

We now use a Lagrangian Relaxation based approach to relax the constraints and include it into the objective constraint. In order to do this we define a non-negative Lagrange multiplier λ and use this multiplier to penalize any constraint violation. We can see if there is a constraint violation by computing

$$\begin{aligned} \Phi_{ij}(\ell',t) &= \sum_{\ell} \frac{1}{c(\ell)} \sum_k \delta_{ij}^k(\ell)\alpha_{ij}^{kt}\, \pi(\ell,\ell',t) \\ \Delta_{ij}(\ell') &= \frac{1}{c(\ell')} \sum_k \delta_{ij}^k(\ell') \left[\frac{e^{\alpha y_{ij}^k}}{\sum_p e^{\alpha y_{ij}^p}}\right] \end{aligned}$$

which are the left and right hand side of the constraint set for a given triple i, j, ℓ', t. There is a constraint violation if $\Phi_{ij}(\ell',t) - \Delta_{ij}(\ell') > 0$ for any i, j, ℓ', t. In the SNN, we use the ReLU function (ReLU is x if $x > 0$ and 0 if $x \leq 0$) to capture this constraint violation. We use $ReLU\,(\Phi_{ij}(\ell',t) - \Delta_{ij}(\ell'))$ to capture the constraint violation for given i, j, ℓ', t. Therefore the total penalty

$$P = \sum_{i,j,\ell',t} ReLU\Big(\Phi_{ij}(\ell',t) - \Delta_{ij}(\ell')\Big).$$

The penalty $P > 0$ if and only if there is a constraint violation. We now define the loss function of the SNN to be $L = M + \lambda P$. The value of λ has to be chosen high enough to ensure that the constraint is satisfied. It is also possible to have a different λ for each constraint and aggregate the penalty in the loss function. We find that having a single λ to penalize the sum of the constraint violations is quite effective and also makes controlling the penalty quite simple. We want to set λ as low as possible to ensure that the inequality constraint is satisfied. The value of λ is initialized to one and increased linearly to $\lambda = 100$. Note that there are no additional constraints since the $\pi(\ell,\ell')$ is unconstrained We use gradient descent to solve this optimization problem. The overall organization of the structured neural network is shown in Figure 3. In this figure, there are three sets of variables:

- The variables in red $\Big(y_{ij}^k, \pi(\ell,\ell',t)\Big)$ are the decision variables
- The variables in purple $\Big(\rho(\ell,t), \beta_{ij}(\ell), \alpha_{ij}^{kt}\Big)$ are input variables

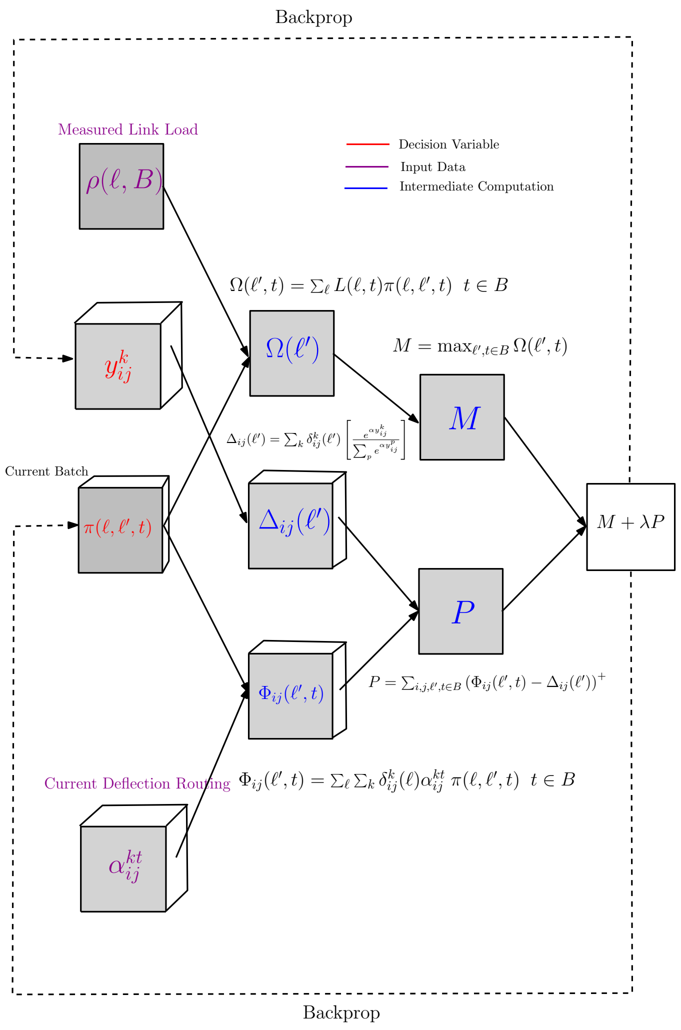
Backprop
Measured Link Load
Decision Variable
Input Data
Intermediate Computation
$\rho(\ell, B)$
$\Omega(\ell', t) = \sum_\ell L(\ell,t)\pi(\ell,\ell',t) \;\; t \in B$
y_{ij}^k
$\Omega(\ell')$
$M = \max_{\ell', t \in B} \Omega(\ell', t)$
M
$\Delta_{ij}(\ell') = \sum_k \delta_{ij}^k(\ell') \left[\frac{e^{\alpha y_{ij}^k}}{\sum_p e^{\alpha y_{ij}^p}}\right]$
Current Batch
$\pi(\ell,\ell',t)$
$\Delta_{ij}(\ell')$
$M + \lambda P$
P
$\Phi_{ij}(\ell', t)$
$P = \sum_{i,j,\ell',t \in B} (\Phi_{ij}(\ell',t) - \Delta_{ij}(\ell'))^+$
Current Deflection Routing
$\Phi_{ij}(\ell',t) = \sum_\ell \sum_k \delta_{ij}^k(\ell)\alpha_{ij}^{kt}\, \pi(\ell,\ell',t) \;\; t \in B$
α_{ij}^{kt}
Backprop

Fig. 3. Structured Neural Network for LASER

- The variables in blue are intermediate computational results

All the input variables are initialized by sampling from a $U[0, 1]$ distribution. This is then input to the SNN where gradient descent with back-propagation is used to solve the problem of determining the deflection routing parameters that minimizes the maximum link utilization for any traffic matrix that could have generated the observed link utilization.

F. Batch SGD

For the multiple snapshot problem, it is generally faster to solve the problem where in each iteration a subset of the snapshots is used. As in standard neural network training algorithms, in each iteration of gradient descent, we only perform descent on a subset of the snapshots. When a batch size B is specified, in each iteration, we either randomly pick B out of T snapshots and use perform gradient descent on these snapshots. The other option is to cycle through the T snapshots in groups of B. The performance of both these approaches is similar. When using a small batch size, the running time per batch is lower but convergence takes longer. If the number of snapshots is not too large (about 10 or so), then it is better run each iteration on all the snapshots. If the number of snapshots is greater than 10 then the algorithm converges faster when using a batch size of 10.

G. Experimental Results

The first set of experiments is on a random topology of 30 nodes and 120 links and is shown in Figure 4. The second set of experiments is on topologies from the Topology Zoo [20]. Since we consider fixed routing schemes (which do not change with varying traffic patterns) we compare the performance of LASER with shortest path routing. Shortest path routing is the most commonly used static routing scheme. As stated in Section III, we are not aware of any existing algorithms in literature that can handle multiple network utilization snapshots. In the first set of experiments, we run experiments

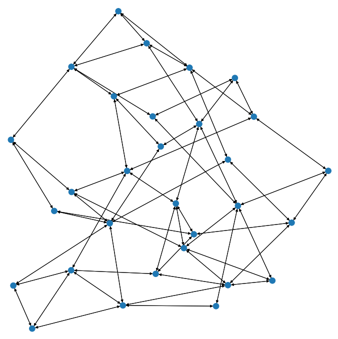

Fig. 4. Network with 30 nodes and 120 links

on the network shown in Figure 4 with 10 snapshots. Each snapshot is generated by generating a traffic matrix and routing it using shortest path routing. We compare the performance of shortest path routing with deflection routing. We use batch gradient descent with a batch size $B = 10$. Figure 5 shows the maximum link utilization for shortest path routing when routing each of the traffic matrices. Each color represents the performance of one snapshot. Figure 6 shows the performance of LASER. The red line is the worst case performance of LASER. Note that the maximum link utilization for shortest path routing over all the traffic matrices is about 1.5 and the worst case link utilization for deflection routing is about 0.85. In Figure 6 the colored lines represents the link utilization for each of the snapshots.

In the second set of experiments, we compare the performance of LASER with shortest path routing on 5 network topologies from the Topology Zoo [20]. The traffic matrices are generated at random and routed on the network using shortest path routing and link load measurements are generated. We generate between 10 and 100 snapshots. The results are shown in Table I. Note that LASER outperforms shortest path routing in all cases. For the experiments where there were 100 snapshots, we perform batch gradient descent with a batch

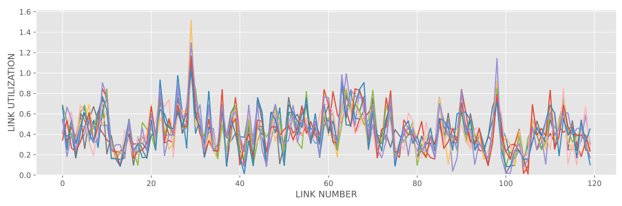

Fig. 5. Multiple Snapshot: Shortest Path Performance

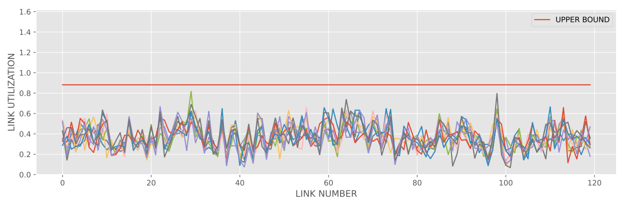

Fig. 6. Multiple Snapshot: Deflection Routing Performance

size of 10. We have performed experiments with uniform and non-uniform link capacities and the performance of LASER is significantly better than shortest path routing. The only cases where the performance of shortest path routing matches LASER is when the topology has limited connectivity (like a tree) and there are no alternative shortest paths to balance the traffic. Since most deployed topologies are at least two-connected, we expect to LASER to outperfrom shortest path routing in all practical topologies.

TABLE I
COMPARISON OF LASER AND SP FOR NETWORKS FROM TOPOLOGY ZOO

Nodes	Links	Traffic Matrices	SP Util	LASER Util
51	132	10	1.02	0.79
125	300	10	1.20	0.86
30	110	10	1.30	0.90
113	366	100	1.10	0.82
24	74	100	1.10	0.89

VI. CONCLUSION

This paper introduces an innovative approach to telemetry-driven routing, with the overarching goal of mitigating network congestion through the utilization of multiple link load measurements obtained at various intervals by a centralized management system. Our objective is to establish a fixed routing policy that enhances network efficiency by minimizing the maximum link utilization across diverse traffic scenarios, leveraging the flexibility to adapt to varying conditions without necessitating frequent reconfiguration. Central to our methodology is the deployment of deflection routing, a robust routing mechanism. We further introduce the LASER algorithm, a gradient-based machine learning approach, to compute the parameters crucial for deflection routing. By employing a combination of variable transformation techniques and Lagrangian methods, we effectively reframe the parameter optimization task as an unconstrained loss minimization problem. This transformed problem is adeptly addressed using a structured neural network implemented within the PyTorch framework.

REFERENCES

[1] Y. Perry, F. V. Frujeri, C. Hoch, S. Kandula, I. Menache, M. Schapira, and A. Tamar, "DOTE: Rethinking (predictive) WAN traffic engineering," in *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*. Boston, MA: USENIX Association, Apr. 2023, pp. 1557–1581. [Online]. Available: https://www.usenix.org/conference/nsdi23/presentation/perry

[2] Z. Xu, F. Y. Yan, R. Singh, J. T. Chiu, A. M. Rush, and M. Yu, "Teal: Learning-accelerated optimization of wan traffic engineering," in *Proceedings of the ACM SIGCOMM 2023 Conference*, 2023, pp. 378–393.

[3] R. Amin, E. Rojas, A. Aqdus, S. Ramzan, D. Casillas-Perez, and J. M. Arco, "A survey on machine learning techniques for routing optimization in sdn," *IEEE Access*, vol. 9, pp. 104 582–104 611, 2021.

[4] M. Polverini, A. Baiocchi, A. Cianfrani, A. Iacovazzi, and M. Listanti, "The power of sdn to improve the estimation of the isp traffic matrix through the flow spread concept," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 6, pp. 1904–1913, 2016.

[5] H. Räcke, "Survey on oblivious routing strategies," in *Mathematical Theory and Computational Practice: 5th Conference on Computability in Europe, CiE 2009, Heidelberg, Germany, July 19-24, 2009. Proceedings 5*. Springer, 2009, pp. 419–429.

[6] G. Rétvári and G. Németh, "Demand-oblivious routing: distributed vs. centralized approaches," in *2010 Proceedings IEEE INFOCOM*. IEEE, 2010, pp. 1–9.

[7] B. Haeupler, H. Räcke, and M. Ghaffari, "Hop-constrained expander decompositions, oblivious routing, and distributed universal optimality," in *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, 2022, pp. 1325–1338.

[8] H. Räcke, "Optimal hierarchical decompositions for congestion minimization in networks," in *Proceedings of the fortieth annual ACM symposium on Theory of computing*, 2008, pp. 255–264.

[9] I. F. Akyildiz, A. Lee, P. Wang, M. Luo, and W. Chou, "A roadmap for traffic engineering in sdn-openflow networks," *Computer Networks*, vol. 71, pp. 1–30, 2014.

[10] Z. Mammeri, "Reinforcement learning based routing in networks: Review and classification of approaches," *Ieee Access*, vol. 7, pp. 55 916–55 950, 2019.

[11] Y. Perry, F. V. Frujeri, C. Hoch, S. Kandula, I. Menache, M. Schapira, and A. Tamar, "{DOTE}: Rethinking (predictive){WAN} traffic engineering," in *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*, 2023, pp. 1557–1581.

[12] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, "Deep reinforcement learning that matters," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[13] R. Bhatia, F. Hao, M. Kodialam, and T. Lakshman, "Optimized network traffic engineering using segment routing," in *2015 IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2015, pp. 657–665.

[14] S. A. Amiri, K.-T. Foerster, R. Jacob, and S. Schmid, "Charting the algorithmic complexity of waypoint routing," *ACM SIGCOMM Computer Communication Review*, vol. 48, no. 1, pp. 42–48, 2018.

[15] C. Filsfils, S. Previdi, L. Ginsberg, B. Decraene, S. Litkowski, and R. Shakir, "Segment Routing Architecture," RFC 8402, Jul. 2018. [Online]. Available: https://www.rfc-editor.org/info/rfc8402

[16] M. S. Bazaraa, J. J. Jarvis, and H. D. Sherali, *Linear programming and network flows*. John Wiley & Sons, 2011.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[18] T. B. e. a. Brown, "Language models are unsupervised multitask learners," *arXiv preprint arXiv:2005.14165*, 2020.

[19] S. K. Roy and M. Harandi, "Constrained stochastic gradient descent: The good practice," in *2017 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*. IEEE, 2017, pp. 1–8.

[20] S. Knight, H. Nguyen, N. Falkner, R. Bowden, and M. Roughan, "The internet topology zoo," *Selected Areas in Communications, IEEE Journal on*, vol. 29, pp. 1765 – 1775, 11 2011.