NOKIA Bell Labs NYU | TANDON

Oblivious Routing using Learning Methods

Ufuk Usubütün, Murali Kodialam, T.V. Lakshman and Shivendra Panwar

GLOBECOM 2023 – Kuala Lumpur, Malaysia

Making Oblivious Routing Practical

Context

Routing in Networks

High volume,

Highly variable traffic

Motivation

Traffic Oblivious Routing

Stable and Robust Routing Solution

Far less complex handling of highly varying traffic

Compatible with current day networks

Contribution

Adversarial Learning Method

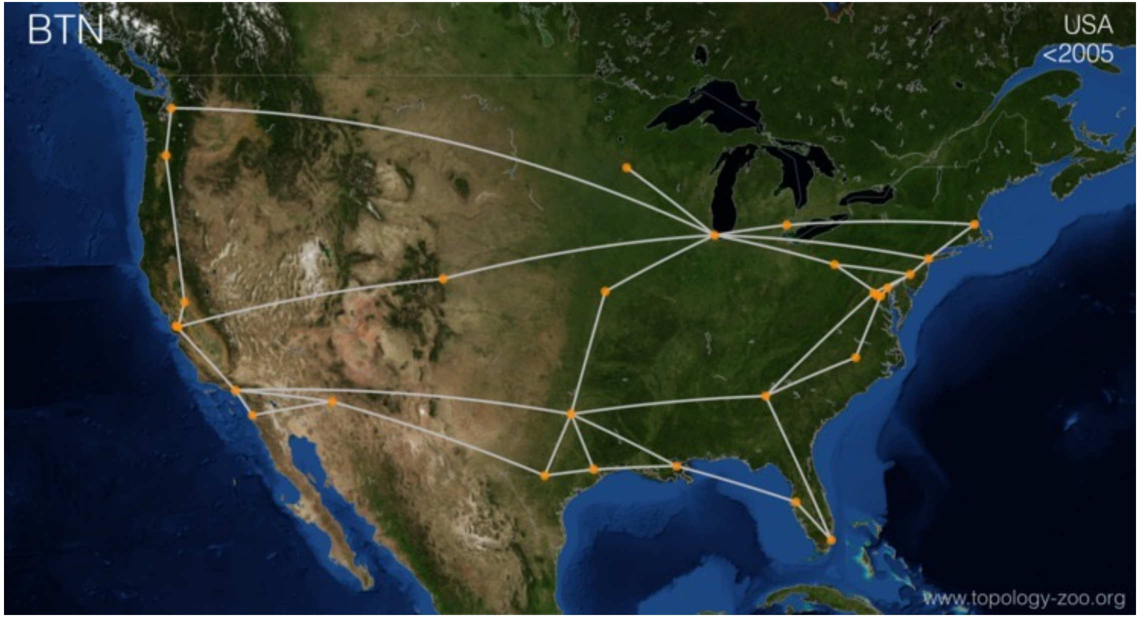
New, fast and parallelized ways to solve optimization problems

Provide performance guarantees

Routing in Networks

How to optimally route flows?

- How to satisfy all demands and constraints?



How to optimally route flows?

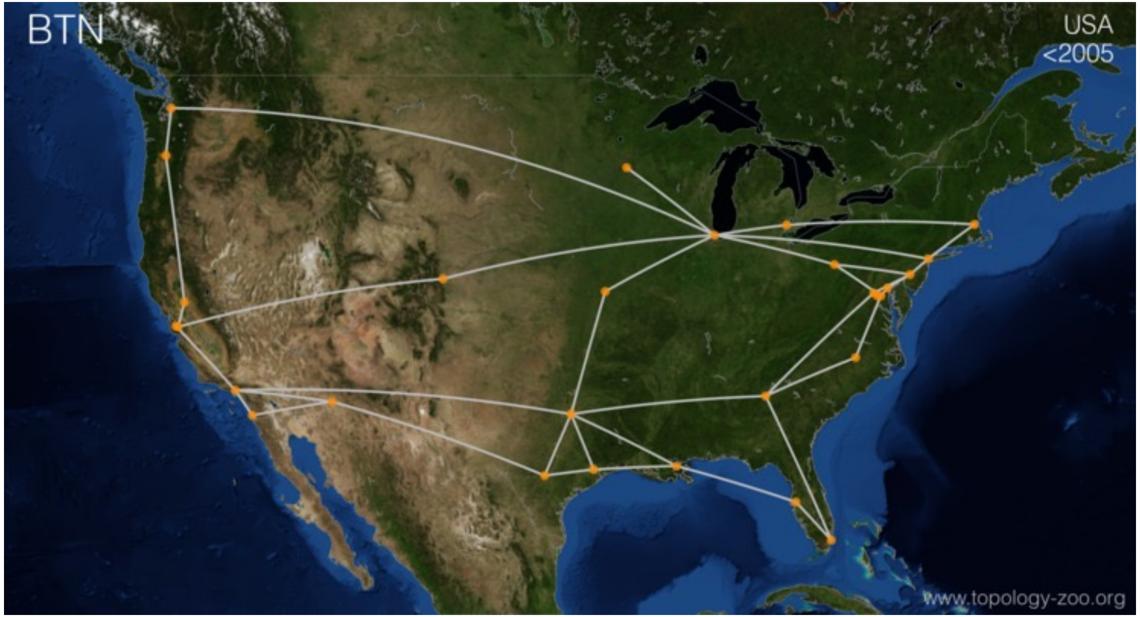
- ❑ How to satisfy all demands and constraints?
  - ❑ Solve a nice optimization problem if we can forecast what traffic to expect

min (max link utilization)

for traffic matrix:

$$[t_{ij}] = \begin{bmatrix} t_{11} & t_{12} & \cdots & t_{1N} \\ t_{21} & t_{22} & & t_{2N} \\ \vdots & & \ddots & \vdots \\ t_{N1} & t_{N2} & \cdots & t_{NN} \end{bmatrix}$$

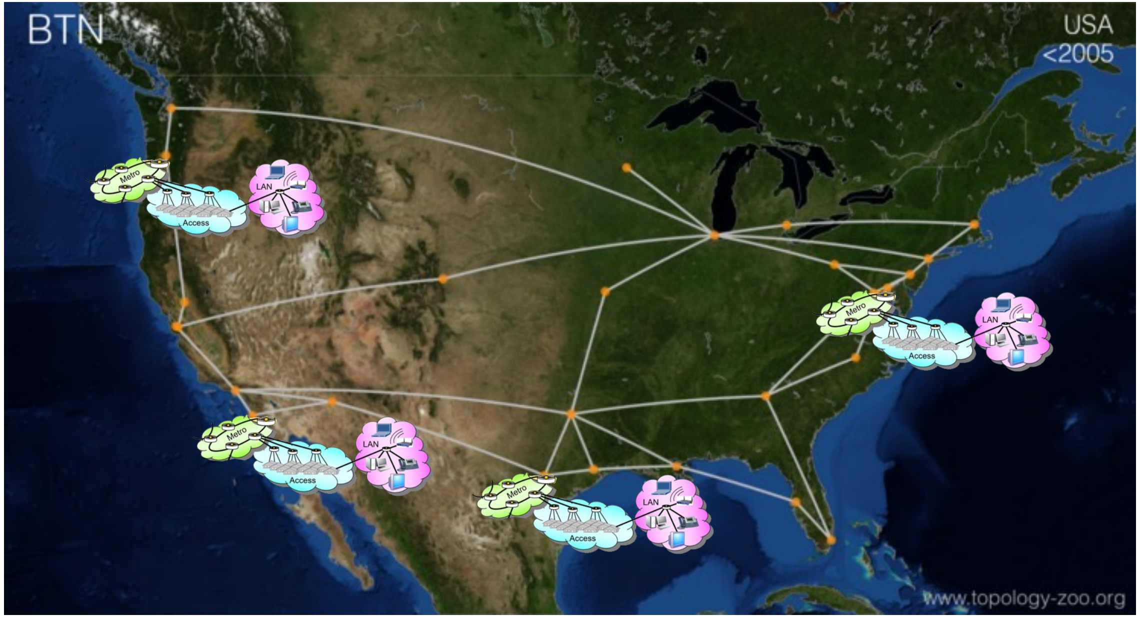
By assigning routes to flows



NYU | TANDON NOKIA Bell Labs

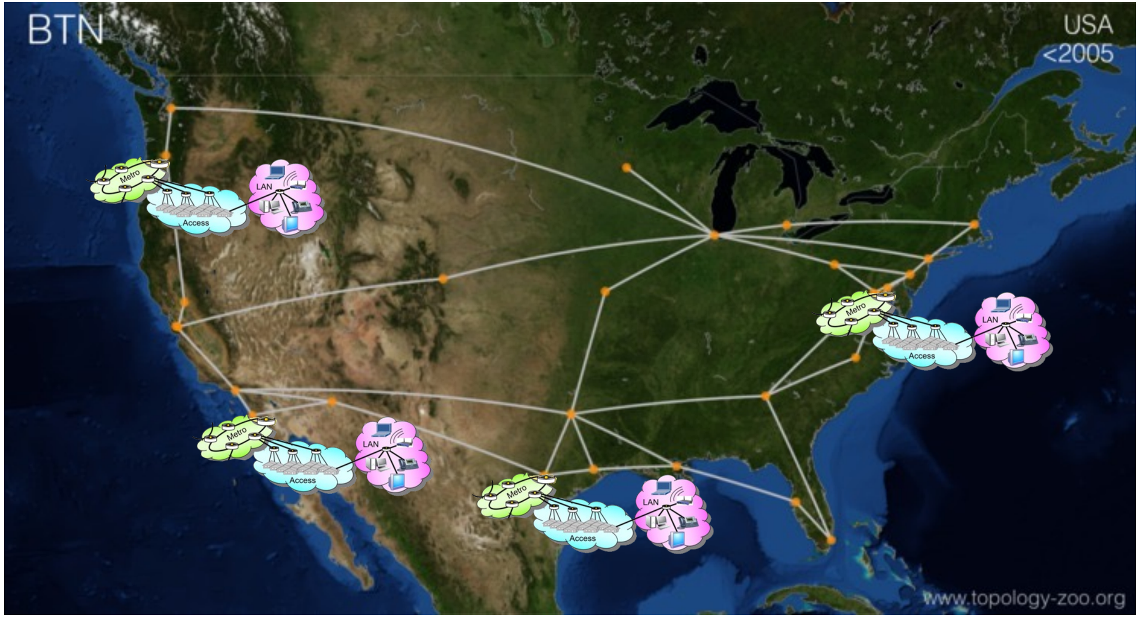
Traffic Patterns Today are Highly Variable and Harder to Predict

- ❑ High number of users
- ❑ Diverse applications
- ❑ Failures in adjacent networks
- ❑ Data Centers
- ❑ Edge Clouds
- ❑ Virtualization and Migration



How to optimize if we can’t forecast traffic?

- ❑ Over-provisioning without performance guarantees?

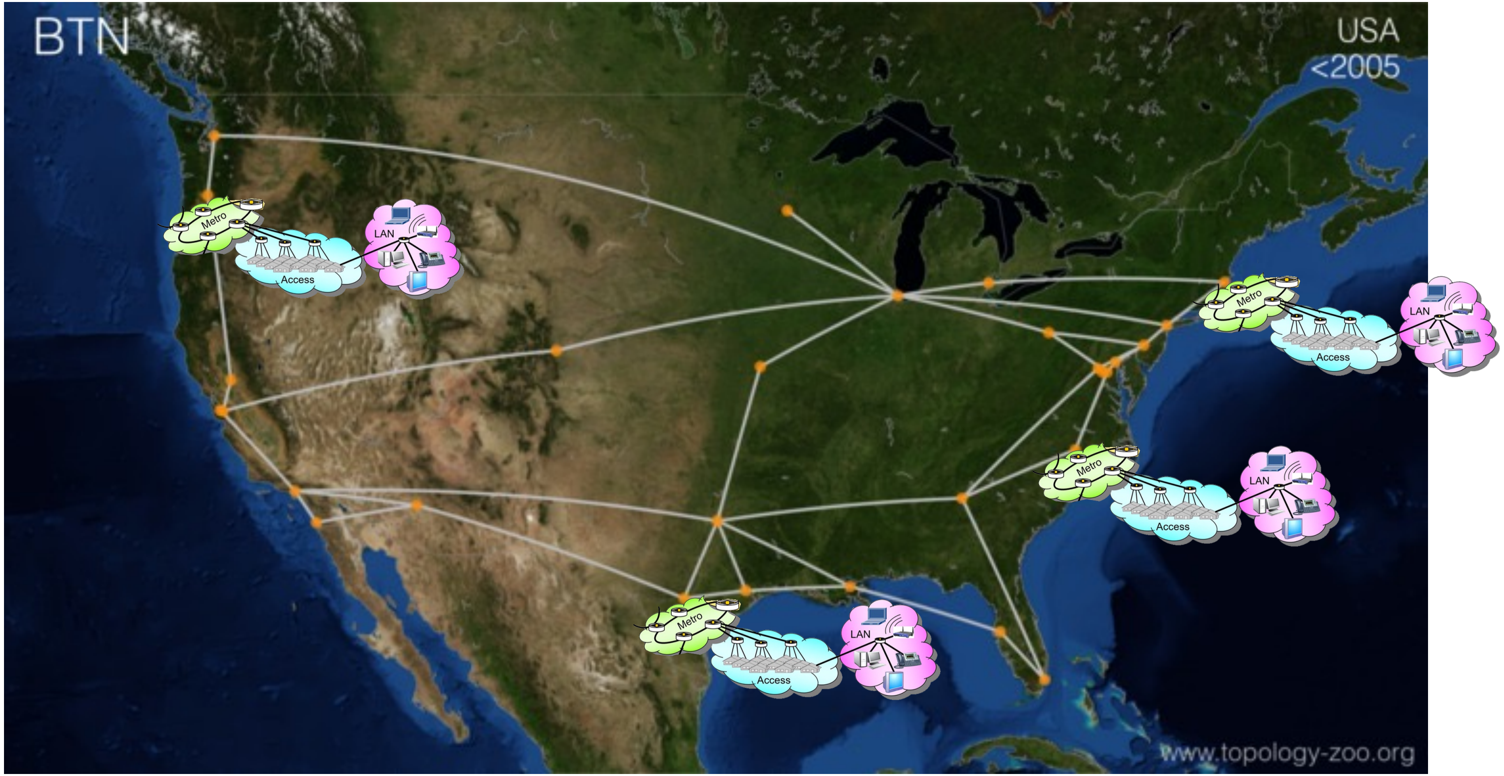


NYU TANDON NOKIA Bell Labs

Oblivious Routing

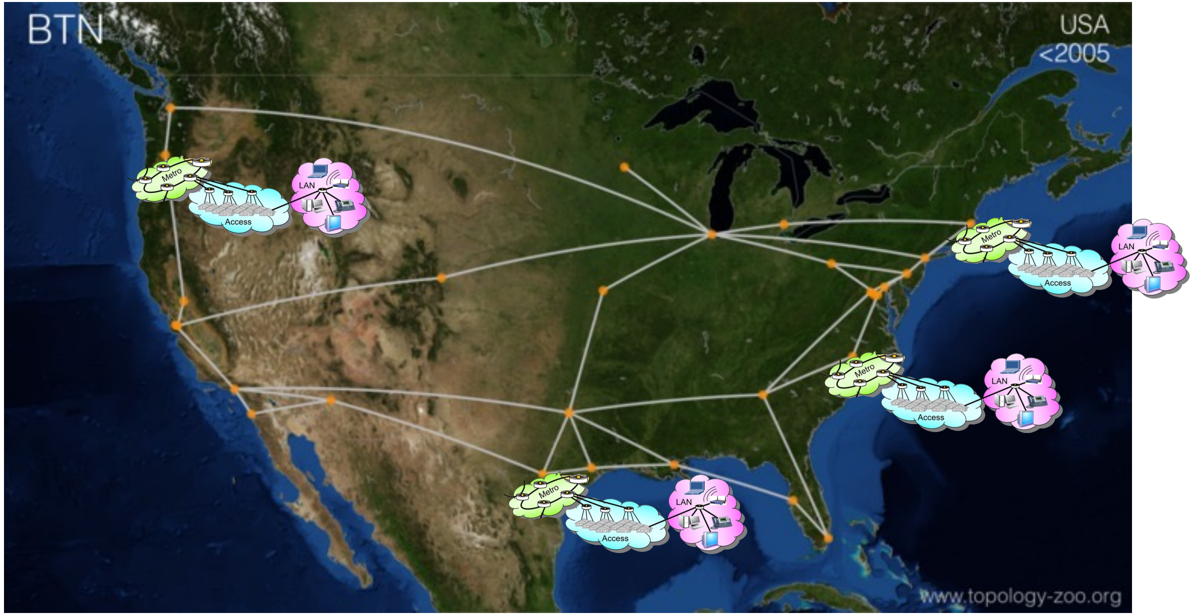
with Hose Constraints

We know link capacities



NYU TANDON NOKIA Bell Labs

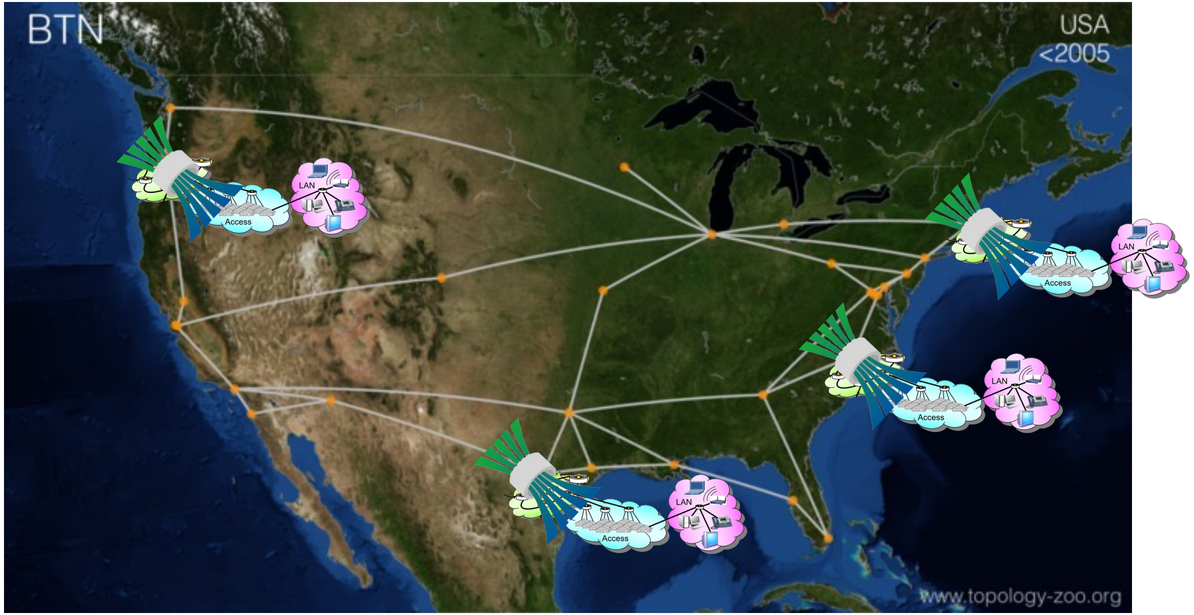
We know one more thing about the network:



NYU TANDON NOKIA Bell Labs

We know one more thing about the network:

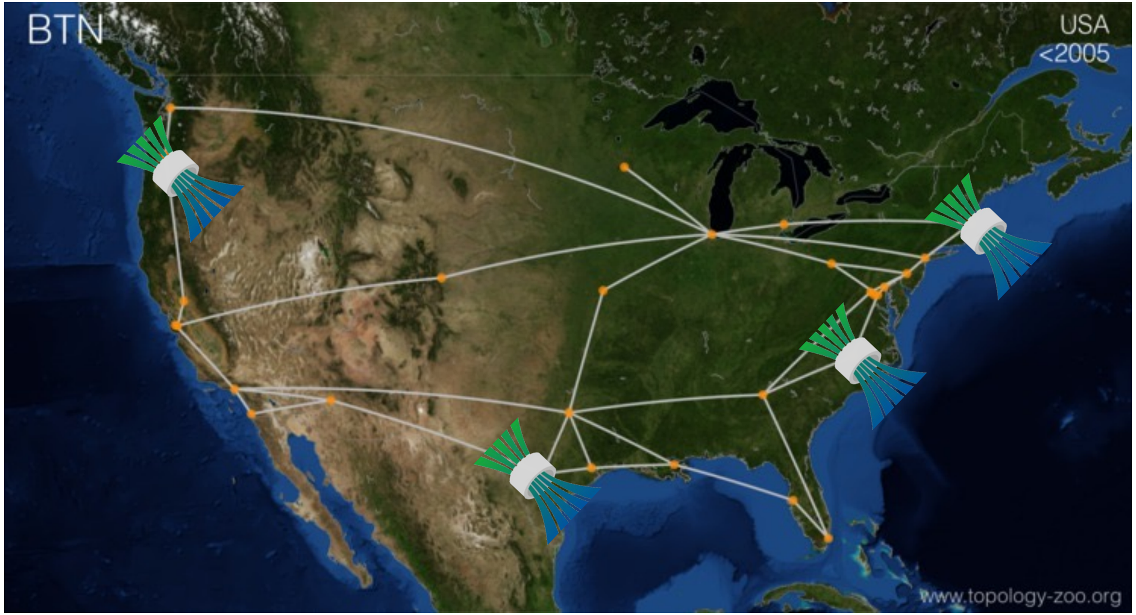
- The capacity of the physical connection of the network with the outside: The Hose



NYU | TANDON NOKIA Bell Labs

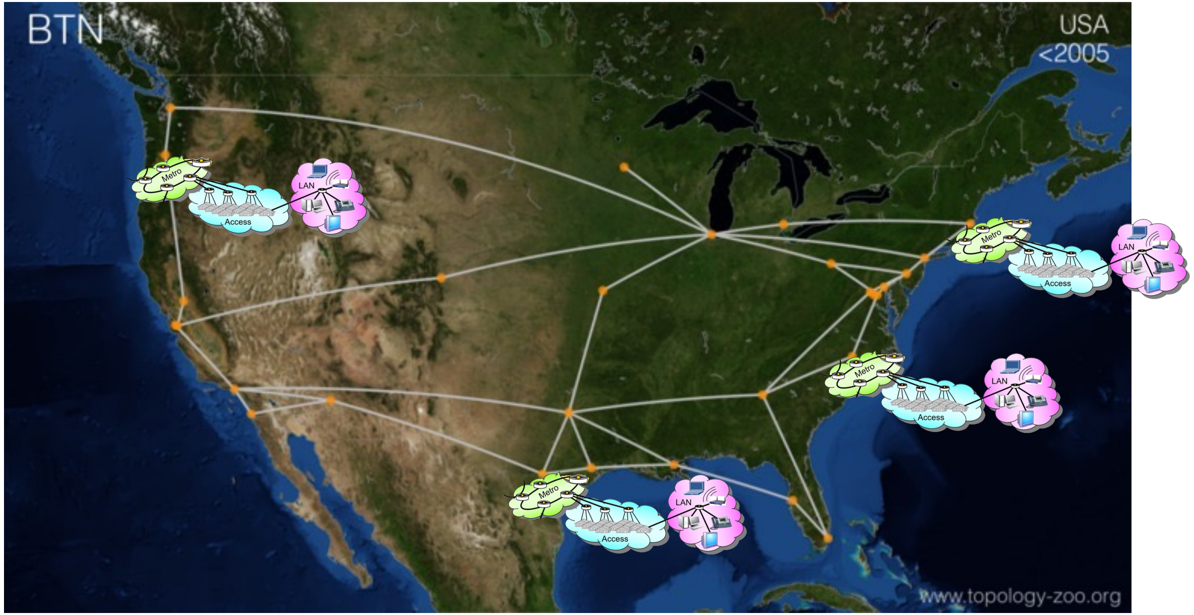
We know one more thing about the network:

- The capacity of the physical connection of the network with the outside: The Hose

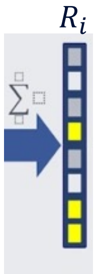


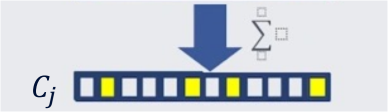
NYU TANDON NOKIA Bell Labs

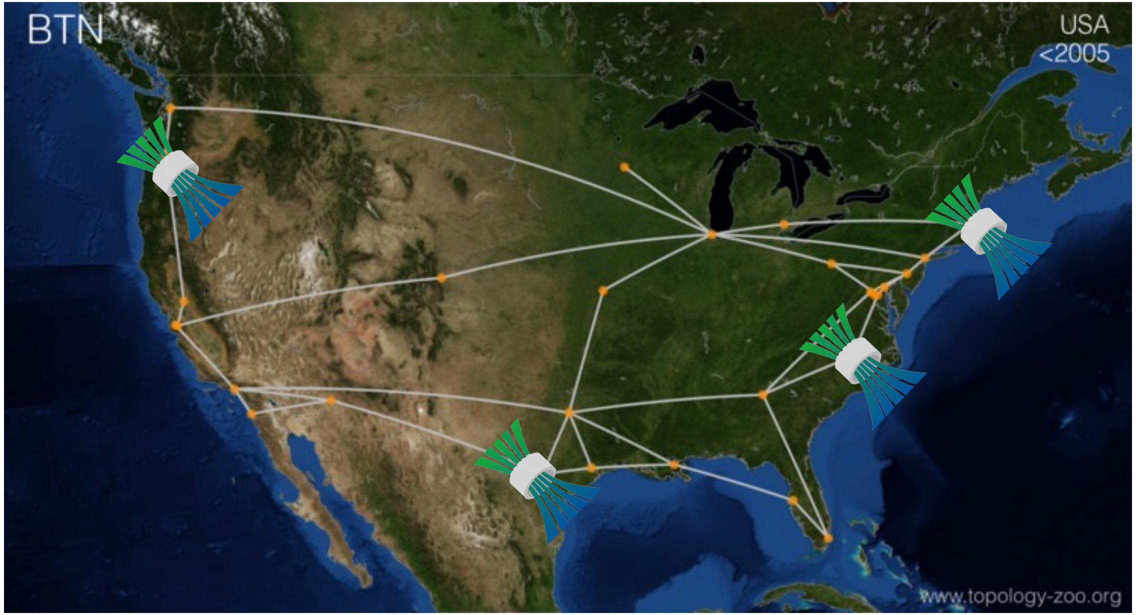
$$[t_{ij}] = \begin{bmatrix} t_{11} & t_{12} & \cdots & t_{1N} \\ t_{21} & t_{22} & & t_{2N} \\ \vdots & & \ddots & \vdots \\ t_{N1} & t_{N2} & \cdots & t_{NN} \end{bmatrix}$$



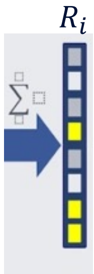
$$[t_{ij}] = \begin{bmatrix} t_{11} & t_{12} & \cdots & t_{1N} \\ t_{21} & t_{22} & & t_{2N} \\ \vdots & & \ddots & \vdots \\ t_{N1} & t_{N2} & \cdots & t_{NN} \end{bmatrix}$$

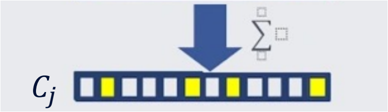




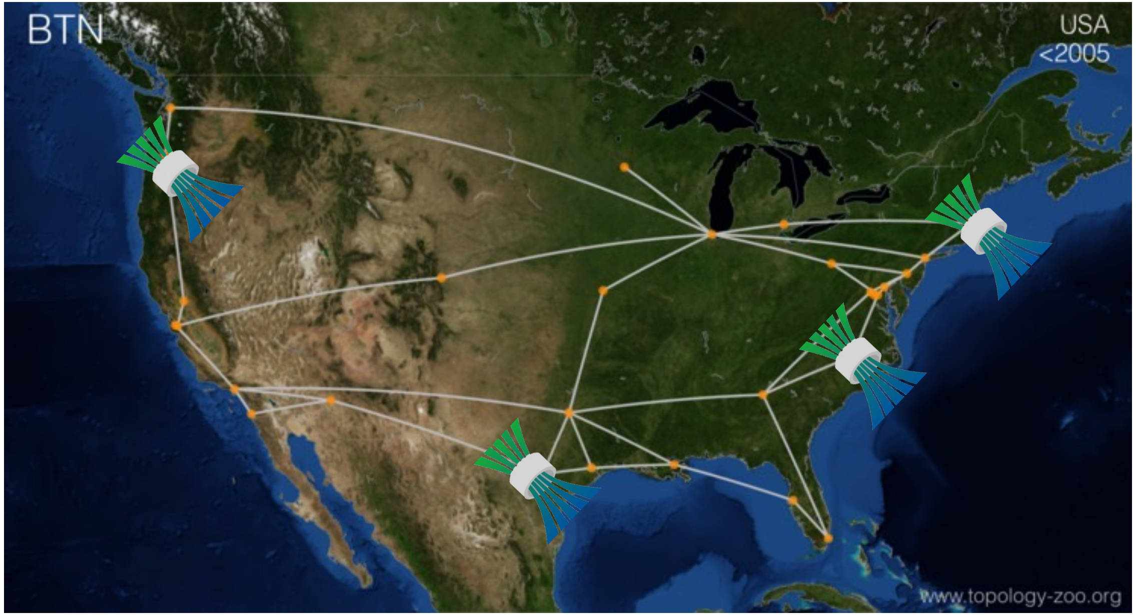


$$[t_{ij}] = \begin{bmatrix} t_{11} & t_{12} & \cdots & t_{1N} \\ t_{21} & t_{22} & & t_{2N} \\ \vdots & & \ddots & \vdots \\ t_{N1} & t_{N2} & \cdots & t_{NN} \end{bmatrix}$$



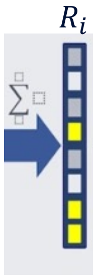


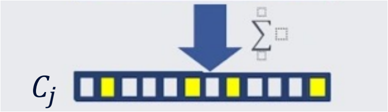
$$\mathcal{T}(\vec{\mathcal{R}},\vec{\mathcal{C}}) = \left\{ [t_{ij}] \,\middle|\, \sum_{j \neq i} t_{ij} \leq R_i \text{ and } \sum_{j \neq i} t_{ji} \leq C_i \;\forall\, i \right\}$$



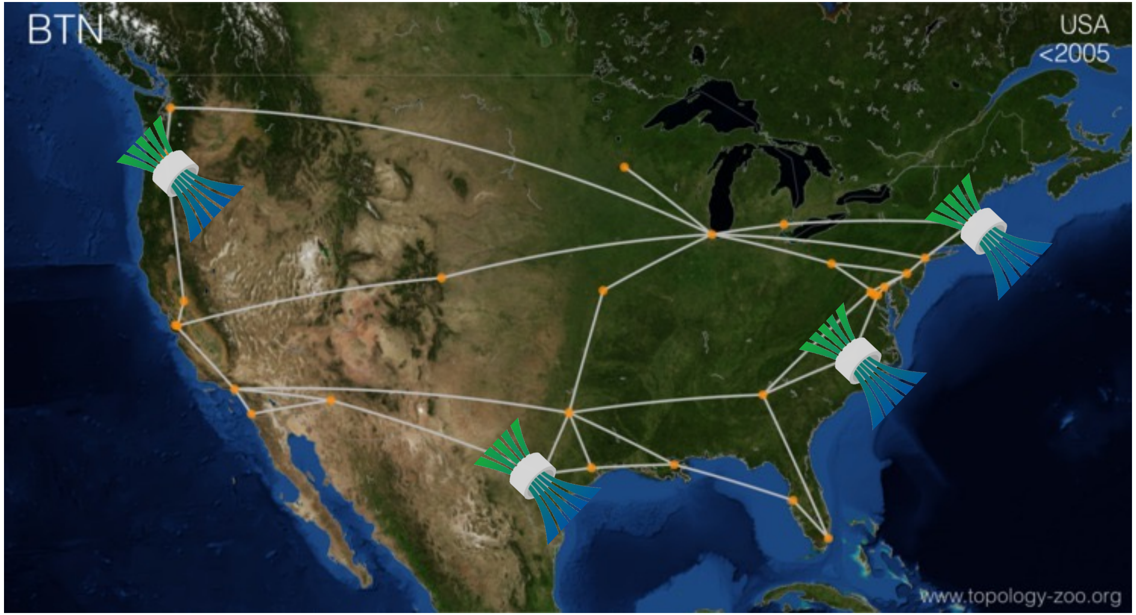
Find a fixed routing rule by solving for all possible traffic matrices

$$[t_{ij}] = \begin{bmatrix} t_{11} & t_{12} & \cdots & t_{1N} \\ t_{21} & t_{22} & & t_{2N} \\ \vdots & & \ddots & \vdots \\ t_{N1} & t_{N2} & \cdots & t_{NN} \end{bmatrix}$$



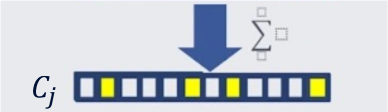


$$\mathcal{T}(\vec{\mathcal{R}},\vec{\mathcal{C}}) = \left\{ [t_{ij}] \;\middle|\; \sum_{j\neq i} t_{ij} \leq R_i \text{ and } \sum_{j\neq i} t_{ji} \leq C_i \;\forall\, i \right\}$$

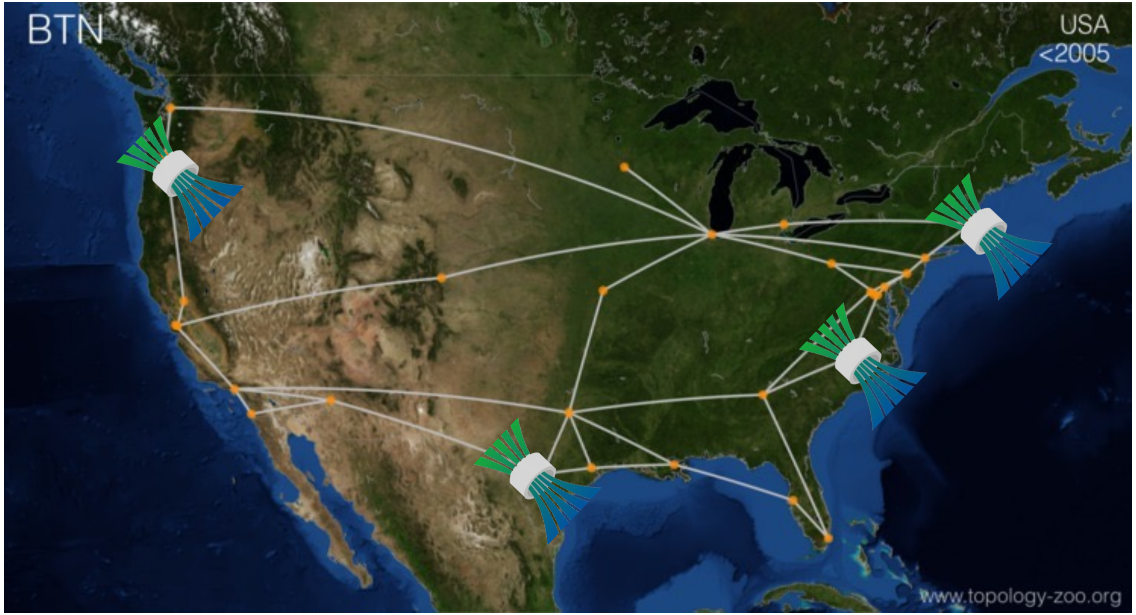


Find a fixed routing rule by solving for all possible traffic matrices

$$[t_{ij}] = \begin{bmatrix} t_{11} & t_{12} & \cdots & t_{1N} \\ t_{21} & t_{22} & & t_{2N} \\ \vdots & & \ddots & \vdots \\ t_{N1} & t_{N2} & \cdots & t_{NN} \end{bmatrix}$$



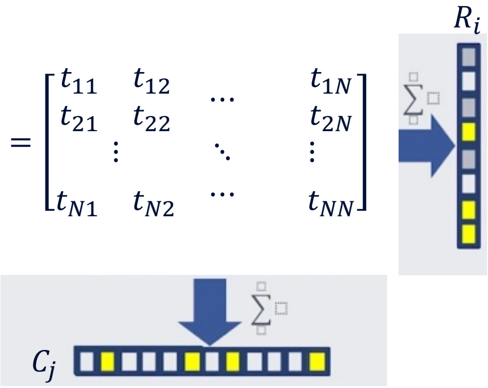
$$\mathcal{T}(\vec{\mathcal{R}},\vec{\mathcal{C}}) = \left\{ [t_{ij}] \,\middle|\, \sum_{j\neq i} t_{ij} \leq R_i \text{ and } \sum_{j\neq i} t_{ji} \leq C_i \;\forall\, i \right\}$$



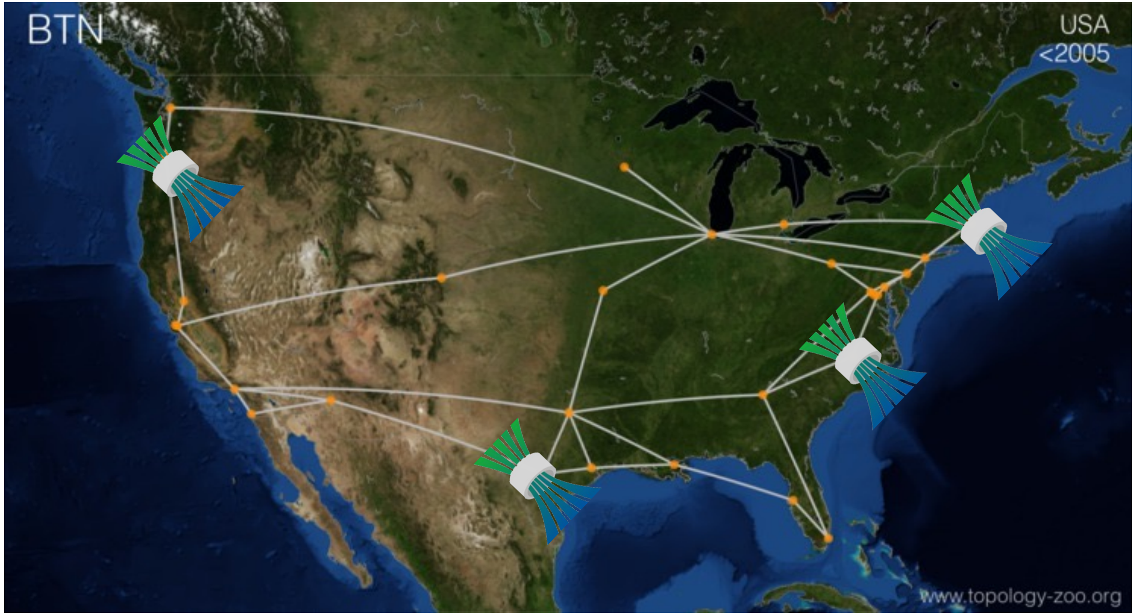
- ❑ Stable and robust scheme reducing network complexity

Find a fixed routing rule by solving for all possible traffic matrices

$$[t_{ij}] = \begin{bmatrix} t_{11} & t_{12} & \cdots & t_{1N} \\ t_{21} & t_{22} & & t_{2N} \\ \vdots & & \ddots & \vdots \\ t_{N1} & t_{N2} & \cdots & t_{NN} \end{bmatrix}$$



$$\mathcal{T}(\vec{\mathcal{R}},\vec{\mathcal{C}}) = \left\{ [t_{ij}] \,\middle|\, \sum_{j\neq i} t_{ij} \le R_i \text{ and } \sum_{j\neq i} t_{ji} \le C_i \;\forall\, i \right\}$$



- ❑ Stable and robust scheme reducing network complexity
- ❑ However, we need to optimize for infinitely many traffic matrices. A complex optimization problem!

NYU TANDON NOKIA Bell Labs

Learning Methods

Machine Learning is a Natural Way of Solving Large Problems

- High performance, highly parallelized and optimized ways of solving **unconstrained** optimization problems
  - Using variants of Gradient Descent

Machine Learning is a Natural Way of Solving Large Problems

- High performance, highly parallelized and optimized ways of solving **unconstrained** optimization problems
  - Using variants of Gradient Descent

- Challenges to ML methods for Oblivious Routing:
  - Incorporating the **constraints**
  - Representing and solving for **infinitely many** traffic matrices

NYU TANDON NOKIA Bell Labs

Machine Learning is a Natural Way of Solving Large Problems

- High performance, highly parallelized and optimized ways of solving **unconstrained** optimization problems
  - Using variants of Gradient Descent
- Challenges to ML methods for Oblivious Routing:
  - Incorporating the **constraints**
  - Representing and solving for **infinitely many** traffic matrices
- Can we use this toolset to solve our complex routing problem?

Machine Learning is a Natural Way of Solving Large Problems

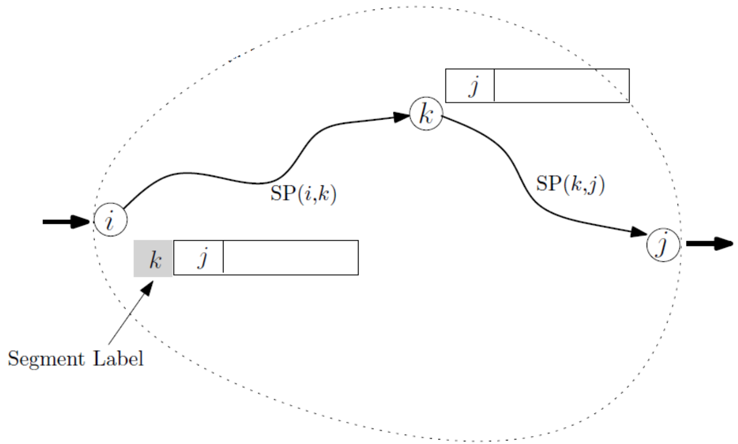
- High performance, highly parallelized and optimized ways of solving **unconstrained** optimization problems
  - Using variants of Gradient Descent
- Challenges to ML methods for Oblivious Routing:
  - Incorporating the **constraints**
  - Representing and solving for **infinitely many** traffic matrices
- Can we use this toolset to solve our complex routing problem?
  - Yes!

How?

How do we engineer traffic?

How do we engineer traffic?

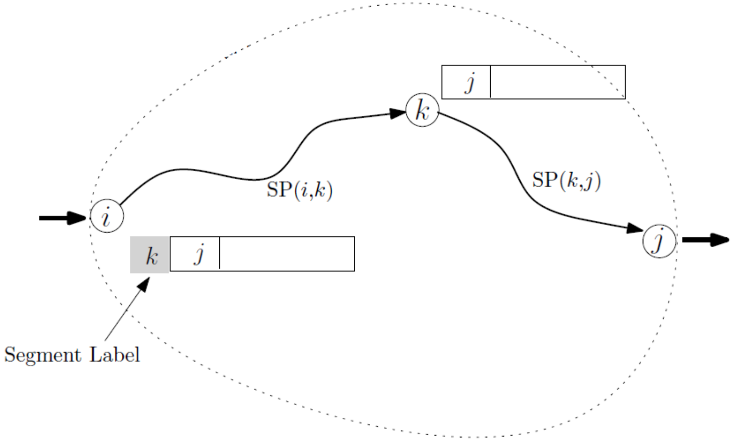
- We need flexibility with split choices:
  - **Segment Routing** is a recent and widely deployed non-shortest path technique



How do we engineer traffic?

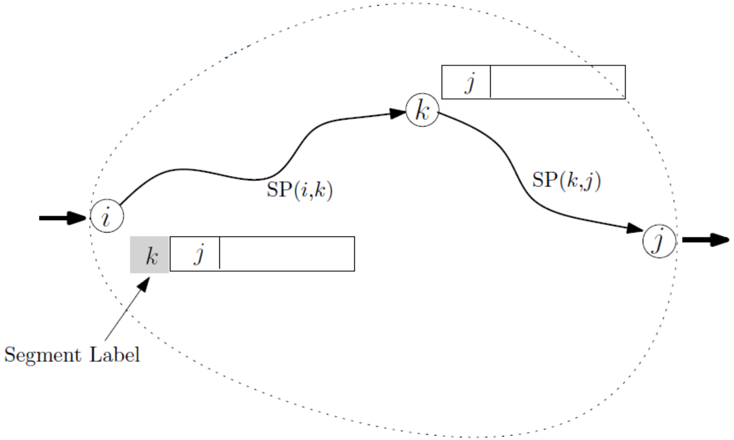
- We need flexibility with split choices:
  - **Segment Routing** is a recent and widely deployed non-shortest path technique

- 2-Segment Routing: $i \rightarrow \mathrm{k} \rightarrow j$
  - Choose an intermediate node k
  - Set what fraction of $i \rightarrow j$ should go through k: α_{ij}^{k}
  - $\sum_k \alpha_{ij}^{k} = 1\,,\ \forall i, j$



How do we engineer traffic?

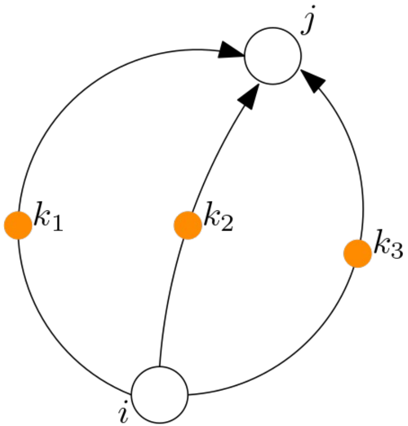
- We need flexibility with split choices:
  - **Segment Routing** is a recent and widely deployed non-shortest path technique
- 2-Segment Routing: $i \to \mathrm{k} \to j$
  - Choose an intermediate node k
  - Set what fraction of $i \to j$ should go through k: α_{ij}^{k}
  - $\sum_k \alpha_{ij}^{k} = 1\,,\ \forall i, j$
- We embed constraints into the objective function!
  - Segment Routing renders paths enumerable!
  - *Softmax* function trick!
    (see paper for details!)



NYU TANDON NOKIA Bell Labs

How to Solve an Oblivious Routing Problem Using ML?

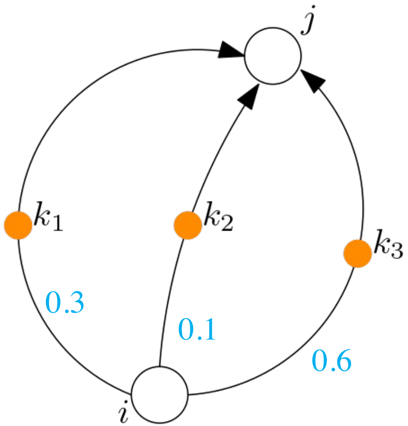
- Follow an adversarial approach



How to Solve an Oblivious Routing Problem Using ML?

- ❑ Follow an adversarial approach

Start with random split fractions α_{ij}^{k}

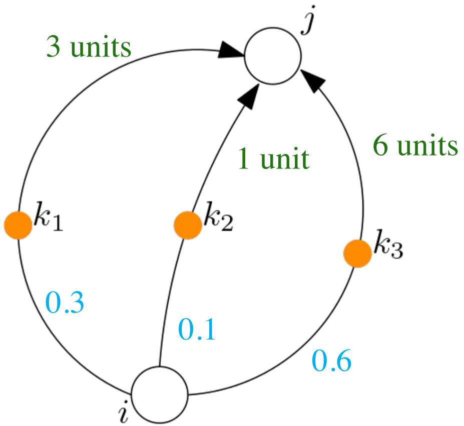


NYU TANDON NOKIA Bell Labs

How to Solve an Oblivious Routing Problem Using ML?

- Follow an adversarial approach

Start with random split fractions α_{ij}^{k}

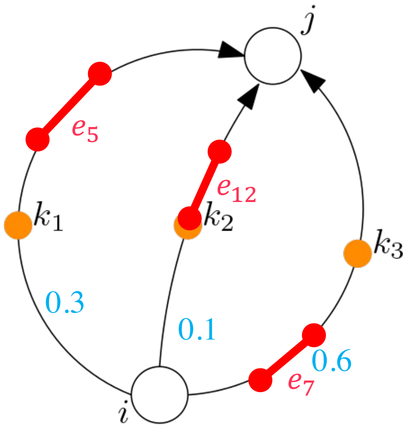


Traffic $i \rightarrow j : 10\ units$

How to Solve an Oblivious Routing Problem Using ML?

- Follow an adversarial approach

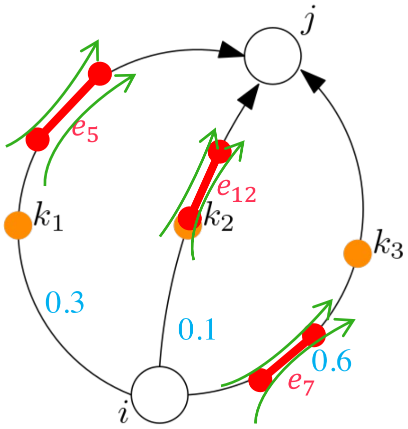
Start with random split fractions α_{ij}^k



How to Solve an Oblivious Routing Problem Using ML?

- Follow an adversarial approach

Start with random split fractions α_{ij}^{k}



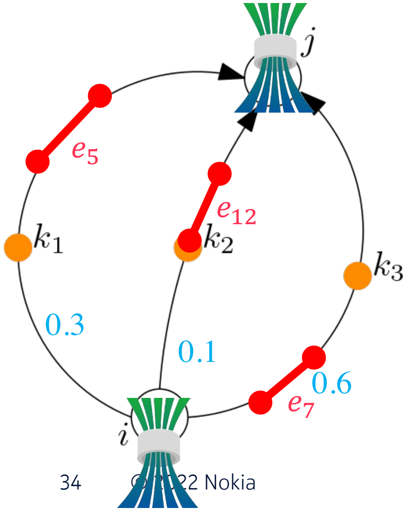
How to Solve an Oblivious Routing Problem Using ML?

- Follow an adversarial approach

Start with random split fractions α_{ij}^{k}

Adversarial Step

Generate the worst possible traffic matrices $[t_{ij}]^{(l)}$ for the current routing fractions α_{ij}^{k} and add to the set of bad matrices $\mathcal{L}$



Generate a traffic matrix corresponding to each edge:

- from the hose constraints
- that maximizes the load

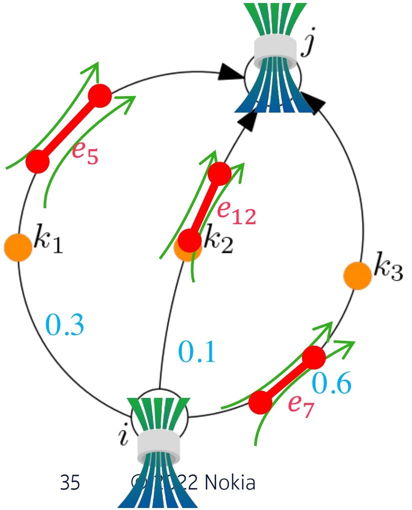
How to Solve an Oblivious Routing Problem Using ML?

- Follow an adversarial approach

Start with random split fractions α_{ij}^k

Adversarial Step

Generate the worst possible traffic matrices $[t_{ij}]^{(l)}$ for the current routing fractions α_{ij}^k and add to the set of bad matrices $\mathcal{L}$



Generate a traffic matrix corresponding to each edge:

- from the hose constraints
- that maximizes the load

How to Solve an Oblivious Routing Problem Using ML?

- Follow an adversarial approach

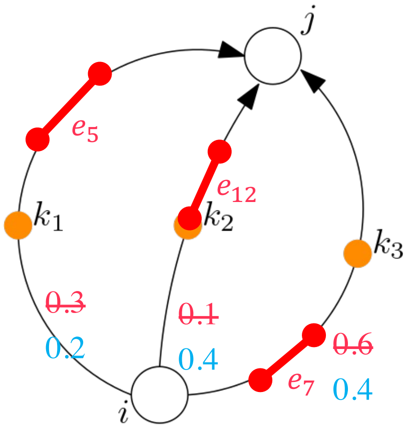
Start with random split fractions α_{ij}^{k}

Adversarial Step

Generate the worst possible traffic matrices $[t_{ij}]^{(l)}$ for the current routing fractions α_{ij}^{k} and add to the set of bad matrices $\mathcal{L}$

Optimization Step

Find the best routing split fractions α_{ij}^{k} for all traffic matrices in set $\mathcal{L}$



Update routing fractions of each $i \rightarrow j$ pair

How to Solve an Oblivious Routing Problem Using ML?

- Follow an adversarial approach

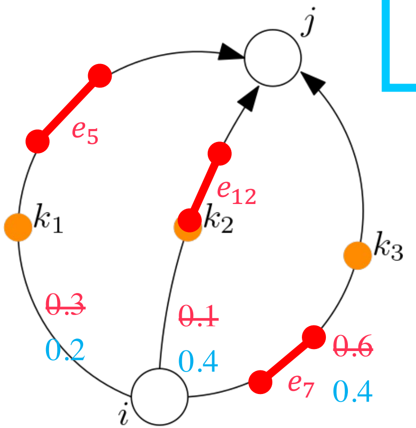
Start with random split fractions α_{ij}^{k}

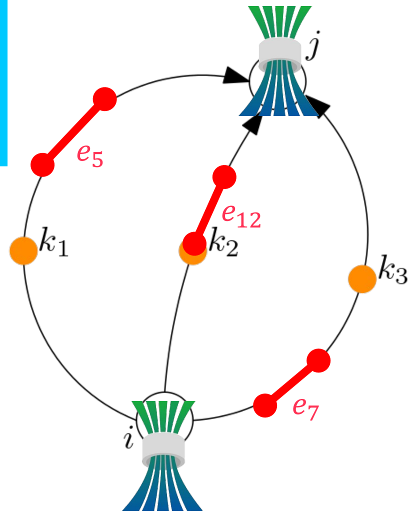
Adversarial Step

Generate the worst possible traffic matrices $[t_{ij}]^{(l)}$ for the current routing fractions α_{ij}^{k} and add to the set of bad matrices $\mathcal{L}$

Optimization Step

Find the best routing split fractions α_{ij}^{k} for all traffic matrices in set $\mathcal{L}$





How to Solve an Oblivious Routing Problem Using ML?

- Follow an adversarial approach

Start with random split fractions α_{ij}^k

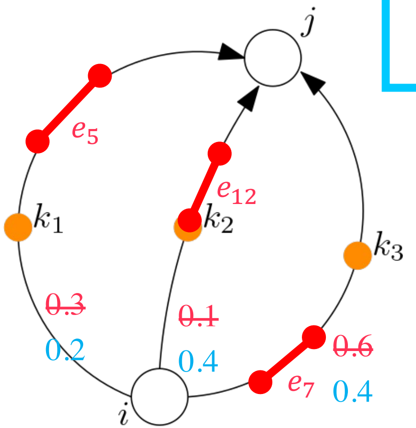
Adversarial Step

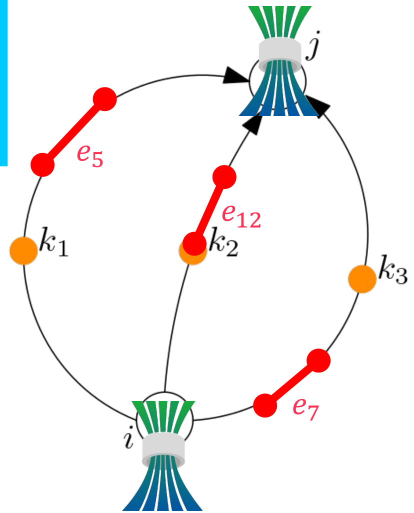
Generate the worst possible traffic matrices $[t_{ij}]^{(l)}$ for the current routing fractions α_{ij}^k and add to the set of bad matrices $\mathcal{L}$

Optimization Step

Find the best routing split fractions α_{ij}^k for all traffic matrices in set $\mathcal{L}$

Stop when performance upper and lower bounds converge





NYU | TANDON NOKIA Bell Labs

How to Solve an Oblivious Routing Problem Using ML?

- Follow an adversarial approach

Start with random split fractions α_{ij}^k

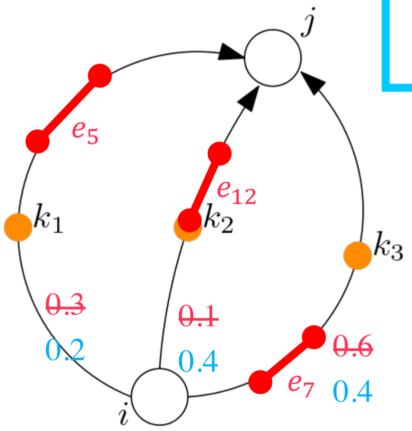
Adversarial Step

Generate the worst possible traffic matrices $[t_{ij}]^{(l)}$ for the current routing fractions α_{ij}^k and add to the set of bad matrices $\mathcal{L}$

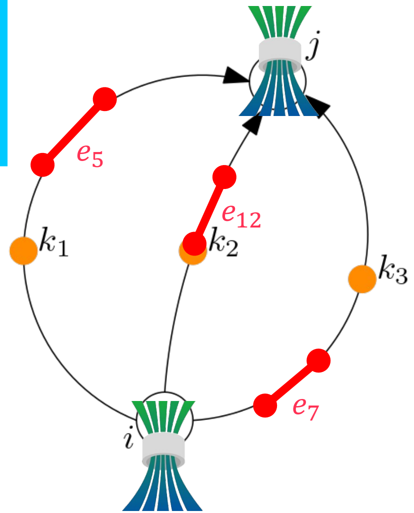
Optimization Step

Find the best routing split fractions α_{ij}^k for all traffic matrices in set $\mathcal{L}$

Stop when performance upper and lower bounds converge



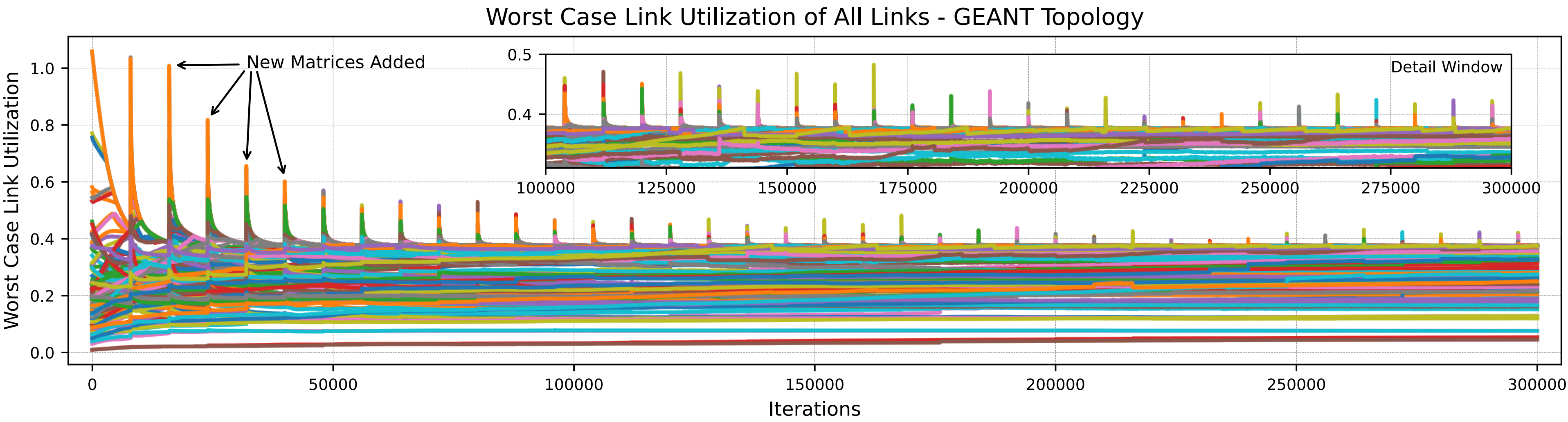
- Optimization step is solved with gradient descent.



Results

The Adversarial Approach converges to the optimal solution

- Worst case utilization of each link on the network is depicted as split fractions are tuned.
- Adversarial traffic matrices are periodically added.



NYU TANDON NOKIA Bell Labs

We Can Solve Large Scale Problems Using the ML Approach

- The Hose-Oblivious Routing problem can be solved using the Adversarial Learning Approach.

We Can Solve Large Scale Problems Using the ML Approach

- The Hose-Oblivious Routing problem can be solved using the Adversarial Learning Approach.

- The method scales well with network sizes and is able to handle hundreds of nodes

We Can Solve Large Scale Problems Using the ML Approach

- The Hose-Oblivious Routing problem can be solved using the Adversarial Learning Approach.

- The method scales well with network sizes and is able to handle hundreds of nodes

- For each solution obtained, we have a performance guarantee for the worst-case scenario.

We Can Solve Large Scale Problems Using the ML Approach

- ❑ The Hose-Oblivious Routing problem can be solved using the Adversarial Learning Approach.
- ❑ The method scales well with network sizes and is able to handle hundreds of nodes
- ❑ For each solution obtained, we have a performance guarantee for the worst-case scenario.
- ❑ Our method is compatible with:
  - ❑ Legacy shortest path routing extensions (e.g., MPLS)
  - ❑ Software Defined Networking

We Can Solve Large Scale Problems Using the ML Approach

- The Hose-Oblivious Routing problem can be solved using the Adversarial Learning Approach.
- The method scales well with network sizes and is able to handle hundreds of nodes
- For each solution obtained, we have a performance guarantee for the worst-case scenario.
- Our method is compatible with:
  - Legacy shortest path routing extensions (e.g., MPLS)
  - Software Defined Networking
- Also available in our paper:
  - Two computationally simple relaxations of this problem –providing bounding solutions

NOKIA Bell Labs NYU | TANDON

Thank You

Oblivious Routing using Learning Methods

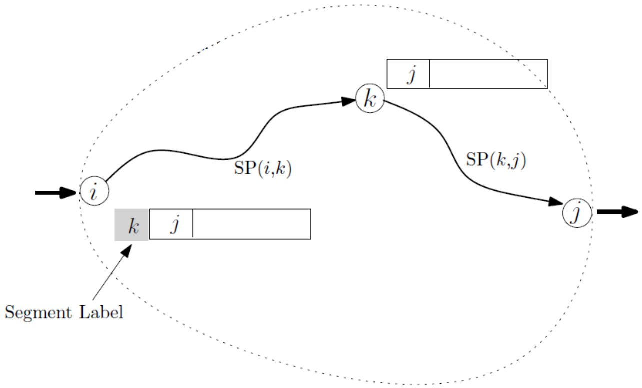
Ufuk Usubutun

usubutun@nyu.edu

Backup Slides

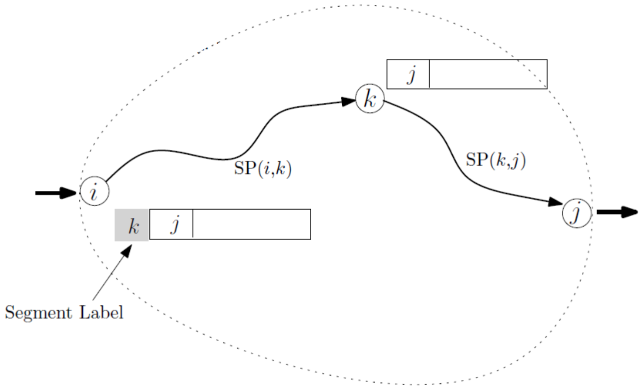
How do we incorporate 2-Segment Routing into the problem formulation?

- 2-Segment Routing: $i \rightarrow \mathrm{k} \rightarrow j$
  - Choose an intermediate node k
  - Set what fraction of $i \rightarrow j$ should go through k
  - $\sum_k \alpha_{ij}^k = 1\,,\ \forall i, j$

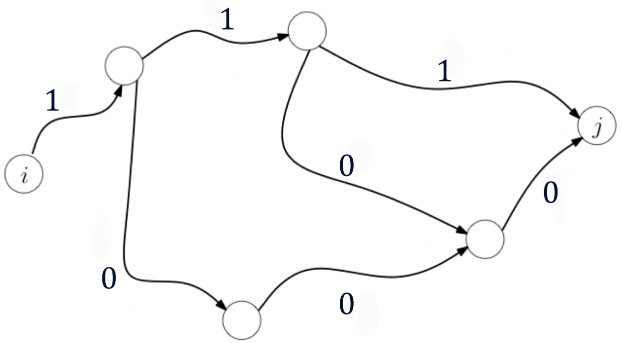


How do we incorporate 2-Segment Routing into the problem formulation?

- 2-Segment Routing: $i \rightarrow \mathrm{k} \rightarrow j$
  - Choose an intermediate node k
  - Set what fraction of $i \rightarrow j$ should go through k
  - $\sum_k \alpha_{ij}^k = 1\,,\ \forall i, j$

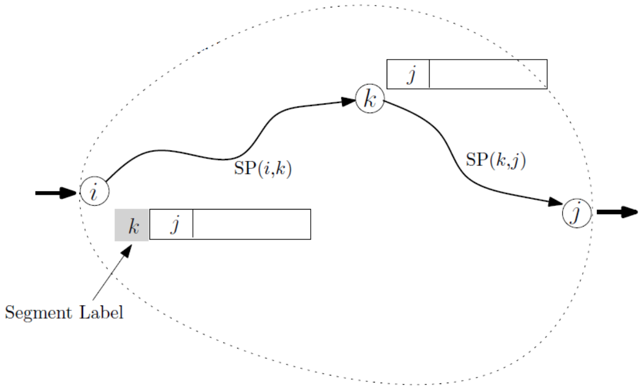


- Define an indicator function $f_{ij}(e) \in \{0,1\}$ where e is an edge
  - If a link e is on the shortest path from i to j: set $f_{ij}(e)$ to 1
  - Otherwise: set $f_{ij}(e)$ to 0



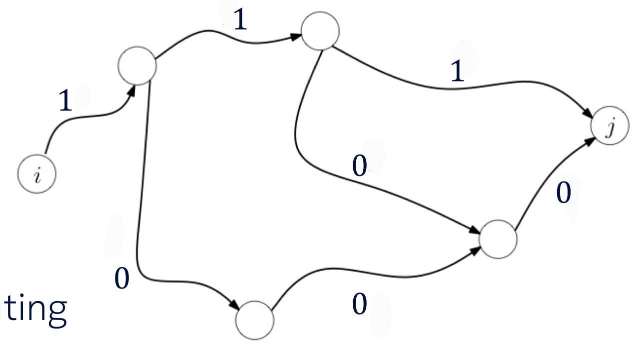
How do we incorporate 2-Segment Routing into the problem formulation?

- 2-Segment Routing: $i \rightarrow \mathrm{k} \rightarrow j$
  - Choose an intermediate node k
  - Set what fraction of $i \rightarrow j$ should go through k
  - $\sum_k \alpha_{ij}^k = 1\ ,\ \forall i, j$



- Define an indicator function $f_{ij}(e) \in \{0,1\}$ where e is an edge
  - If a link e is on the shortest path from i to j: set $f_{ij}(e)$ to 1
  - Otherwise: set $f_{ij}(e)$ to 0

- Define $g_{ij}^k(e) = f_{ik}(e) + f_{kj}(e)$ to accommodate 2-Segment Routing



NYU TANDON NOKIA Bell Labs

ECMP Extension

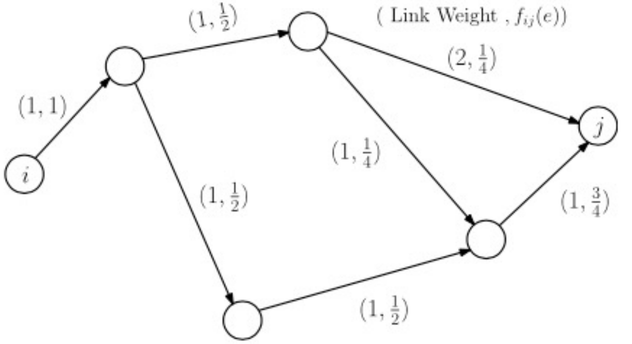
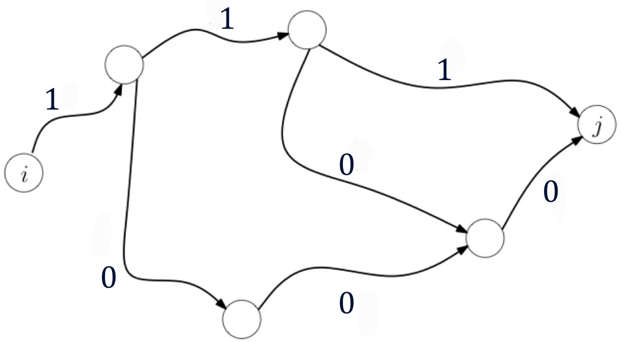
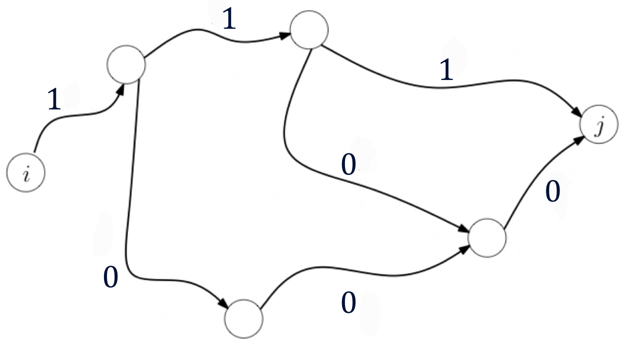


Fig. 1. Definition of $f_{ij}(e)$ with ECMP. The first number next to the link represents the link weight and the second number is $f_{ij}(e)$. The shortest path length is 4 and there are three shortest paths.



ECMP Extension

- Define an indicator function $f_{ij}(e) \in \{0,1\}$ where e is an edge
  - If a link e is on the shortest path from i to j: set $f_{ij}(e)$ to 1
  - Otherwise: set $f_{ij}(e)$ to 0



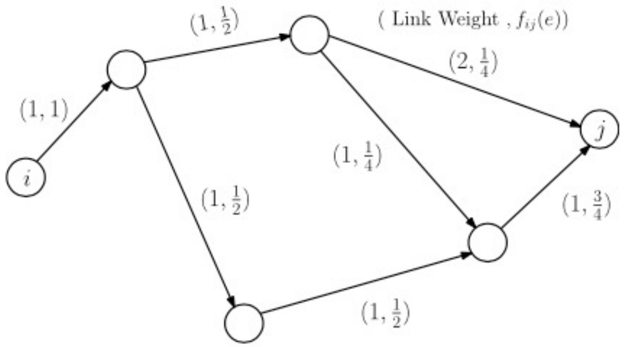
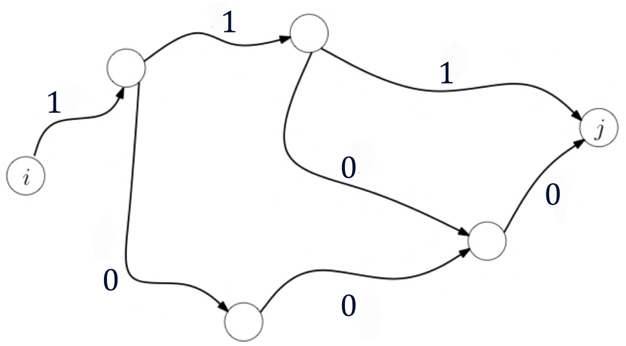


Fig. 1. Definition of $f_{ij}(e)$ with ECMP. The first number next to the link represents the link weight and the second number is $f_{ij}(e)$. The shortest path length is 4 and there are three shortest paths.

ECMP Extension

- Define an indicator function $f_{ij}(e) \in \{0,1\}$ where e is an edge
  - If a link e is on the shortest path from i to j: set $f_{ij}(e)$ to 1
  - Otherwise: set $f_{ij}(e)$ to 0



- ECMP Extension

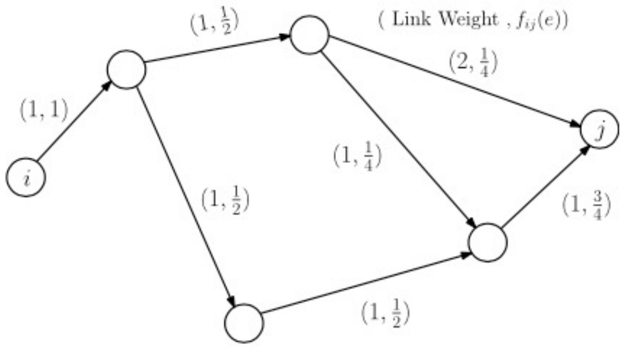


Fig. 1. Definition of $f_{ij}(e)$ with ECMP. The first number next to the link represents the link weight and the second number is $f_{ij}(e)$. The shortest path length is 4 and there are three shortest paths.

Formulating a 2-Segment Routing Problem

Formulating a 2-Segment Routing Problem

- Total traffic from i to j traveling over node k:

$$t_{ij}\alpha_{ij}^{k}$$

NYU | TANDON NOKIA Bell Labs

Formulating a 2-Segment Routing Problem

- Total traffic from i to j traveling over node k:

$$t_{ij}\alpha_{ij}^{k}$$

- The load on a link e caused by traffic from i to j thru k:

$$t_{ij}\alpha_{ij}^{k}g_{ij}^{k}(e)$$

Formulating a 2-Segment Routing Problem

- Total traffic from i to j traveling over node k:

$$t_{ij}\alpha_{ij}^{k}$$

- The load on a link e caused by traffic from i to j thru k:

$$t_{ij}\alpha_{ij}^{k}g_{ij}^{k}(e)$$

Formulating a 2-Segment Routing Problem

- Total traffic from i to j traveling over node k:

$$t_{ij}\alpha_{ij}^{k}$$

- The load on a link e caused by traffic from i to j thru k:

$$t_{ij}\alpha_{ij}^{k}g_{ij}^{k}(e)$$

- The total load on a link e:

$$\sum_{ijk} t_{ij}\alpha_{ij}^{k}g_{ij}^{k}(e)$$

How to Solve an Oblivious Routing Problem Using ML?

- Follow an adversarial approach

How to Solve an Oblivious Routing Problem Using ML?

- Follow an adversarial approach

Start with random split fractions α_{ij}^{k}

How to Solve an Oblivious Routing Problem Using ML?

- Follow an adversarial approach

Start with random split fractions α_{ij}^k

→

Adversarial Step

Generate the worst possible traffic matrices $[t_{ij}]^{(l)}$ for the current routing fractions α_{ij}^k and add to the set of bad matrices $\mathcal{L}$

$$\mathcal{T}(\vec{\mathcal{R}},\vec{\mathcal{C}}) = \left\{ [t_{ij}] \,\middle|\, \sum_{j\neq i} t_{ij} \leq R_i \text{ and } \sum_{j\neq i} t_{ji} \leq C_i \; \forall\, i \right\}$$

How to Solve an Oblivious Routing Problem Using ML?

- Follow an adversarial approach

Start with random split fractions α_{ij}^{k}

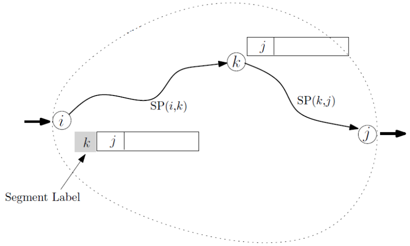
Adversarial Step

Generate the worst possible traffic matrices $[t_{ij}]^{(l)}$ for the current routing fractions α_{ij}^{k} and add to the set of bad matrices $\mathcal{L}$

Optimization Step

Find the best routing split fractions α_{ij}^{k} for all traffic matrices in set $\mathcal{L}$

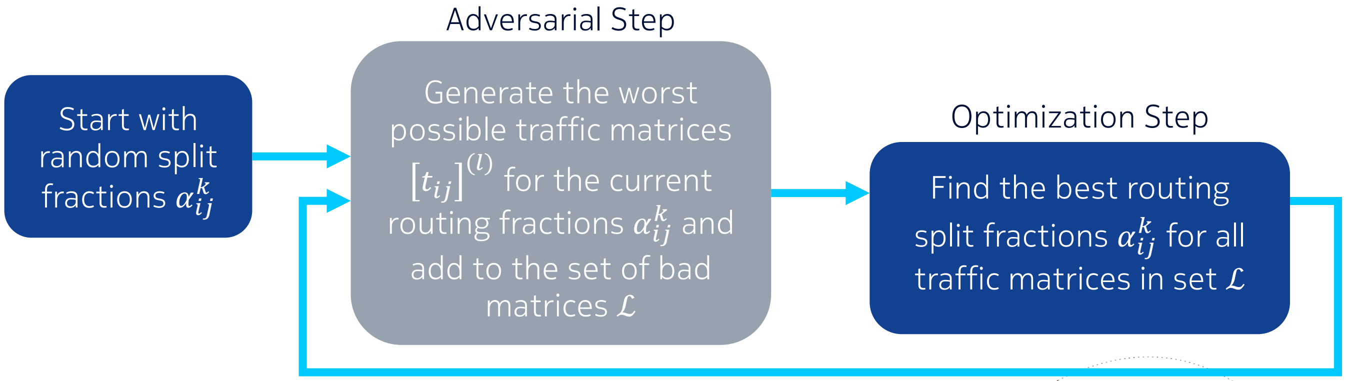
$$\mathcal{T}(\vec{\mathcal{R}},\vec{\mathcal{C}}) = \left\{ [t_{ij}] \,\middle|\, \sum_{j\neq i} t_{ij} \leq R_i \text{ and } \sum_{j\neq i} t_{ji} \leq C_i \;\forall\, i \right\}$$



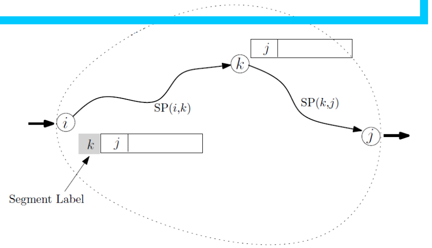
NYU TANDON NOKIA Bell Labs

How to Solve an Oblivious Routing Problem Using ML?

- Follow an adversarial approach



$$\mathcal{T}(\vec{\mathcal{R}}, \vec{\mathcal{C}}) = \left\{ [t_{ij}] \;\middle|\; \sum_{j \neq i} t_{ij} \leq R_i \text{ and } \sum_{j \neq i} t_{ji} \leq C_i \;\forall\, i \right\}$$



NYU TANDON NOKIA Bell Labs

How to Solve an Oblivious Routing Problem Using ML?

- Follow an adversarial approach

Start with random split fractions α_{ij}^{k}

Adversarial Step

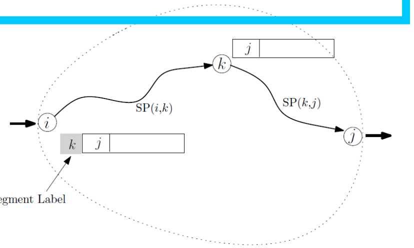
Generate the worst possible traffic matrices $[t_{ij}]^{(l)}$ for the current routing fractions α_{ij}^{k} and add to the set of bad matrices $\mathcal{L}$

Optimization Step

Find the best routing split fractions α_{ij}^{k} for all traffic matrices in set $\mathcal{L}$

Stop when performance upper and lower bounds converge

$$\mathcal{T}(\vec{\mathcal{R}},\vec{\mathcal{C}}) = \left\{ [t_{ij}] \;\middle|\; \sum_{j\neq i} t_{ij} \le R_i \text{ and } \sum_{j\neq i} t_{ji} \le C_i \ \forall\, i \right\}$$



NYU TANDON NOKIA Bell Labs

How to Solve an Oblivious Routing Problem Using ML?

- Follow an adversarial approach

Start with random split fractions α_{ij}^k

Adversarial Step

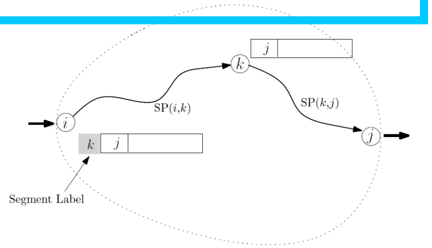
Generate the worst possible traffic matrices $[t_{ij}]^{(l)}$ for the current routing fractions α_{ij}^k and add to the set of bad matrices $\mathcal{L}$

Optimization Step

Find the best routing split fractions α_{ij}^k for all traffic matrices in set $\mathcal{L}$

Stop when performance upper and lower bounds converge

$$\mathcal{T}(\vec{\mathcal{R}},\vec{\mathcal{C}}) = \left\{ [t_{ij}] \,\middle|\, \sum_{j\neq i} t_{ij} \leq R_i \text{ and } \sum_{j\neq i} t_{ji} \leq C_i \;\forall\, i \right\}$$



- Optimization step is solved with gradient descent.

NYU TANDON NOKIA Bell Labs

Formulating a 2-Segment Oblivious Routing Problem

Formulating a 2-Segment Oblivious Routing Problem

- Let the capacity of a given link e is u_e. The optimization problem becomes:

$$\min_{\alpha_{ij}^k} \mu \;\; \text{s.t.}$$

$$\sum_{ijk} t_{ij} \alpha_{ij}^k g_{ij}^k(e) \le \mu u_e \,, \qquad \forall i,j, \forall e, \forall [t_{ij}] \in T(R,C)$$

$$\sum_k \alpha_{ij}^k = 1 \,, \qquad \forall i,j$$

$$\alpha_{ij}^k \ge 0 \,, \qquad \forall i,j,k$$

Formulating a 2-Segment Oblivious Routing Problem

- Let the capacity of a given link e is u_e. The optimization problem becomes:

$$\min_{\alpha_{ij}^k} \mu \;\text{ s.t.}$$

$$\sum_{ijk} t_{ij}\alpha_{ij}^k g_{ij}^k(e) \le \mu u_e\,, \qquad \forall i,j,\forall e,\forall[t_{ij}] \in T(R,C)$$

$$\sum_k \alpha_{ij}^k = 1\,, \qquad \forall i,j$$

$$\alpha_{ij}^k \ge 0\,, \qquad \forall i,j,k$$

$$\mathcal{T}(\vec{\mathcal{R}},\vec{\mathcal{C}}) = \left\{ [t_{ij}] \;\middle|\; \sum_{j\neq i} t_{ij} \le R_i \text{ and } \sum_{j\neq i} t_{ji} \le C_i \;\forall\, i \right\}$$

- How do we work with infinitely many traffic matrices?

How to Solve an Oblivious Routing Problem Using ML?

The Optimization Step

- Reshape the 2-segment routing problem to embed some constraints into the objective function

$$\min_{\alpha_{ij}^k} \mu \ \text{s.t.}$$
$$\sum_{ijk} t_{ij}^{(l)} \alpha_{ij}^k g_{ij}^k(e) \le \mu u_e \,, \qquad \forall i,j, \forall l, \forall e$$
$$\sum_k \alpha_{ij}^k = 1 \,, \qquad \forall i,j$$
$$\alpha_{ij}^k \ge 0 \,, \qquad \forall i,j,k$$

$\Rightarrow$

$$\min_{\alpha_{ij}^k}\{\max_{e,l} \frac{\sum_{ijk} t_{ij}^{(l)} \alpha_{ij}^k g_{ij}^k(e)}{u_e}\} \ \text{s.t.}$$
$$\sum_k \alpha_{ij}^k = 1 \,, \qquad \forall i,j$$
$$\alpha_{ij}^k \ge 0 \,, \qquad \forall i,j,k$$

How to Solve an Oblivious Routing Problem Using ML?
The Optimization Step

- Reshape the 2-segment routing problem to embed some constraints into the objective function

$$\min_{\alpha_{ij}^k} \mu \ \text{s.t.}$$
$$\sum_{ijk} t_{ij}^{(l)} \alpha_{ij}^k g_{ij}^k(e) \leq \mu u_e\,, \qquad \forall i,j,\forall l,\forall e$$
$$\sum_k \alpha_{ij}^k = 1\,, \qquad \forall i,j$$
$$\alpha_{ij}^k \geq 0\,, \qquad \forall i,j,k$$

$$\min_{\alpha_{ij}^k}\{\max_{e,l} \frac{\sum_{ijk} t_{ij}^{(l)} \alpha_{ij}^k g_{ij}^k(e)}{u_e}\} \ \text{s.t.}$$
$$\sum_k \alpha_{ij}^k = 1\,, \qquad \forall i,j$$
$$\alpha_{ij}^k \geq 0\,, \qquad \forall i,j,k$$

- Use the *softmax* function to keep solutions in the feasible set.

$$\alpha_{ij}^k = SM(y_{ij}^k) = \frac{e^{y_{ij}^k}}{\sum_k e^{y_{ij}^k}}, \qquad \forall i,j$$

How to Solve an Oblivious Routing Problem Using ML?
The Optimization Step

- Reshape the 2-segment routing problem to embed some constraints into the objective function

$$\min_{\alpha_{ij}^k} \mu \;\; \text{s.t.}$$
$$\sum_{ijk} t_{ij}^{(l)} \alpha_{ij}^k g_{ij}^k(e) \leq \mu u_e\,, \qquad \forall i,j,\forall l,\forall e$$
$$\sum_k \alpha_{ij}^k = 1\,, \qquad \forall i,j$$
$$\alpha_{ij}^k \geq 0\,, \qquad \forall i,j,k$$

$$\min_{\alpha_{ij}^k}\{\max_{e,l} \frac{\sum_{ijk} t_{ij}^{(l)} \alpha_{ij}^k g_{ij}^k(e)}{u_e}\} \;\; \text{s.t.}$$
$$\sum_k \alpha_{ij}^k = 1\,, \qquad \forall i,j$$
$$\alpha_{ij}^k \geq 0\,, \qquad \forall i,j,k$$

- Use the *softmax* function to keep solutions in the feasible set.

$$\alpha_{ij}^k = SM(y_{ij}^k) = \frac{e^{y_{ij}^k}}{\sum_k e^{y_{ij}^k}}, \qquad \forall i,j$$

$$\min_{\alpha_{ij}^k}\{\max_{e,l} \frac{\sum_{ijk} t_{ij}^{(l)} SM(y_{ij}^k) g_{ij}^k(e)}{u_e}\} \;\; \text{s.t.}$$

Generate worst case matrices:
The Adversarial Step

Given a choice of split fractions α_{ij}^{k}

Generate one worst case traffic matrix for each link e

Append all new matrices to set $\mathcal{L}$

$$[\mathrm{t}_{ij}]^{(l)} = arg\,max_{[t_{ij}]} \sum_{ijk} t_{ij} \alpha_{ik}^{k} g_{ij}^{k}(e) \quad \text{s.t.}$$

$$\sum_{i} t_{ij} \le C_j \,, \qquad \forall j$$

$$\sum_{j} t_{ij} \le R_i \,, \qquad \forall i$$

$$t_{ij} \ge 0 \,, \qquad \forall i, j$$

NOKIA Bell Labs

NYU | TANDON