Dynamic resource allocation problems in communication networks:

Machine Learning for resource allocation

Alexandre Reiffers-Masson

Equipe Maths&Net, IMT Atlantique, CS department
LabSTICC (UMR CNRS 6285)

July 14, 2023

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Lab-STICC

Context

A simple optimization problem

Realistic telecommunication networks

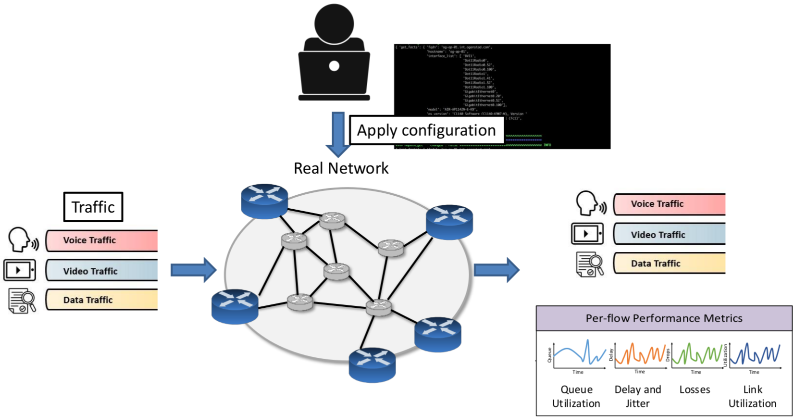

Figure: Taken from "Graph Neural Networking challenge 2023: Building a Network Digital Twin using data from real networks"

Issue with a realistic networking problem

- Simple models do not predict correctly the **traffic**.

Issue with a realistic networking problem

- Simple models do not predict correctly the **traffic**.
- The network is controlled through a given set of configurations. Most of the time the software used is **proprietary**, and therefore, we cannot control as much as we want.

Issue with a realistic networking problem

- Simple models do not predict correctly the **traffic**.
- The network is controlled through a given set of configurations. Most of the time the software used is **proprietary**, and therefore, we cannot control as much as we want.
- The size of the queues is not known and the routing paths are not exact, etc...

Traffic pattern

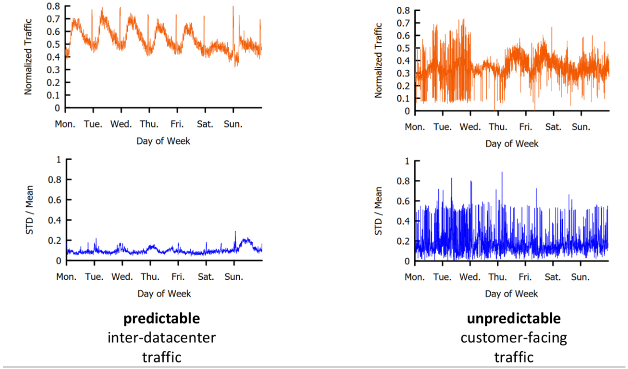

Figure: Taken from "DOTE: Rethinking (Predictive) WAN Traffic Engineering"

Control: Configuration[1]

- Topology, Link Capacity

[1] Graph Neural Networking challenge 2023: Building a Network Digital Twin using data from real networks.

Control: Configuration[1]

- Topology, Link Capacity
- Routing:

[1] Graph Neural Networking challenge 2023: Building a Network Digital Twin using data from real networks.

Control: Configuration[1]

- Topology, Link Capacity
- Routing:
  1. Overlay: SRv6, MPLS...
  2. Underlay: OSPF, BGP...

[1] Graph Neural Networking challenge 2023: Building a Network Digital Twin using data from real networks.

Control: Configuration[1]

- Topology, Link Capacity
- Routing:
    1. Overlay: SRv6, MPLS...
    2. Underlay: OSPF, BGP...
- Scheduling Policy (arbitrary)

[1] Graph Neural Networking challenge 2023: Building a Network Digital Twin using data from real networks.

Control: Configuration[1]

- Topology, Link Capacity
- Routing:
  1. Overlay: SRv6, MPLS...
  2. Underlay: OSPF, BGP...
- Scheduling Policy (arbitrary)
  1. Queue Length
  2. Policy
  3. Hierarchy of policies

[1] Graph Neural Networking challenge 2023: Building a Network Digital Twin using data from real networks.

Control: Configuration[1]

- Topology, Link Capacity
- Routing:
  1. Overlay: SRv6, MPLS...
  2. Underlay: OSPF, BGP...
- Scheduling Policy (arbitrary)
  1. Queue Length
  2. Policy
  3. Hierarchy of policies
- ECMP, LAG, etc

[1] Graph Neural Networking challenge 2023: Building a Network Digital Twin using data from real networks.

Context

A simple optimization problem

Optimisation

The network is modeled as a capacitated graph $G = (V, E, c)$, where the function $c()$ assigns a capacity to each edge.

Tunnels: Each source vertex s communicates with each destination vertex t via a set of network paths, or "tunnels" P_{st}. P_{st} can be interpreted as the routing matrix associated with the couple source/destination (s, t).

Optimisation

The network is modeled as a capacitated graph $G = (V, E, c)$, where the function $c()$ assigns a capacity to each edge.

Tunnels: Each source vertex s communicates with each destination vertex t via a set of network paths, or "tunnels" P_{st}. P_{st} can be interpreted as the routing matrix associated with the couple source/destination (s, t).

Traffic demands: A demand matrix (DM) D is an $|V| \times |V|$ a matrix whose (i, j)-th entry $D_{i,j}$ specifies the traffic demand between source i and destination j.

Optimisation

The network is modeled as a capacitated graph $G = (V, E, c)$, where the function $c()$ assigns a capacity to each edge.

Tunnels: Each source vertex s communicates with each destination vertex t via a set of network paths, or "tunnels" P_{st}. P_{st} can be interpreted as the routing matrix associated with the couple source/destination (s, t).

Traffic demands: A demand matrix (DM) D is an $|V| \times |V|$ a matrix whose (i, j)-th entry $D_{i,j}$ specifies the traffic demand between source i and destination j.

Configurations: A given a network graph and demand matrix, a *configuration* specifies for each source vertex s and destination vertex t how the $D_{s,t}$ traffic from s to t is split across the tunnels in P_{st}.

Objective: minimizing maximum-link utilization.

Objective: minimizing maximum-link utilization.

$$\min_{x\geq 0} \quad \max_{e} \frac{\sum_{s,t}\sum_{p\in P_{st},e\in p} D_{s,t}x_p}{c_e} \tag{1}$$

$$s.t. \qquad \sum_{p\in P_{st}} x_p = 1,\ \forall(s,t). \tag{2}$$

Objective: minimizing maximum-link utilization.

$$\min_{x\geq 0} \quad \max_{e} \frac{\sum_{s,t}\sum_{p\in P_{st},e\in p} D_{s,t}x_p}{c_e} \tag{1}$$

$$s.t. \qquad \sum_{p\in P_{st}} x_p = 1,\ \forall (s,t). \tag{2}$$

This problem is a convex optimization problem (actually an LP problem).

Objective: minimizing maximum-link utilization.

$$\min_{x\geq 0} \quad \max_{e} \frac{\sum_{s,t}\sum_{p\in P_{st},e\in p} D_{s,t}x_p}{c_e} \tag{1}$$

$$s.t. \quad \sum_{p\in P_{st}} x_p = 1,\ \forall(s,t). \tag{2}$$

This problem is a convex optimization problem (actually an LP problem).

What can we do priori knowledge of the traffic demands?

General Traffic Model

The demand matrix D_t is generated according to an **unknown** H-Markov process with transition probabilities such that:

$$\mathbb{P}(D_t|D_{t-1}, \ldots, D_{t-H}) = \mathbb{P}(D_t|D_{t-1}, ..., D_1). \quad (3)$$

We assume that the Markov chain has reached it's *stationary regime.*

Approach 1: Demand-Prediction-Based

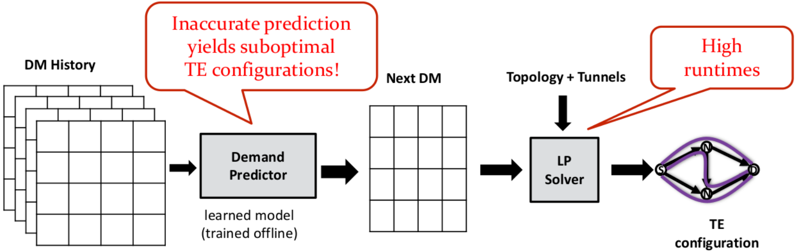

Figure: From "DOTE: Rethinking (Predictive) WAN Traffic Engineering"

Approach 1: Demand-Prediction-Based

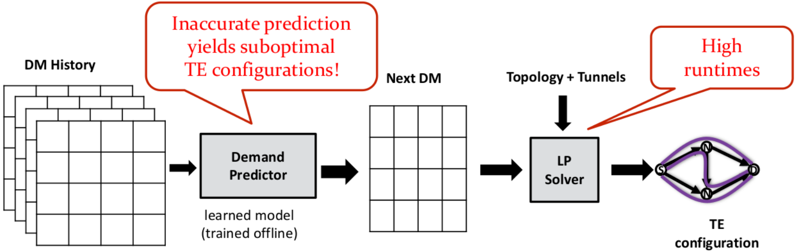

Figure: From "DOTE: Rethinking (Predictive) WAN Traffic Engineering"

- Advantages: Can be reuse for other tasks. In case of a problem it is easy to identity where the problem is coming from.

Approach 1: Demand-Prediction-Based

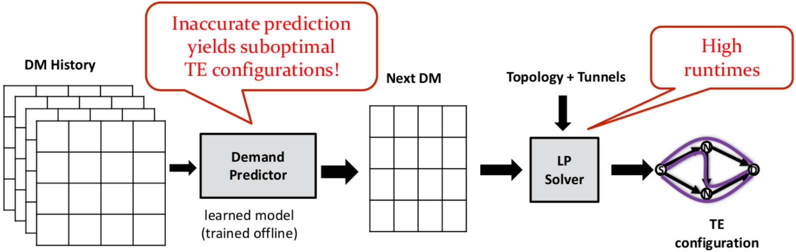

Figure: From "DOTE: Rethinking (Predictive) WAN Traffic Engineering"

- Advantages: Can be reuse for other tasks. In case of a problem it is easy to identity where the problem is coming from.
- Disadvantage: The demand predictor is not tuned for optimising Configuration.

Approach 2: Direct Optimisation

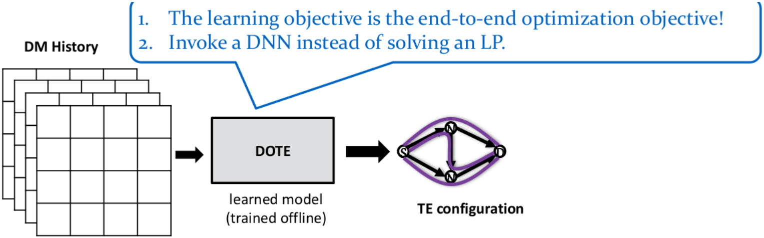

Figure: From "DOTE: Rethinking (Predictive) WAN Traffic Engineering"

- Advantage: Optimal for TE.

Approach 2: Direct Optimisation

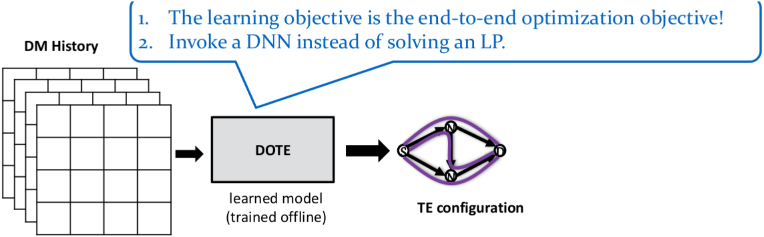

Figure: From "DOTE: Rethinking (Predictive) WAN Traffic Engineering"

- Advantage: Optimal for TE.
- Disadvantage: Cannot be used for other tasks.

Algorithm

DOTE

- **Input:** Observe $D(0)$ and the capacity c.
- **Set** $\theta := \theta_0$;
- **For** $t = 0, 2, \ldots,$ **do:**
  1. **Predict** the allocation $\pi(D_{t-1}, \ldots, D_{t-H}; \theta)$
  2. **Observe** the traffic matrix D_t for which the allocation has been done. Compute
  3. **Compute** the gradient (automatic differentiation) of :

$$f(\theta) := \max \frac{\sum_{s,t} \sum_{p \in P_{st}, e \in p} D_{s,t} \pi(D_t, \ldots, D_{t-1-H}; \theta)}{c_e}.$$

  4. **Update** the parameter θ as follow:

$$\theta = \theta - \alpha \nabla_\theta f(\theta).$$

Definition: Random Neural Networks

- p is a probability distribution on ω and $\phi : \mathcal{X} \times \Omega \to \mathbb{R}$ is a function such that $\sup_{x,\theta} |\phi(x, \theta)| \leq 1.$

Definition: Random Neural Networks

- p is a probability distribution on ω and $\phi : \mathcal{X} \times \Omega \to \mathbb{R}$ is a function such that $\sup_{x,\theta} |\phi(x, \theta)| \leq 1$.
- Let us define:

Definition: Random Neural Networks

- p is a probability distribution on ω and $\phi : \mathcal{X} \times \Omega \to \mathbb{R}$ is a function such that $\sup_{x,\theta} |\phi(x,\theta)| \leq 1$.
- Let us define:

$$\begin{aligned}
\mathcal{F}_p &:= \left\{ f(x) = \int_\Omega \alpha(\theta)\phi(x,\theta)d\theta \middle| |\alpha(\theta)| \leq Cp(\theta) \right\} \\
\mathcal{F}_\theta &:= \left\{ f(x) = \sum_{k=1}^{K} \alpha_k \phi(x,\theta_k) \middle| |\alpha_k| \leq \frac{C}{K} \right\}
\end{aligned}$$

Definition: Random Neural Networks

- p is a probability distribution on ω and $\phi : \mathcal{X} \times \Omega \to \mathbb{R}$ is a function such that $\sup_{x,\theta} |\phi(x,\theta)| \leq 1$.
- Let us define:

$$\begin{aligned}
\mathcal{F}_p &:= \left\{ f(x) = \int_\Omega \alpha(\theta)\phi(x,\theta)d\theta \,\middle|\, |\alpha(\theta)| \leq Cp(\theta) \right\} \\
\mathcal{F}_\theta &:= \left\{ f(x) = \sum_{k=1}^{K} \alpha_k \phi(x,\theta_k) \,\middle|\, |\alpha_k| \leq \frac{C}{K} \right\}
\end{aligned}$$

- Note that $\mathcal{F}_p$ consists of functions whose weights $\alpha(\theta)$ decays more rapidly than the given sampling distribution $p(\theta)$.

Performance

Let f be a function from $\mathcal{F}_p$. If μ is a probability measure on $\mathcal{X}$, $\theta_1, ..., \theta_K$ are drawn iid from p, then for all $\delta > 0$, there exist with a probability $1 - \delta$ a function $\hat{f} \in \mathcal{F}_\theta$ such that :

$$||f - \hat{f}||_{2,\mu} \leq \frac{C}{\sqrt{K}} \left(1 + \sqrt{2 \log \frac{1}{\delta}} \right),$$

with $||f - g||_{2,\mu}^2 = \int_{\mathcal{X}} (f - g)^2 d\mu$.

Numerical Illustrations

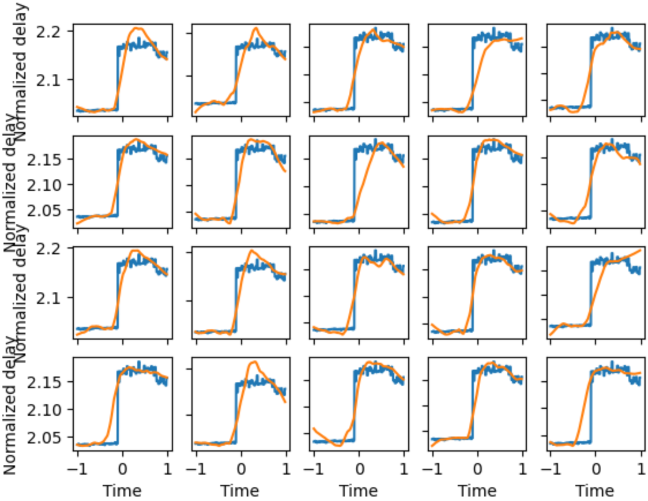

Figure: Random number of samplings: 20, $K = 20$.

Open problems

- We don't know the capacity on each link. How can we compute the gradient?
- How to transfer a learned network to another set-up?
- How to manage when the network is changing of states over time (queues, failures, etc.)?

Open problems

- We don't know the capacity on each link. How can we compute the gradient? **Sol: One point gradient estimator**
- How to transfer a learned network to another set-up? **Sol: understand the notion of scaling**. Scaling to larger networks often entails more aspects beyond the topology size. In particular, there are two main properties that we can observe as networks become larger:
    1. higher link capacities, as core links of the network typically aggregate more traffic,
    2. different flow-level delay distributions, as end-to-end paths are larger and they can traverse links with higher capacities.
- How to manage when the network is changing of states over time (queues, failures, etc.)? **Sol: Deep RL**

Scaling

Bibliography

- A simple way to play with a realistic set-up for optimization of configuration of a network: `https://bnn.upc.edu/challenge/gnnet2022/`, `https://bnn.upc.edu/challenge/gnnet2023/`. Especially the following link: `https://github.com/BNN-UPC/GNNetworkingChallenge/blob/2022_DataCentricAI/quickstart.ipynb`
- DOTE: Rethinking (Predictive) WAN Traffic Engineering : `https://www.usenix.org/conference/nsdi23/presentation/perry`