Towards AI/ML-Driven Network Traffic Engineering

Quazi Mishkatul Alam
Amazon Web Services
Seattle, WA, USA
qalam@amazon.com

Vinay Kolar
Amazon Web Services
San Jose, CA, USA
vinkolar@amazon.com

Marina Thottan
Amazon Web Services
New York, NY, United States
mthottan@amazon.com

Abstract

Internet is one of the largest scale distributed system made up of multiple networks that is used to digitally connect billions of users. Traffic Engineering (TE) is a core problem in networking, which is responsible for routing packets across networks to provide the best user experience while ensuring a secure, stable, well-utilized and cost-efficient network. The time-varying graph nature of the network along with unexpected topology and traffic changes makes TE in large operational networks challenging. This paper first provides a holistic sketch of different viewpoints taken by researchers and practitioners to formulate and solve the TE problem. We first cover the *systems view*, where researches have defined the problem and used creative heuristic protocols to manage large networks. We then focus on the theoretical formulations from *optimization and control theory* to provide optimal and stable networks. These formulations provide clear definitions and provable properties for designing TE. We devise a taxonomy of existing studies on how such theoretical problems are being solved today. Finally, we present the *AI/ML and heuristic* perspective for near-optimal TE. Owing to the dynamism and large scale of the networks, especially planet-scale cloud networks, these theoretical methods need real-time data and greedy approximations to solve TE. Recent progress in AI/ML provides encouraging tools here, where time-series and graph based AI/ML models can be used to detect/predict the network state, and control algorithms such as Reinforcement Learning (RL) be powerful tools to solve TE. We survey these approaches and propose promising directions towards AI/ML in network control and TE.

ACM Reference Format:
Quazi Mishkatul Alam, Vinay Kolar, and Marina Thottan. 2024. Towards AI/ML-Driven Network Traffic Engineering. In *4th International Conference on AI-ML Systems (AIMLSystems 2024), October 08–11, 2024, Baton Rouge, LA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3703412.3703436

1 Introduction

Computer networking is one of the largest distributed systems today that provide the backbone for digital connectivity. Operating almost invisibly, networking enables everything from simple web browsing to Internet of Things (IoT). It employs diverse technologies such as optical fibers, copper, cellular, satellite networks, and a large system of protocols to connect users to their requested applications, usually hosted in large data-centers or servers.

Artificial Intelligence and Machine Learning (AI/ML) provides promising tools to optimize networks. Each network measures performance metrics such as utilization, memory, congestion, loss and latency, across each network node and link. In addition, logs at various nodes provide valuable information about the state of each node. The TE system therefore has to learn across different data modalities in this network graph. There are dynamic topological (graph) learning aspects, such as various nodes connected to each other. There are time-series and distribution effects of networking performance metrics between nodes, and congestion on these nodes, that determine the connectivity experience. Finally, log data is another important source to analyze what is happening across the network nodes. AI/ML provides a promising methodology to decipher patterns across systems and data-modalities, and autonomously alert or even autonomously fix the network.

Certain areas of networking have used AI/ML to learn and act, while there is a wide open opportunity for its application in other areas [2, 4, 53]. A majority of AI/ML applications use anomaly detection, fault localization and root-causing algorithms for networking or security metrics, often using time-series and graph modalities [33, 42, 67]. However, the success of having standard models or neural network architectures, deriving insights and taking automated actions using AI/ML is arguably underwhelming in the networking domain when compared to other large systems such as vision or language based systems. The main reasons can be attributed to: (a) dynamism of traffic and topology changes on large networks, which creates unexpected and noisy data to learn from; (b) local changes that lead to global effects across other parts of the network, thus making it hard to pinpoint without jointly analyzing network graph, time-series of metrics and logs; and (c) issues about data availability and readiness because of non-standardized data being owned by multiple network operators [10]. Hence, it is common to see individual use cases of AI/ML that uses local observation (e.g., monitoring network link between nodes, congestion on the device) for alerting or making local decisions, but not a coordinated holistic solution – like routing across a large operational network.

Traffic Engineering is one of the primary areas where such measurements and coordination across the network is vital. Traffic Engineering (TE) is responsible for routing packets across an operational network such that users get the best experience and the network is stable, secure, well-utilized and cost-efficient. TE is especially essential for planet-scale operational networks to ensure that traffic demands are constantly well-placed while reacting to network degradation, topology changes or unexpected traffic demand surges. Researchers and practitioners are solving TE using various approaches, ranging from development of creative protocols using system heuristics to formulating a control theoretical or AI/ML

AIMLSystems 2024, October 08–11, 2024, Baton Rouge, LA, USA
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-1161-9/24/10
https://doi.org/10.1145/3703412.3703436

models to provide near-optimal TE. This paper surveys the general problem of network control and TE from three viewpoints.
1. We summarize the *systems view* on how the current protocols are regulating the network based on different distributed and quasi-centralized network architectures. However, these system protocols do not clearly define the optimization problem that they are solving and are not guaranteed to provide best or stable routing.
2. We review studies which focus on *optimality and control mechanisms.* Here, optimization theory is used to formulate how the network should behave. Some studies extend this by using control theory to design feedback mechanisms that provide optimal and stable network configurations. Centralized network architectures provide means of running a global optimization problem. However, it is computationally infeasible to run in large networks such as planet-scale cloud backbone.
3. Finally, we discuss how *AI/ML* provides tools to solve the TE problem to provide near-optimal control mechanisms. AI algorithms for modeling time-varying network graphs and control algorithms, such as Reinforcement Learning (RL), can be used for learning the state of the network and routing flows near-optimally in a large network.

We first review the systems view and then provide motivation for AI/ML in Section 2. We provide a generalized optimal TE model and review optimality and control theory work in Section 3. We then present a taxonomy of studies that propose and solve optimal TE in Section 4. Finally, we discuss AI/ML based approaches in solving optimal TE in Section 5, and conclude in Section 6.

2 TE Systems and AI/ML Motivation

We discuss the background about network routing and TE, and then present the opportunity and challenges for using AI/ML.

2.1 Routing and Traffic Engineering

Large operational networks today are planet-scale. For example, AWS network spans across 34 regions, 108 Availability Zones and 600+ points of presence [6]. Network operators manage these networks to cater to varying traffic demands by designing good network architectures, and provisioning capacity at devices and links. An important function of this network involves measuring performance, and controlling traffic routing that satisfy various demands across the network such that users get the best experience and network routing is efficient and stable; this is Traffic Engineering (TE) [8, 44]. There are different approaches and architectures for routing and TE, which we categorize and describe below.

Distributed Shortest Path Routing: A straight-forward approach is to route packets within a single owner network (called Autonomous System or AS) using variations of shortest-path routing protocols. These protocols can be proprietary, but usually, networks employ well known Intra-AS protocols such as Interior Gateway Protocols (IGP). Many IGP protocols, such as Open Shortest Path First (OSPF), make local routing decisions on a per-packet basis using distributed algorithms. Network routers (nodes in a graph) advertise their connectivity (links) to all other nodes by flooding link-state advertisements that propagate the edge weight for the link based on various performance and capacity metrics. Each node stores the link-state, and use a shortest path algorithm, like Dijkstra's algorithm, to route the packet.

A disadvantage of this approach is that the traffic is concentrated only on shortest paths. Any peaks in traffic will induce congestion on hot-spot links and nodes, resulting in network degradation – even when other parts of the network have capacity. Such hot spots degrade network availability, and will require over-provisioning. But, this is not scalable since high capacity needs to be added only at few hot spots in the network.

Distributed Traffic Spreading: The next-generation of TE used traffic spreading protocols such as Multi-Protocol Label Switching with TE (MPLS TE) [9]. Often, large networks use MPLS TE based mechanisms to optimize the traffic across the backbone to ensure availability and best performance [47, 49]. These protocols enable an ingress router to spread the traffic over multiple paths across the network, thus making the network more: (a) scalable by spreading the traffic across multiple areas of the network; and (b) available by routing via backup paths when the primary path breaks down.

MPLS TE enabled routers spread traffic on various paths, with some additional control logic for coordinating announcement and path changes on a network. Large networks typically design routing such that traffic can be sent via the best paths (e.g., low latency), such that the network is well utilized and there is no single point of failure. There are several additional mechanisms for fast reroute to handle network failures in few milliseconds by adjusting the active paths in the network [7].

Return of centralized controllers: Recently, centralized controllers have become popular in large networks, since this enables a controller to take optimal routing decisions based on a global network view. Software Defined Networking (SDNs) is one such architecture [25, 39, 57]. SDNs enabled a completely programmable network which decoupled two main functionalities:
1. Control plane: This defines the control decisions taken on how to route (e.g., computing routes between each pair of nodes). This final set of policies and paths is then disseminated to all the routers.
2. Data plane: Each router then takes a decision on a per-packet basis on how to forward packets based on the policies received from the control plane.

This separation of duties enabled having centralized SDN controllers which have a global view rather than distributed protocols such as MPLS. The idea of a centralized controller and its responsibility to find the best routes across the network is attractive, since this enables a single point where AI/ML can be used to understand data and take actions. There are variations of SDNs in large cloud networks, which use multiple controllers so that the network can be scaled. However, each controller still has a fairly large view – rather than a router-level view – to compute best performing paths.

2.2 Opportunity and challenges for using AI/ML

The data and network complexity make AI/ML promising tools to solve TE. Data has to be analyzed across different modalities, and then the entire network has to take action (control element) so that the network performs as intended. The data modalities usually consist of graphs, time-series and logs that need to be collectively analyzed across the network because local effects can lead to global changes. For example, changes to the topology graph of the network combined with the time-series of network metrics are constantly varying – sometimes at a millisecond granularity. Logs from routers and subsystems provide additional information to determine local changes at these network elements. In addition, there is a control element where feedback within the network can be used to understand network perturbations, and act at fine time granularities to get the network back to near-optimal. This rich set of time-varying data across a network topology graph where controls can be applied for near-optimal operation presents a fertile ground to solve the problem near-optimally using AI/ML.

While the use of AI/ML is encouraging, there are two primary challenges to overcome to effectively use AI/ML for TE:

1. Local change, global effects: Local effects, such as congestion on one link, can cause the TE protocol to widely shift traffic in a network that causes global effects on other flows across a different part of the network [50]. Hence, unlike image data or even sequence-to-sequence data such as language or time-series, it is challenging to predict global actions across the network based on observing time-varying network graph.

2. Dynamism: The network often has to place varying traffic demands from multiple users on fixed capacity links. In large cloud networks, traffic demand spikes cannot always be predicted. For example, cloud customers hosting popular games or shows can cause large traffic spikes in particular geo-regions when many clients access these resources at the same time. This often cannot be predicted since a cloud network hosts multiple customers, and each customer may be releasing content or having other network-heavy operations, like backups, based on a schedule that is unknown to the cloud. The network has to dynamically adjust to such loads.

3 Optimal TE formulation

We first compare system and theoretical approaches towards TE. We then describe approaches to model optimal TE, and describe the key optimization and control models applied in this field.

System protocols, explained in Section 2.1, are creative system ideas that balance and weigh several metrics for effective network operation, and have experimentally shown to be effective in running today's large networks. However, these system heuristics do not clearly define and optimize protocol behavior such as: (a) what is the objective that the system as-a-whole is trying to optimize?, (b) is the system performing in the optimal way, and if not, what is the optimality gap?, (c) does the solution converge under various dynamic traffic and topology changes? Answering such questions is important because it is uncertain if the intended network behavior is being delivered by the operational network. More importantly, there can be unintended effects which were not captured by experimentally determined system parameters; such conditions could pose a risk for catastrophic network events.

We now generalize the important aspects of the optimal TE problem to discuss research papers that analyze system protocols and build better routing and TE.

We represent the network as a graph $G(\mathcal{N}, \mathcal{E})$ where $\mathcal{N}$ denotes the set of nodes (routers) and $\mathcal{E}$ represents the set of edges (direct link or configured routing segments) between the nodes that can communicate. Let C_{ij} be the capacity for edge (i, j); this can be the bandwidth of the link (e.g., 10 Gbps). We assume that there are several flows that need to be routed from a source (ingress) router to a destination (egress) router. For a given aggregate flow k, let (s_k, d_k, t_k) denote the source, destination and expected demand (throughput). If the demand is known apriori, then t_k is a constant. In some use cases, networks are optimized to provide maximum throughput for all applications. In such cases t_k is a variable that is to be maximized. Let $\mathcal{F}$ be the set of flows that are using the network. Let $x_{k,ij}$ be the capacity dedicated by link (i, j) to carry traffic for flow k. Two trivial constraints are that these flow variables $x_{k,ij}$ should be non-negative, and that the total flow on the link should be less than its capacity C_{ij}.

The other main constraint is *flow conservation constraint*, which provides a constraint to route t_k units from source to destination. Routing has to ascertain that whatever the source emits is consumed by the destination, i.e., : (a) only the source s_k emits the traffic exactly equal to the demand t_k for flow k; and (b) only destination d_k consumes the traffic exactly equal to t_k; (c) none of the intermediate nodes generate any extra traffic for flow k.

Utility function: The objective function is formulated based on the utility to either end-applications or network. These functions are called as *utility function.*

Hence, these class of problems are named as Network Utility Maximization (NUM) problems [38, 46, 56]. The objective function generally takes the form of: Maximize $\sum_{k \in \mathcal{F}} U(k)$.
The most common variables in utility functions are the network performance metrics, which may consist of metrics such as minimizing congestion or latency or maximizing throughput or fairness.

Utility functions can be classified into two types: (a) Flow or end-user utility; (b) Network utility. Flow utility maximizes the utility across all the end flows. Examples of end-user utility are minimizing the end-to-end latency/loss or maximizing the flow rate (throughput) since they directly benefit the flows/applications/users. However, there are other formulations that maximize overall network utility, such as minimizing the cost of running the network or spreading the traffic. Spreading the traffic is often achieved via minimizing the maximum link utilization [51, 63], or some function that allows for maximum residual capacity across edges.

Certain studies optimize multiple metrics by using techniques such as weighted objective that considers multiple parameters such as minimizing maximum link utilization and also minimizing latency [51]. In addition to performance metrics, we can also formulate metrics that improve network performance, such as decrease the cost or guarantee stability. Stability of routes intend to avoid route switching when demands or topology changes dynamically within certain bounds. There are several attractive properties that emerge out of NUM formulations, which we will discuss in Section 4.2.

Flow Splitting: A major decision in formulating the TE problem is the decision to allow flow splitting. Flow splitting allows a router to split a single flow fractionally across its multiple outgoing links. If we allow the flow to be split infinitesimally at every node, the model is referred to as a *fluid model.* Such problems can be formulated as a Convex Optimization problem, including Linear Programming (LP). The advantage of these problems are that they are polynomial time solvable. Despite this, in large cloud networks, the large number of constraints and variables makes it practically impossible to run these models periodically [66]. For example, CAIDA's AS-dataset with a relatively small network size of around 2000 nodes, 9000 edges and 3 million flows has over 25 billion constraints [15, 66]. Today's large cloud networks will be much larger than this network.

The second case is to restrict a flow to take a single or few pre-specified paths. This is helpful in networks where there are large flows that should not be split because of networking effects such as packet reordering. Such formulation converts the problem into a Multi-Commodity Flow (MCF), which is Mixed Integer Linear Programming (MILP) and NP-hard [3].

4 Optimal TE Taxonomy

We present a taxonomy of solutions for optimal TE based on the structure of the problem, as shown in Figure 1. A summary of research papers and its categorization is in Table 1, with separate AI/ML studies in Table 2.

The first two categories in the taxonomy *Network Type* and *Protocol Type* are network architecture centric. The solutions either solve for Cloud [13, 16, 28, 57, 66] or Internet Service Provider (ISP) network type [22, 37, 43, 59, 63]. The protocol type implies the type of protocols used for TE, which includes either plain IP protocols, MPLS [11, 14, 22, 47, 52] or SDN [28, 29, 34, 57]. We do not describe specific papers here under these categories, as our focus is on generalized optimal TE formulation and AI/ML solutions.

4.1 Controller properties

This describes the classification based on what kind of controller is being used in the network for which the optimal TE is formulated. The network controller can be distinguished in the following ways:
1. Distributedness: Controllers can be distributed, centralized or hybrid. Distributed controllers are usually deployed in IP and MPLS type of networks where there is no centralized point for controller [30]. Several control-theory based NUM formulation based protocols such as FAST TCP [26, 64], explained in Section 3, are instances of distributed network control derived from an optimal formulation. Centralized controllers are most amenable to solving optimization since there is a global view of the network at a central point [28, 48, 66]. As explained in Section 2.1, architectures such as SDNs have central controllers where such algorithms can reside. The third type is the hybrid scheme where there are centralized subcontrollers, which optimize parts of the network, and there are distributed protocols or other hierarchical controllers that optimize across these centralized subcontrollers. Very large networks such as planet-scale clouds often have this scheme where there is a need to coordinate across multiple subcontrollers that optimize different geo-regions of the network. Optimizing TE for such hybrid networks is an open area of research.
2. Control Domain: TE can optimize a within a network domain (intra-domain) or inter-domain. A large number of optimal traffic engineering work focuses on intra-domain because ISPs, cloud and enterprise networks would first optimize traffic within their own networks [30, 35, 61]. These network owners can run any proprietary protocols within their network, even though most networks run well-known protocols such as OSPF.

Inter-domain routing optimizes routes across different networks – typically operated by different owners. Several system protocols have been proposed to optimize inter-domain TE [19, 54], but there are not many optimal formulations. While the optimal TE formulations and heuristics can be theoretically extended for inter-domain routing, there are challenges: (a) lack of transparency because individual network owners often implement proprietary policies, and the data across networks is not shared; and (b) there is a general agreement to use standardized protocols, such as Border Gateway Protocol (BGP), for inter-domain routing so that any network can talk to any other network; this limits the implementation scope of the control mechanisms.
3. Transmission: In networks, a node can transmit packets from one node to another (unicast transmission), or can transmit the same information to multiple nodes (multicast). A vast majority of the Internet and optimal TE formulations focus on unicast transmission. However, there are certain networks that support specialized applications that have to optimize for multicast [11, 60]. There are also optimal TE for special types of communication modes, such as multiple nodes to single node [51].

4.2 Optimizer Properties

The studies can be classified based on how the optimal formulation is trained/executed and the approximations used.
1. Training mode: The training can be offline or online. Offline models are run or trained either periodically at a large time-granularity (e.g., once a day) and the resulting model is used by setting the output routing for the rest of the period [11, 37, 51, 58, 63, 66], without changing the model parameters. However, offline methods are not reactive to the real-time network conditions such as topology, performance, traffic demands, or routing table changes. Online algorithms, where the model parameters are updated at a fine-time granularity, are hence more useful in running network operations in large networks [14, 22, 28, 35, 57].
2. Layering: The other interesting dimension is the layer at which the model operates. Many models provide the output for path-computation, where the model emits the routes to be taken, and other rules or protocols configure these paths. There are a different set of algorithms that operate cross-layer and provide multiple network functionalities such as joint rate-control and scheduling.

There are noteworthy contributions from control theory here on how to derive optimal protocols [38, 46, 56]. They not only propose NUM formulations for optimal network control, but also go further by using duality theory for designing optimal and stable cross-layered algorithms such as rate-control and scheduling [17, 46, 56].

The studies prove that joint cross-layered optimization such as rate-control (employed at transport layer for each ingress router or application source) and scheduling (running at routing layer on each router) can converge to optimal and has provable robustness properties. These works have reverse engineered distributed TCP/IP congestion control protocols starting from an optimal problem, and show the stability of the TCP/IP type congestion control – despite feedback delays – in the Internet. FAST TCP [26, 64] developed for optimizing long distance and high latency links was motivated by decomposition of a primal NUM problem.
3. Algorithm: Some problems can be solved by solvers such as LP, MILP or Convex Optimization Solvers [18, 24, 31, 45]. However, as we discussed in Section 3, even smaller networks with around 2000 nodes can have billions of constraints when modelled as a graph

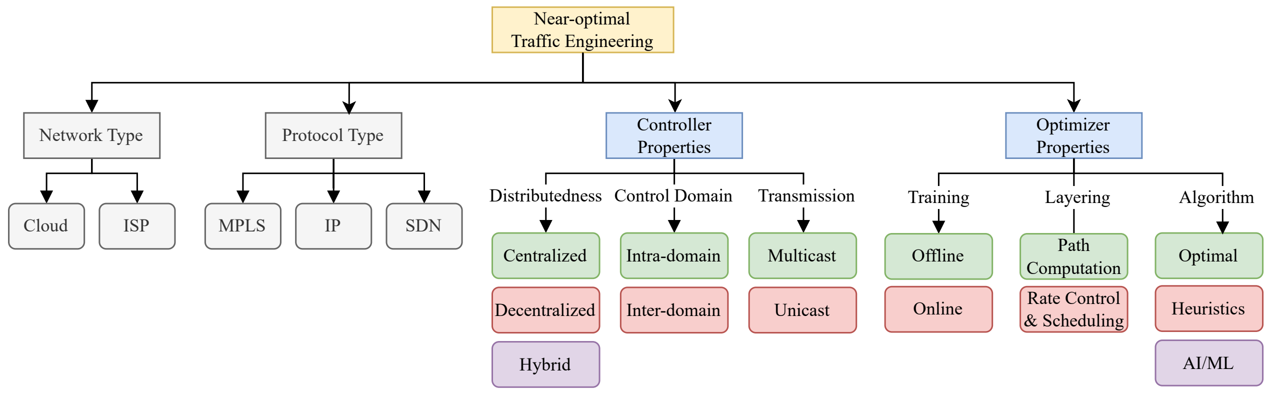

Figure 1: Taxonomy of Optimal TE

to solve optimal TE. Hence, researchers have devised hand-crafted approximation heuristics that provide near-optimal solutions[22, 35, 62]. For example, TeXCP formulates the optimization problem to minimize maximum link utilization, and then proposes a distributed online heuristic [35]. They instantiate agents on multiple nodes that use heuristics to reduce oscillations in the network while being reactive to network changes. This is another area where AI/ML models can act as fast blackbox inference engines that provide heuristic optimal TE; we discuss this in the next section.

5 AI and ML Techniques

Section 2.2 discussed two main uses of AI/ML: ability to handle different data modalities, and modeling network as a graph to capture non-local effects. In addition to these, AI/ML is also useful for learning: (a) the state of the network from various forms of data; and (b) how to control the network using feedback mechanisms to reach near-optimal and stable TE.

1. Learning the state of the network: Operational networks need to understand the network topology, metrics and demands from different data sources such as active/passive measurements, topology databases and time-varying traffic demands. The graph, time-series and log modalities of data can be scanned together to observe the state of the nodes and edges in the network. This enables solving optimal TE given the properties of the time-evolving network graph, using architectures such as GNN.

For example, one study predicts path failures in the network using GNN to capture a snapshot of the network at a given time, and the temporal dynamics of the nodes and edges using Long Short Term Memory (LSTM) [27] to form a time-evolving neural network called Long Short-Term Memory R-GCN (LRGCN) [42]. However, solving with real-time metrics of network is more challenging than evaluation on static datasets, since it is vital for operators and downstream software components to act based on real-time data to fix network degradation observed by customers. This is an open area of research.

2. Learning to control in real-time: Studies employ Reinforcement Learning (RL) to learn how to control routing, in addition to learning the state of the network [5, 23, 48, 65]. They usually simulate multiple scenarios using multi-agent centralized RL where each agent is assigned to find paths for a flow, and then each agent gets the same reward based on collective network state. Such reward mechanism enables observing global effects because of local agent's routing action, and collectively rewarding all agents towards overall network optimality.

The GDDR paper proposes a GNN based routing by modeling the network using a GNN and using RL [30]. TEAL is another interesting study that proposes a different variety of graph formulation for optimizing large networks [66]. They first precompute up to four possible assigned paths for each flow, and then create a new graph called FlowGNN which is a bipartite graph with paths on one set (called path nodes) and the edges of the original network (called edge nodes) on the other set. This FlowGNN then utilizes RL to learn how to split traffic along the four paths for a flow.

A more ambitious challenge and an open area of research is to learn control mechanisms in real-time operating networks in an online fashion. This will open a valuable research area where control theory meets online optimization of large operational networks that not only optimize the network, but also can react and stabilize the network in real-time.

6 Conclusions

Traffic Engineering (TE) in networks is responsible for routing decisions to provide the best user/application experience while assuring a stable, secure and cost-efficient network. Solving this problem involves learning the network topology, traffic demands and routing decisions in a time-varying and highly dynamic environment. We study this area from three perspectives: (a) systems, (b) designing optimal control mechanisms using optimization and control theory, and (c) solving these optimal problems using AI/ML to approximate computationally complex optimal models. We start from a theoretical formulation and then explain heuristics employed to solve for large operational networks. We present a taxonomy of existing work based on network- and formulation-centric properties. We then categorize the studies based on different varieties of the problem being solved and approximations. Finally, we identify open research topics for the application of AI/ML towards building a system for realizing near-optimal routing and TE.

Title of the Paper	Optimization Objective	Categories
Explicit Routing Algorithms for Internet Traffic Engineering (1999) [63]	Minimizing the maximum of link utilization across all links	ISP, MPLS, Centralized, Unicast, Offline, Optimal, Path-computation
Traffic Engineering using Multiple Multipoint-to-Point LSPs (2000) [51]	Minimize the number of LSP routes while ensuring availability, and reduce maximum link utilization on paths with a limited hop count	ISP, MPLS, Centralized, Multicast, Offline, Optimal, Path-computation
Minimum Interference Routing of Bandwidth Guaranteed Tunnels with MPLS TE Applications (2000) [37]	Minimize LSP interference, maximize future accommodations	ISP, MPLS, Decentralized, Unicast, Offline, Heuristics, Path-computation
Engineering the Multi-Service Internet: MPLS and IP-based Techniques (2001) [59]	Minimize LSP bandwidth allocation, Satisfy QoS, distribute load, minimize overall cost	ISP, IP/MPLS, Centralized, Unicast, Offline, Heuristics, Path-computation + Dynamic
MATE: MPLS Adaptive Traffic Engineering (2001) [22]	Provide traffic distribution among preconstructed static LSPs, probing and load-balancing, minimize total cost	ISP, MPLS, Decentralized, Unicast, Online, Heuristics, Path-computation
DORA: Efficient Routing for MPLS Traffic Engineering (2002) [14]	Maximize future traffic accommodation by minimizing interference and maximizing residual bandwidth	MPLS, Unicast, Online, Heuristics, Dynamic
SPeCRA: A Stochastic Performance Comparison Routing Algorithm (2002) [20]	Minimize the rejection ratio of LSP setup requests, minimize interference	MPLS, Unicast, Online, Heuristics, Dynamic
Quality of Service Provisioning through Traffic Engineering (2003) [58]	Minimize network load (own definition link utilization) after satisfying QoS constraints	ISP, DiffServ/MPLS, Offline, Optimal, Path-computation
Creating Multipoint-to-Point LSPs for Traffic Engineering (2003) [11]	Minimize the number of M-t-p trees, optimal merging	ISP, MPLS, Centralized, Multicast, Offline + Online, Heuristics, Path-computation
A Preemption-Aware On-line Routing Algorithm for MPLS Networks (2003) [12]	Maximize network performance (utilization) by preemption and provides stability	ISP, MPLS, Decentralized, Online, Heuristics
TEAM: A Traffic Engineering Automated Manager for DiffServ-Based MPLS Networks (2004) [52]	Maximize network performance (utilization) by preemption and provides stability	ISP, DiffServ/MPLS, Centralized, Online, Heuristics
New Preemption Policies for DiffServ-Aware Traffic Engineering (2004) [21]	Maximize network performance by minimizing rerouting	ISP, MPLS, Heuristics
Walking the Tightrope: Responsive Yet Stable Traffic Engineering (2005) [35]	Minimize the maximum utilization without creating oscillations	Intra-domain, IP, Decentralized, Online, Heuristics
COPE: Traffic Engineering in Dynamic Networks (2006) [62]	Optimize performance while providing worst case guarantee	ISP, Heuristics
Latency Inflation with MPLS-Based Traffic Engineering (2011) [47]	Suggests guidelines to alleviate latency inflation in service networks	Cloud, MPLS, Heuristics
Achieving High Utilization with Software-Driven WAN (2013) [28]	Maximize throughput while choosing low-latency paths, min-max fairness	Cloud, SDN, Centralized, Online, Heuristics
B4: Experience with a globally-deployed software defined WAN (2013) [57]	Maximize link utilization/throughput and ensure min-max fairness across priority classes	Cloud, SDN, Centralized, Online
Traffic Engineering with Forward Fault Correction (2014) [43]	Incorporate fault tolerance by proactively handling up to k simultaneous faults to maintain TE allocations	ISP, Centralized, Online, Heuristics
Calendaring for Wide Area Networks (2014) [36]	Optimize large, long-term transfer scheduling based on pre-assigned deadlines to maximize throughput	Cloud, Centralized, Online, Heuristics
Optimizing Bulk Transfers with Software-Defined Optical WAN (2016) [34]	Achieve higher throughputs by rapid reconfiguration of optical topology for bulk transfers	Cloud, ISP, SDN, Centralized, Online, Heuristics
Dynamic Pricing and Traffic Engineering for Timely InterDatacenter Transfers (2016) [32]	Dynamically price and schedule bandwidth transfers based on deadlines and priority to optimize costs	Cloud, SDN, Centralized, Online, Heuristics
Taking the Edge off with Espresso: Scale, Reliability and Programmability for Global Internet Peering (2017) [68]	Optimize client performance at the edge by moving packet processing to software	Cloud, ISP, SDN, Centralized, Unicast, Online, Heuristics
RADWAN: Rate Adaptive Wide Area Network (2018) [55]	Maximize bandwidth while minimizing reconfiguration churn on optical links	Cloud, IP, Centralized, Online
B4 and After (2018) [29]	Scale and manage hierarchical SDN to meet availability and traffic demands	Cloud, SDN, Centralized-decentralized, Hierarchical, Online
Semi-oblivious Traffic Engineering: The Road Not Taken (2018) [41]	Improve load balancing and link failure resilience using oblivious routing	Cloud/ISP, Centralized, Online
TEAVAR: Striking the Right Utilization-availability Balance in WAN Traffic Engineering (2019) [13]	Optimize WAN traffic considering probabilistic link failures to avoid overprovisioning capacity	Cloud, SDN, Centralized, Offline
Lancet: Better Network Resilience by Designing for Pruned Failure Sets (2019) [16]	Design network resilience strategies for specific failure sets	SDN, Offline + Online, Heuristics
Cost-effective Cloud Edge Traffic Engineering with Cascara (2021) [54]	Minimize inter-domain bandwidth costs using latency-equivalent peer links	Inter-domain, Cloud, Centralized, SDN, Offline, Heuristics
Contracting Wide-area Network Topologies to Solve Flow Problems Quickly (2021) [1]	Scale traffic engineering by contracting network paths	SDN, Offline, Heuristics
CodedBulk: Inter-datacenter Bulk Transfers using Network Coding (2021) [60]	Enable high-throughput inter-datacenter bulk transfers using network coding	Cloud, TCP/MPLS, Multicast, Optimal
Shoofly: Cost-effective Capacity Provisioning in Wide Area Networks with Shoofly (2021) [54]	Optimize WAN topology by identifying cost-effective optical bypasses	Cloud, Centralized, Offline
Decentralized Cloud Wide-area Network Traffic Engineering with BLASTSHIELD (2022) [40]	Enhance fault tolerance in centralized cloud TE by partitioning the WAN into slices	Cloud, Decentralized, Online

Table 1: Summary of Traffic Engineering Papers

Name of the Paper	Optimization Objective	Categories
Learning to Route (2017) [61]	Learn effective routing configurations using machine learning	Intra-domain, ISP, IP, Online, Machine Learning
Experience-driven Networking: A Deep Reinforcement Learning based Approach (2018) [65]	Implement a model-free control framework using deep reinforcement learning (DRL)	SDN, Centralized, Online, Machine Learning
CFR-RL: Traffic Engineering With Reinforcement Learning in SDN (2020) [69]	Use reinforcement learning to select and reroute critical flows in SDN	SDN, Centralized, Online, Machine Learning, Reinforcement Learning
Towards Real-time Routing Optimization with Deep Reinforcement Learning: Open Challenges (2021) [5]	Optimize real-time routing using deep reinforcement learning	Online, Machine Learning
GDDR: GNN-based Data-driven Routing (2021) [30]	Apply GNN-based data-driven routing for enhanced network performance	Intra-domain, Decentralized, Online, Machine Learning, Reinforcement Learning
TEAL: Learning-accelerated Optimization of WAN Traffic Engineering (2023) [66]	Accelerate WAN traffic engineering optimization through learning techniques	Cloud, Centralized, Online, Machine Learning, Reinforcement Learning
DOTE: Rethinking (Predictive) WAN Traffic Engineering (2023) [48]	Rethink WAN traffic engineering with predictive and machine learning approaches	Centralized, Online, Machine Learning
Traffic Engineering in a Shared Inter-DC WAN via Deep Reinforcement Learning (2023) [23]	Optimizes inter-DC and internet traffic using MILP formulation and RL techniques with real-world traces	Cloud, Centralized, Online, Machine Learning, Reinforcement Learning

Table 2: Summary of AI/ML Traffic Engineering Papers

References

[1] Firas Abuzaid, Srikanth Kandula, Behnaz Arzani, Ishai Menache, Matei Zaharia, and Peter Bailis. 2021. Contracting wide-area network topologies to solve flow problems quickly. In *18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*. 175–200.

[2] Ijaz Ahmad, Shariar Shahabuddin, Hassan Malik, Erkki Harjula, Teemu Leppänen, Lauri Lovén, Antti Anttonen, Ali Hassan Sodhro, Muhammad Mahtab Alam, Markku Juntti, Antti Ylä-Jääski, Thilo Sauter, Andrei Gurtov, Mika Ylianttila, and Jukka Riekki. 2020. Machine Learning Meets Communication Networks: Current Trends and Future Challenges. *IEEE Access* 8 (2020), 223418–223460. https://doi.org/10.1109/ACCESS.2020.3041765

[3] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. 1993. *Network flows: theory, algorithms, and applications*. Prentice-Hall, Inc., USA.

[4] Mohammed Ali Al-Garadi, Amr Mohamed, Abdulla Khalid Al-Ali, Xiaojiang Du, Ihsan Ali, and Mohsen Guizani. 2020. A Survey of Machine and Deep Learning Methods for Internet of Things (IoT) Security. *IEEE Communications Surveys & Tutorials* 22, 3 (2020), 1646–1685. https://doi.org/10.1109/COMST.2020.2988293

[5] Paul Almasan, José Suárez-Varela, Bo Wu, Shihan Xiao, Pere Barlet-Ros, and Albert Cabellos-Aparicio. 2021. Towards real-time routing optimization with deep reinforcement learning: Open challenges. In *2021 IEEE 22nd International Conference on High Performance Switching and Routing (HPSR)*. IEEE, 1–6.

[6] Amazon Web Service. 2024. *AWS Global Infrastructure*. https://aws.amazon.com/about-aws/global-infrastructure/

[7] Alia Atlas, George Swallow, and Ping Pan. 2005. Fast Reroute Extensions to RSVP-TE for LSP Tunnels. RFC 4090. https://doi.org/10.17487/RFC4090

[8] Daniel O Awduche. 1999. MPLS and traffic engineering in IP networks. *IEEE communications Magazine* 37, 12 (1999), 42–47.

[9] Daniel O. Awduche, Lou Berger, Der-Hwa Gan, Tony Li, Dr. Vijay Srinivasan, and George Swallow. 2001. RSVP-TE: Extensions to RSVP for LSP Tunnels. RFC 3209. https://doi.org/10.17487/RFC3209

[10] Roman Beltiukov, Wenbo Guo, Arpit Gupta, and Walter Willinger. 2023. In Search of netUnicorn: A Data-Collection Platform to Develop Generalizable ML Models for Network Security Problems. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security* (Copenhagen, Denmark) *(CCS '23)*. Association for Computing Machinery, New York, NY, USA, 2217–2231. https://doi.org/10.1145/3576915.3623075

[11] Sudeept Bhatnagar, Samrat Ganguly, and Badri Nath. 2003. Creating Multipoint-to-Point LSPs for traffic engineering. In *Workshop on High Performance Switching and Routing, 2003, HPSR*. IEEE, 201–207.

[12] Francois Blanchy, Laurent Mélon, and Guy Leduc. 2003. A preemption-aware on-line routing algorithm for MPLS networks. *Telecommunication Systems* 24 (2003), 187–206.

[13] Jeremy Bogle, Nikhil Bhatia, Manya Ghobadi, Ishai Menache, Nikolaj Bjørner, Asaf Valadarsky, and Michael Schapira. 2019. TEAVAR: striking the right utilization-availability balance in WAN traffic engineering. In *Proceedings of the ACM Special Interest Group on Data Communication*. 29–43.

[14] Raouf Boutaba, Wayne Szeto, and Youssef Iraqi. 2002. DORA: Efficient routing for MPLS traffic engineering. *Journal of Network and Systems Management* 10 (2002), 309–325.

[15] CAIDA. 2024. *The CAIDA UCSD AS Relationships*. https://www.caida.org/catalog/datasets/as-relationships/

[16] Yiyang Chang, Chuan Jiang, Ashish Chandra, Sanjay Rao, and Mohit Tawarmalani. 2019. Lancet: Better network resilience by designing for pruned failure sets. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 3, 3 (2019), 1–26.

[17] Mung Chiang, Steven H. Low, A. Robert Calderbank, and John C. Doyle. 2007. Layering as Optimization Decomposition: A Mathematical Theory of Network Architectures. *Proc. IEEE* 95, 1 (2007), 255–312. https://doi.org/10.1109/JPROC.2006.887322

[18] CVX Research, Inc. 2024. *CVX*. https://cvxr.com/

[19] Sukrit Dasgupta, Jaudelice C. De Oliveira, and Jean-Philippe Vasseur. 2007. Path-Computation-Element-Based Architecture for Interdomain MPLS/GMPLS Traffic Engineering: Overview and Performance. *IEEE Network* 21, 4 (2007), 38–45. https://doi.org/10.1109/MNET.2007.386468

[20] Jaudelice Cavalcante de Oliveira, Francesco Martinelli, and C Scoglio. 2002. SPeCRA: A stochastic performance comparison routing algorithm for LSP setup in MPLS networks. In *Global Telecommunications Conference, 2002. GLOBECOM'02. IEEE*, Vol. 3. IEEE, 2190–2194.

[21] Jaudelice Cavalcante de Oliveira, Caterina Scoglio, Ian F Akyildiz, and George Uhl. 2004. New preemption policies for DiffServ-aware traffic engineering to minimize rerouting in MPLS networks. *IEEE/ACM transactions on networking* 12, 4 (2004), 733–745.

[22] Anwar Elwalid, Cheng Jin, Steven Low, and Indra Widjaja. 2001. MATE: MPLS adaptive traffic engineering. In *Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society (Cat. No. 01CH37213)*, Vol. 3. IEEE, 1300–1309.

[23] Yingya Guo, Yulong Ma, Huan Luo, and Jianping Wu. 2022. Traffic Engineering in a Shared Inter-DC WAN via Deep Reinforcement Learning. *IEEE Transactions on Network Science and Engineering* 9, 4 (2022), 2870–2881. https://doi.org/10.1109/TNSE.2022.3172283

[24] Gurobi. 2024. *Gurobi solver*. https://www.gurobi.com/

[25] James Hamilton. 2016. *AWS re:Invent 2016: Tuesday Night Live with James Hamilton*. https://www.youtube.com/watch?v=AyOAjFNPAbA

[26] Sanjay Hegde, David Lapsley, Bartek Wydrowski, Jan Lindheim, David Wei, Cheng Jin, Steven Low, and Harvey Newman. 2004. FAST TCP in High-Speed Networks: An Experimental Study. In *Proceedings of GridNet*.

[27] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[28] Chi-Yao Hong, Srikanth Kandula, Ratul Mahajan, Ming Zhang, Vijay Gill, Mohan Nanduri, and Roger Wattenhofer. 2013. Achieving high utilization with software-driven WAN. In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*. 15–26.

[29] Chi-Yao Hong, Subhasree Mandal, Mohammad Al-Fares, Min Zhu, Richard Alimi, Chandan Bhagat, Sourabh Jain, Jay Kaimal, Shiyu Liang, Kirill Mendelev, et al. 2018. B4 and after: managing hierarchy, partitioning, and asymmetry for availability and scale in google's software-defined WAN. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*. 74–87.

[30] Oliver Hope and Eiko Yoneki. 2021. GDDR: GNN-based data-driven routing. In *2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 517–527.

[31] IBM. 2024. *CPLEX*. https://www.ibm.com/products/ilog-cplex-optimization-studio

[32] Virajith Jalaparti, Ivan Bliznets, Srikanth Kandula, Brendan Lucier, and Ishai Menache. 2016. Dynamic pricing and traffic engineering for timely inter-datacenter transfers. In *Proceedings of the 2016 ACM SIGCOMM Conference*. 73–86.

[33] Weiwei Jiang. 2022. Graph-based deep learning for communication networks: A survey. *Computer Communications* 185 (2022), 40–54. https://doi.org/10.1016/j.comcom.2021.12.015

[34] 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 *Proceedings of the 2016 ACM SIGCOMM Conference*. 87–100.

[35] Srikanth Kandula, Dina Katabi, Bruce Davie, and Anna Charny. 2005. Walking the tightrope: Responsive yet stable traffic engineering. *ACM SIGCOMM Computer Communication Review* 35, 4 (2005), 253–264.

[36] Srikanth Kandula, Ishai Menache, Roy Schwartz, and Spandana Raj Babbula. 2014. Calendaring for wide area networks. In *Proceedings of the 2014 ACM conference on SIGCOMM*. 515–526.

[37] Koushik Kar, Murali Kodialam, and T VIEEE Lakshman. 2000. Minimum interference routing of bandwidth guaranteed tunnels with MPLS traffic engineering applications. *IEEE Journal on selected areas in communications* 18, 12 (2000), 2566–2579.

[38] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan. 1998. Rate Control for Communication Networks: Shadow Prices, Proportional Fairness and Stability. *The Journal of the Operational Research Society* 49, 3 (1998), 237–252. http://www.jstor.org/stable/3010473

[39] Hyojoon Kim and Nick Feamster. 2013. Improving network management with software defined networking. *IEEE Communications Magazine* 51, 2 (2013), 114–119. https://doi.org/10.1109/MCOM.2013.6461195

[40] Umesh Krishnaswamy, Rachee Singh, Nikolaj Bjørner, and Himanshu Raj. 2022. Decentralized cloud wide-area network traffic engineering with {BLASTSHIELD}. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*. 325–338.

[41] Praveen Kumar, Yang Yuan, Chris Yu, Nate Foster, Robert Kleinberg, Petr Lapukhov, Chiun Lin Lim, and Robert Soulé. 2018. {Semi-oblivious} traffic engineering: The road not taken. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*. 157–170.

[42] Jia Li, Zhichao Han, Hong Cheng, Jiao Su, Pengyun Wang, Jianfeng Zhang, and Lujia Pan. 2019. Predicting Path Failure In Time-Evolving Graphs *(KDD '19)*. Association for Computing Machinery, New York, NY, USA, 1279–1289. https://doi.org/10.1145/3292500.3330847

[43] Hongqiang Harry Liu, Srikanth Kandula, Ratul Mahajan, Ming Zhang, and David Gelernter. 2014. Traffic engineering with forward fault correction. In *Proceedings of the 2014 ACM Conference on SIGCOMM*. 527–538.

[44] Deep Medhi and Karthik Ramasamy. 2018. Chapter 7 - IP Traffic Engineering. In *Network Routing (Second Edition)* (second edition ed.), Deep Medhi and Karthik Ramasamy (Eds.). Morgan Kaufmann, Boston, 214–258. https://doi.org/10.1016/B978-0-12-800737-2.00009-0

[45] Mosek. 2024. *MOSEK*. https://www.mosek.com/

[46] D.P. Palomar and Mung Chiang. 2006. A tutorial on decomposition methods for network utility maximization. *IEEE Journal on Selected Areas in Communications* 24, 8 (2006), 1439–1451. https://doi.org/10.1109/JSAC.2006.879350

[47] Abhinav Pathak, Ming Zhang, Y Charlie Hu, Ratul Mahajan, and Dave Maltz. 2011. Latency inflation with MPLS-based traffic engineering. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*. 463–472.

[48] Yarin Perry, Felipe Vieira Frujeri, Chaim Hoch, Srikanth Kandula, Ishai Menache, Michael Schapira, and Aviv Tamar. 2023. {DOTE}: Rethinking (Predictive){WAN} Traffic Engineering. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*. 1557–1581.

[49] Yibo Pi, Sugih Jamin, Peter Danzig, and Feng Qian. 2020. Latency Imbalance Among Internet Load-Balanced Paths: A Cloud-Centric View. *Proc. ACM Meas. Anal. Comput. Syst.* 4, 2, Article 32 (jun 2020), 29 pages. https://doi.org/10.1145/3392150

[50] Gomathi Ramachandran. 2024. Control-Plane Observability Through Data-Plane Performance. In *2024 16th International Conference on COMmunication Systems & NETworkS (COMSNETS)*. 889–896. https://doi.org/10.1109/COMSNETS59351.2024.10426959

[51] Hiroyuki Saito, Yasuhiro Miyao, and Makiko Yoshida. 2000. Traffic engineering using multiple multipoint-to-point LSPs. In *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No. 00CH37064)*, Vol. 2. IEEE, 894–901.

[52] Caterina Scoglio, Tricha Anjali, Jaudelice Cavalcante de Oliveira, Ian F Akyildiz, and G Uhl. 2004. TEAM: A traffic engineering automated manager for DiffServ-based MPLS networks. *IEEE Communications Magazine* 42, 10 (2004), 134–145.

[53] Kamran Shaukat, Suhuai Luo, Vijay Varadharajan, Ibrahim A. Hameed, and Min Xu. 2020. A Survey on Machine Learning Techniques for Cyber Security in the Last Decade. *IEEE Access* 8 (2020), 222310–222354. https://doi.org/10.1109/ACCESS.2020.3041951

[54] Rachee Singh, Nikolaj Bjorner, Sharon Shoham, Yawei Yin, John Arnold, and Jamie Gaudette. 2021. Cost-effective capacity provisioning in wide area networks with Shoofly. In *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*. 534–546.

[55] Rachee Singh, Manya Ghobadi, Klaus-Tycho Foerster, Mark Filer, and Phillipa Gill. 2018. RADWAN: rate adaptive wide area network. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*. 547–560.

[56] Rayadurgam Srikant. 2004. *The Mathematics of Internet Congestion Control (Systems and Control: Foundations and Applications)*. SpringerVerlag.

[57] Sushant Jain et. al. 2013. B4: Experience with a globally-deployed software defined WAN. *ACM SIGCOMM Computer Communication Review* 43, 4 (2013), 3–14.

[58] Panos Trimintzios, Timothy Baugé, George Pavlou, Paris Flegkas, and Richard Egan. 2003. Quality of service provisioning through traffic engineering with applicability to IP-based production networks. *Computer Communications* 26, 8 (2003), 845–860.

[59] Panos Trimintzios, Leonidas Georgiadis, George Pavlou, David Griffin, Carlos Frederico Marcelo da Cunha Cavalcanti, Panos Georgatsos, and C Jacquenet. 2001. Engineering the multi-service internet: MPLS and IP-based techniques. (2001).

[60] Shih-Hao Tseng, Saksham Agarwal, Rachit Agarwal, Hitesh Ballani, and Ao Tang. 2021. {CodedBulk}:{Inter-Datacenter} Bulk Transfers using Network Coding. In *18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*. 15–28.

[61] Asaf Valadarsky, Michael Schapira, Dafna Shahaf, and Aviv Tamar. 2017. Learning to route. In *Proceedings of the 16th ACM workshop on hot topics in networks*. 185–191.

[62] Hao Wang, Haiyong Xie, Lili Qiu, Yang Richard Yang, Yin Zhang, and Albert Greenberg. 2006. COPE: Traffic engineering in dynamic networks. In *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*. 99–110.

[63] Yufei Wang and Zheng Wang. 1999. Explicit routing algorithms for internet traffic engineering. In *Proceedings Eight International Conference on Computer Communications and Networks (Cat. No. 99EX370)*. IEEE, 582–588.

[64] David X. Wei, Cheng Jin, Steven H. Low, and Sanjay Hegde. 2006. FAST TCP: Motivation, Architecture, Algorithms, Performance. *IEEE/ACM Transactions on Networking* 14, 6 (2006), 1246–1259. https://doi.org/10.1109/TNET.2006.886335

[65] Zhiyuan Xu, Jian Tang, Jingsong Meng, Weiyi Zhang, Yanzhi Wang, Chi Harold Liu, and Dejun Yang. 2018. Experience-driven networking: A deep reinforcement learning based approach. In *IEEE INFOCOM 2018-IEEE conference on computer communications*. IEEE, 1871–1879.

[66] Zhiying Xu, Francis Y. Yan, Rachee Singh, Justin T. Chiu, Alexander M. Rush, and Minlan Yu. 2023. Teal: Learning-Accelerated Optimization of WAN Traffic Engineering. In *Proceedings of the ACM SIGCOMM 2023 Conference* (New York, NY, USA) *(ACM SIGCOMM '23)*. Association for Computing Machinery, New York, NY, USA, 378–393. https://doi.org/10.1145/3603269.3604857

[67] Lei Xue, Xiaobo Ma, Xiapu Luo, Edmond W. W. Chan, Tony T. N. Miu, and Guofei Gu. 2018. LinkScope: Toward Detecting Target Link Flooding Attacks. *IEEE Transactions on Information Forensics and Security* 13, 10 (2018), 2423–2438. https://doi.org/10.1109/TIFS.2018.2815555

[68] Kok-Kiong Yap, Murtaza Motiwala, Jeremy Rahe, Steve Padgett, Matthew Holliman, Gary Baldus, Marcus Hines, Taeeun Kim, Ashok Narayanan, Ankur Jain, et al. 2017. Taking the edge off with espresso: Scale, reliability and programmability for global internet peering. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*. 432–445.

[69] Junjie Zhang, Minghao Ye, Zehua Guo, Chen-Yu Yen, and H Jonathan Chao. 2020. CFR-RL: Traffic engineering with reinforcement learning in SDN. *IEEE Journal on Selected Areas in Communications* 38, 10 (2020), 2249–2259.