The Robust Network Design Problem

Joint work with:
N. Goyal (Microsoft India), N. Olver (Vrije Univ. Amsterdam)
and
C. Chekuri (UIUC), G. Oriolo (Rome), M.G. Scutella (Pisa)
Alex Fréchette (UBC), Marina Thottan and Peter Winzer
(Bell Labs)
Overview

I What is Network Design?

II Uncertainty and Network Design

III Routing Models with Uncertain or Changing Traffic

IV The VPN Problem
I. Minimum Cost Network Design

Input:

- a supply graph $G$ with a
- per-unit cost $c(e)$ for each edge $e \in E(G)$
- a symmetric demand matrix $D_{ij}$ representing the demand between nodes $i, j$

Assumption: our networks and flows are undirected.
**Minimum Cost Network Design**

**Input:**
- a supply graph $G$ with a
- per-unit cost $c(e)$ for each edge $e \in E(G)$
- a symmetric demand matrix $D_{ij}$ representing the demand between nodes $i, j$

**Output:** Minimum cost edge capacity reservation $x(e)$ that supports the simultaneous routing of all demands $D_{ij}$. 
Minimum Cost Network Design

\[ x(e) = 2 \]
Network Design with Side Constraints

Normally have side constraints:

- edge capacities
- routing constraints (unsplittable or conﬂuent ﬂow)
- buy-at-bulk costs
- node costs
- resilience requirements etc.

\[ x(e) = 2 \]
II. Network Design with Uncertain Demand

Don't know a single demand matrix - we have a collection of possible "valid" demand matrices:

**The Universe** $\mathcal{U}$

Take $\mathcal{U}$ as a convex region of demands $(D_{ij})$. 
An edge capacity vector is **ROBUST** if it has enough network capacity to “support” each demand matrix

\[(D_{ij}) \in \mathcal{U}.\]

**Goal:** Find a **robust** edge capacity vector \( x(e) \) that minimizes

\[\sum_e c(e)x(e).\]
Some “Known” Universes

1. **Forecasts on Future Traffic**

Given traffic “forecasts” \( D^i \) for \( i = 1, 2, \ldots, s \):

\[
\mathcal{U} = \text{conv}\{D^1, D^2, \ldots, D^s\}.
\]
Some “Known” Universes

1. **Forecasts on Future Traffic**

Given target matrix $D$ and other “forecasts” $D^i$ for $i = 1, 2, \ldots, s$:

$$\mathcal{U} = \text{conv}\{D, D^1, D^2, \ldots, D^s\}.$$ 

2. **(Räcke) “G-Demands”**

$$\mathcal{U} = \{D : D \text{ can be routed in } G\}$$
3. “Unit Ball” Demand Polyhedra

Consider a unit ball around $\tilde{D}_{ij}$ defined by some norm $L$.

$$\mathcal{U} = P(\tilde{D}, L) = \{D : ||D - \tilde{D}||_L \leq 1\}.$$ 

- $l_p$ norms give rise to classes of concave cost flow problems.
- The $l_2$-norm and ellipsoidal balls were considered by Ben Tal and Nemirovski / Belotti and Pinar, to study associated stochastic optimization problems.

Design cheapest network such that the probability a link’s capacity is exceeded is at most $p(e)$. 
4. **The Hose Model** (Fingerhut et al. 1997, Duffield et al. 1999)

**Bounds on Injected Traffic**

Given a subset $W$ of terminals/nodes/vertices that wish to communicate.

**The Hose Polytope**

$$
U = \{ \text{symmetric } (D_{ij}) : \sum_j D_{ij} \leq 1, \sum_i D_{ij} \leq 1 \ \forall i, j \in W \}.
$$
Given marginal demands $D_i$ for each node $i$

$\mathcal{U} = \{(D_{ij}) : \sum_j D_{ij} \leq D_i \text{ and } \sum_j D_{ji} \leq D_i \quad \forall i\}$. 
III. Routing Models for Uncertain Demands

**ROBUSTNESS:** enough network capacity to “support” each demand matrix in the universe $\mathcal{U}$ (e.g., the Hose Polytope).

“support”? 

Most basic question: What are we allowed to do when demand patterns change?
(Fully) Dynamic Routing: Demands = AB, AC
Dynamic Routing: Demands AC, AD

Reroute the demand AC
Partly Dynamic Control Plane

Leave existing traffic alone.
Partly Dynamic Control Plane

One must leave existing traffic alone.

Neither model is realistic in modern data networks.
- Can’t change routings on the fly.
- Do not have the global picture of traffic patterns $D_{i,j}$. 
A Third Way: Oblivious Packet Routing

“...the route taken by each packet is determined entirely by itself. The other packets can only influence the rate at which the route is traversed”

Valiant ’81
Oblivious Routing

Specify a **Routing Template** ahead of time, so routing is independent of current conditions in the network.

For each pair of nodes $i, j$ we designate an $i - j$ path $P_{ij}$

**Interpretation:** If in the future we handle some traffic matrix $D$, then we should send $D_{ij}$ units of flow down path $P_{ij}$. 
Fractional Oblivious Routing

Specify a **Routing Template** ahead of time, so routing is independent of current conditions in the network.

For each pair of nodes $i, j$ we have designated flow values $f(P)$ to $i - j$ paths $P$ such that:

$$\sum_{P \text{ joins } i \text{ and } j} f(P) = 1$$
Fractional Oblivious Routing

Specify a **Routing Template** ahead of time, so routing is independent of current conditions in the network.

For each pair of nodes $i, j$ we have designated flow values $f(P)$ to $i - j$ paths $P$ such that:

$$
\sum_{P \text{ joins } i \text{ and } j} f(P) = 1
$$

**Interpretation:** If in the future we handle some traffic matrix $D$, then we should send $f(P)D_{ij}$ flow down path $P$. 
Oblivious Routing

Specify a Routing Template ahead of time, so routing is independent of current conditions in the network.

For each pair of nodes $i, j$ we have designated flow values $f(P)$ to paths $P$ joining $i$ and $j$ such that:

$$\sum_P f(P) = 1$$

Single Path Routing (SPR): $f(P) = 0$ or 1. (Template $\mathcal{T} = (P_{ij})$)

Multi-Path Routing (MPR): $f(P)$ allowed to be fractions.
Examples:

Tree Routing
Randomized Load Balancing (Valiant)
IV. The VPN Problem

Hose Polytope = Fractional $D$-Matching Polytope:

$$\mathcal{U} = \{ (D_{ij}) : \sum_j D_{ij} \leq 1, \sum_j D_{ji} \leq 1 \quad \forall i \in W \}$$

Input: Undirected $G = (V, E)$, edge costs $c(e)$, and a hose polytope $\mathcal{U}$ defined for terminals $W \subseteq V$.

Find: Minimum Cost Edge-Capacity that supports All Demands in $\mathcal{U}$ via *Single-Path Oblivious Routing* (SPR).
IV. The VPN Problem

**Input:** an undirected network $G = (V, E)$ with edge costs $c(e)$, and a hose polytope $\mathcal{U}$ for terminals $W$.

Consider a single-path routing (SPR) template $\mathcal{T} = (P_{ij})$ for each pair of terminals $i, j$.

How much capacity is needed on edge $e$?

$$u_{\mathcal{T}}(e) = \max_{D \in \mathcal{U}} \sum_{P_{ij} : e \in P_{ij}} D_{ij}$$
IV. The VPN Problem

**Input:** an undirected network $G = (V, E)$ with edge costs $c(e)$, and a hose polytope $U$ for terminals $W$.

Consider a single-path routing template $\mathcal{T} = (P_{ij})$ for each pair of terminals $i, j$. How much capacity is needed on edge $e$:

$$u_\mathcal{T}(e) = \max_{D \in U} \sum_{P_{ij}: e \in P_{ij}} D_{ij}$$

**Find:** a template $\mathcal{T}$ that minimizes $\sum_e c(e) u_\mathcal{T}(e)$.

Call the resulting capacities $u(e)$ a VPN.

Note: The capacities in a VPN may be fractional even though template is integral.
The VPN Theorem

There is an optimal VPN induced by a template $P_{i,j}$ only using edges in some fixed tree $T$.

I.e., there is an optimal VPN induced by a Tree-Routing Template.
The VPN Theorem

There is an optimal VPN induced by a template $P_{ij}$ only using edges in some fixed tree $T$.

I.e., there is an optimal VPN induced by a Tree-Routing Template.

**Corollary 1.** An optimal VPN can be computed in polytime.

Not obvious this is true even if $G$ is a ring.
But in fact the optimal MPR solution is even a tree!
(Hurkens, Keijsper, Stougie 2007)
(and reproved in 2007)
The MPR/Fractional Solution is a Tree in Rings?

Must support any matching
\{123\}, \{13\}, \{23\}

Cost = 2

\[
\begin{array}{c}
\frac{1}{2} \\
\frac{1}{2} \\
\frac{1}{2}
\end{array}
\]

Cost = \frac{3}{2}
NOT a Counterexample

Must support any matching: $\{123, \{13\}, \{23\}\}$

Cost $= 2$

Must support the fractional matching:

$\frac{1}{2}$

$\frac{1}{2}$

$\frac{1}{2}$

$\Rightarrow$ NOT ENOUGH CAPACITY
Can we compute the Best Tree-Template?

Let $\text{OPT}$ be the optimal cost of a VPN.

Let $\text{VPN-\textsc{tree}}$ denote the optimal capacity cost if we are only allowed tree-templates.

**Theorem:** (Fingerhut, Suri, Turner 1997, Gupta, Kleinberg, Kumar, Rastogi, Yener 2001)

$$\text{VPN-\textsc{tree}} \leq 2 \text{OPT}$$
Step 1. Given a fixed tree, how much capacity is needed?

\[ \min \{ D_{left}, D_{right} \} \]
Step 2. Find the “center” of a tree

\[ \leq \sum \frac{D_v}{2} \]
There is enough capacity for terminals to route to the Center.

**Corollary.** Terminals should route on shortest paths
⇒ find best VPN-tree by computing $n$ shortest path trees.
Alternative Oblivious Template: HUB Routing
Algorithm to Find Optimal Tree-VPN

For each node $v \in V$, find the shortest path routing tree $T_v$ rooted at $v$.

Route one unit of flow from each terminal in $W$ to $v$ on $T_v$. Let $C(v)$ be the cost.

The optimal Tree-VPN is obtained from using hub routing to $v$ where $C'(v)$ has the minimum cost.
Finding the Optimal Tree Template

For each node $v \in V$, find the shortest path routing tree $T_v$ rooted at $v$.

Route one unit of flow from each terminal in $W$ to $v$ on $T_v$. Let $C(v)$ be the cost.

The optimal Tree-VPN is obtained from using hub routing to $v$ where $C'(v)$ has the minimum cost.

But Why is this Tree Routing the best VPN?

That is too complicated, but we can show that it is at most $2 \text{ OPT}$.
Observation 1. We Must Route Uniform Multiflow

Note that the demand matrix

$$D_{ij} = \frac{1}{|W|} \quad \forall \ i, j \in W$$

lies within the hose universe.
Observation 1. We Must Route Uniform Multiflow

Note that the demand matrix

\[ \hat{D}_{ij} = \frac{1}{|W|} \quad \forall \ i, j \in W \]

lies within the hose universe.

\[ \Rightarrow \text{OPT} \geq \text{minimum cost flow for } \hat{D} \]
Observation 1. We Must Route Uniform Multiflow

Note that the demand matrix

\[ \hat{D}_{ij} = \frac{1}{|W|} \quad \forall \ i, j \in W \]

lies within the hose universe.

\[ \Rightarrow \text{OPT} \geq \text{minimum cost flow for } \hat{D} \]

The cheapest we could ever route this single demand matrix would be to send each $ij$-flow on a shortest $ij$-path.

\[ \Rightarrow \text{OPT} \geq \text{shortest path routing of } \hat{D} \]
The cost of the flows to a single terminal $i \in W$?

We can view this as the shortest path flows to $i$, i.e.,

$$
\frac{1}{|W|} C(i)
$$

$C(i)$ is the cost of sending one unit from each $j \in W$ to $i$. 
Pay Twice for each \( ij \)

Consider the cost of adding all these capacities single-terminal flows

\[
\frac{1}{|W|} \sum_{i \in W} C(i)
\]

This is like reserving capacity \( \frac{1}{|W|} \) from \( i \) to \( j \) and from \( j \) to \( i \), for each \( i, j \).

I.E, it is like reserving capacity \( \frac{2}{|W|} \) on each shortest \( ij \)-path. This is at most \( 2 \text{ OPT} \) since it is twice the cost to route the uniform matrix.
Pay Twice for each $ij$

The cost of this capacity

$$\frac{1}{|W|} \sum_{i \in W} C(i) \leq 2OPT$$

Choose the terminal $v \in W$ such that $C(v)$ is minimized. One easily checks that

$$C(v) \leq \frac{1}{|W|} \sum_{i \in W} C(i) \leq 2OPT$$
Pay Twice for each \( ij \)

The cost of this capacity

\[
\frac{1}{|W|} \sum_{i \in W} C(i) \leq 2OPT
\]

Choose the terminal \( v \in W \) such that \( C(v) \) is minimized. One easily checks that

\[
C(v) \leq \frac{1}{|W|} \sum_{i \in W} C(i) \leq 2OPT
\]

But \( C(v) \) is the cost of a hub routing from \( W \) to \( v \). As we saw, Hub Routings can handle any demand matrix in our universe.

Hence this tree \( T_v \) yields a solution not worse than twice the optimum.