Network Design Tools & Multicommodity Flows

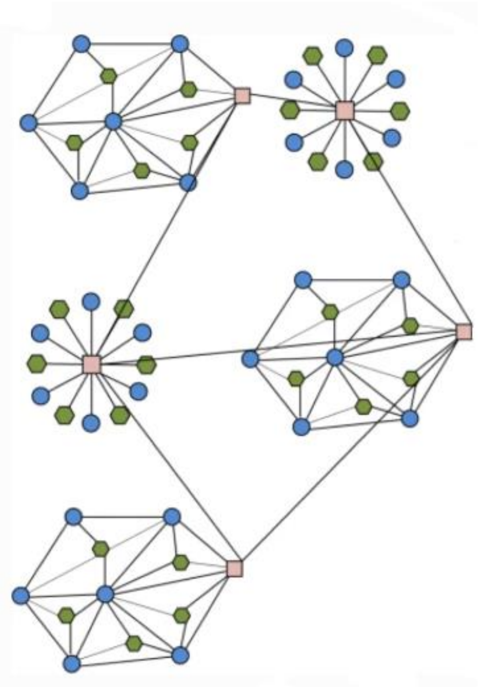
Iraj Saniee
Math & Algorithms Research
Bell Labs, Nokia
Murray Hill, NJ

DANniversary@MIPS, DIMACS May 23, 2022

NOKIA Bell Labs

- Collaborations with Dan Bienstock
- A simplified multicommodity flow model
- Solution outline
- Observations
- Numerical confirmation
- Epilogue

Network Design Tools -- Our (early 1990s) Little Cottage Industry at Bellcore



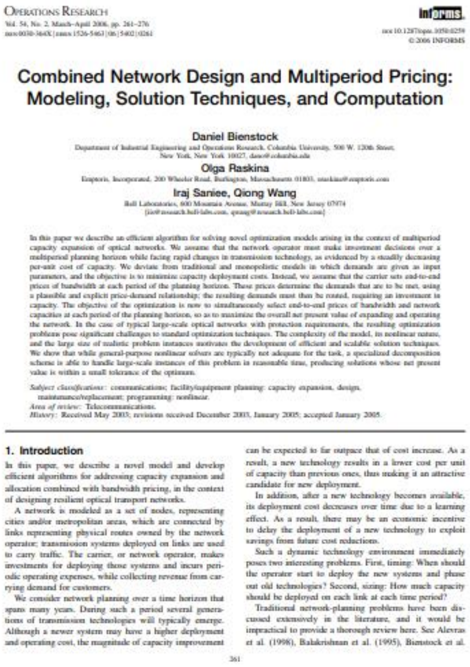
Data: Square nodes have large fixed cost (switches), Hexagonal nodes have medium fixed cost, Circular nodes and links have per unit cost. There is partial capacity.

Problem: Given end-to-end new demands, determine min-cost expansion to accommodate all new demands.

Type: MILP

Technologies: SONET, Frame Relay, ATM, IP over optics.

Novel Extension -- Beyond Typical Regulated Monopoly-Inspired Models


OPERATIONS RESEARCH

Combined Network Design and Multiperiod Pricing: Modeling, Solution Techniques, and Computation

Daniel Bienstock

Olga Raskina

Iraj Saniee, Qiong Wang

1. Introduction

Data: Demand-price elasticity and previous fixed costs

Problem: maximize profit (revenue – cost) over multi-year planning horizon

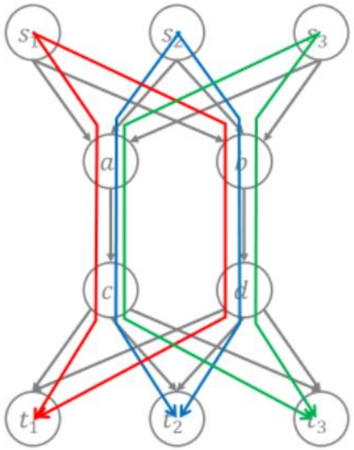
Type: Novel (very) non-linear objective function

Context: Inspired by companies like Level3 who were putting tons of dark fibers to own telecom infrastructure globally (early 2000s)

Multicommodity Flow Problem (myMCF) -- Can one solve it 'analytically'?*

- Include fixed costs? Too complex
- Path-based solution? Maybe later
- Cost minimization? Start with capacity-guided:

Given an arbitrary graph $\boldsymbol{G}$ with weighted symmetric adjacency matrix $\boldsymbol{A} = \boldsymbol{A}^{\boldsymbol{T}} = (a_{ij})$ and traffic matrix $\boldsymbol{T} = (t_{kl})$, the required flow from k to l, route of t_{kl} at k and each intermediate node j is proportional to a_{ij} until flow reaches k.



Trivial bookkeeping exercise?

____

*Joint work with O. Narayan, UCSC.

Extension to non-uniform edge weights and arbitrary demand

$\boldsymbol{G} = (\boldsymbol{N}, \boldsymbol{L})$is aconnected graph
$\boldsymbol{A} =$ an NxN symmetric matrix of non$-$negative weighted adjacencies

$d_j = \sum_j a_{ij}$ is weighted degree of node j
$\boldsymbol{D} =$ an NxN diaginal matrix of weighted adjacencies
$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$ is the graph Laplacian

$\boldsymbol{\Pi} = (N^{-1})1_{NxN}$ perturbation matrix with N^{-1} in all entries
$\boldsymbol{M} = \boldsymbol{L} + \boldsymbol{\Pi}$ is the perturbed graph Laplacian

$\boldsymbol{T} = (t_{kl})$ is an NxN symmetric non-negative flow matrix
$T_k = \sum_k t_{kl}$ the total flow out of node j
$\widetilde{\boldsymbol{T}} = (T_k)$ is an NxN diagonal matrix of total nodal out flows
$\boldsymbol{L_T} = \widetilde{\boldsymbol{T}} - \boldsymbol{T}$ for symmetric T

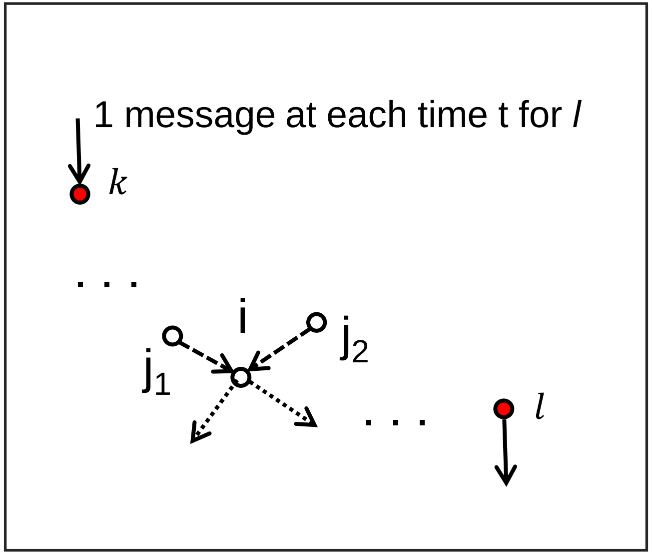
Load on each node in myMCF for a specific commodity k-l

- Start **with 1 unit of flow for a given source-destination pair** $(\boldsymbol{k}, \boldsymbol{l})$
- Let f_i^{kl} be the flow at node i. Then

$$f_i^{kl} = \delta_{ik} - \delta_{il} + \sum_{j \sim i} \frac{a_{ji}}{d_j} f_j^{kl}$$

with the boundary condition $f_l^{kl} = 0$*. In vector form

$$\boldsymbol{f}^{kl} = \boldsymbol{v}^k - \boldsymbol{v}^l + \boldsymbol{A}\boldsymbol{D}^{-1}\boldsymbol{f}^{kl} \qquad (*)$$



____

*We'll add back the total flow terminating at each l in (*****).

Derivation of random load, 1

$\boldsymbol{f}^{kl} = \boldsymbol{v}^k - \boldsymbol{v}^l + \boldsymbol{A}\boldsymbol{D}^{-1}\boldsymbol{f}^{kl}$ with $f_l^{kl} = 0$ $(*)$

Using the substitution $r_j^{kl} = f_j^{kl}/d_j$, or $\boldsymbol{r}^{kl} = \boldsymbol{D}^{-1}\boldsymbol{f}^{kl}$ (*) becomes

$\boldsymbol{D}\,\boldsymbol{r}^{kl} = \boldsymbol{v}^k - \boldsymbol{v}^l + \boldsymbol{A}\,\boldsymbol{r}^{kl}$

Or

$\boldsymbol{L}\,\boldsymbol{r}^{kl} = \boldsymbol{v}^k - \boldsymbol{v}^l$ (**)

If $\boldsymbol{L}$ were invertible we could just write $\boldsymbol{r}^{kl} = \boldsymbol{L}^{-1}(\boldsymbol{v}^k - \boldsymbol{v}^l)$ and we'd be nearly there. Need some adjustment. Notice that (**) is already telling us we have an under-determined system of equations. We add a homogeneous boundary condition $r_l^{kl} = 0$ which we'll update at the end.

Aside – Uniform Perturbation of the Laplacian

Let $\{0 = \lambda_0 \leq \lambda_1 \leq \ldots \leq \lambda_{N-1} \leq 2d_{\max}$ be the eigenvalues and $\{\boldsymbol{\xi}^{\alpha}; \alpha = 0, 1, \ldots, N-1\}$ be the corresponding orthonormal eigenvectors of $\boldsymbol{L}$. Recall corresponding to $0 = \lambda_0$, have

$$(\boldsymbol{\xi}^0)^T = (1,1, \ldots, 1)^T/\sqrt{N}$$

Consider projection $\boldsymbol{\Pi} = \left(\frac{1}{N}\right)_{NxN}$ and $\boldsymbol{M} = \boldsymbol{L} + \boldsymbol{\Pi}$. Notice that $\boldsymbol{M}$ has the same eigen system as $\boldsymbol{L}$ except that λ_0(M)= 1.

$$\boldsymbol{M}\,\boldsymbol{\xi}^0 = (\boldsymbol{L} + \boldsymbol{\Pi})\boldsymbol{\xi}^0 = 0 + \sum_i \xi_i^0\, \mathbf{1}_N = N\left(\frac{1}{N}\right)\left(\frac{1}{\sqrt{N}}\right)\mathbf{1}_N = \boldsymbol{\xi}^0$$

and

$$\boldsymbol{M}\,\boldsymbol{\xi}^{\alpha} = (\boldsymbol{L} + \boldsymbol{\Pi})\boldsymbol{\xi}^{\alpha} = \boldsymbol{L}\boldsymbol{\xi}^{\alpha} + \boldsymbol{\Pi}\,\boldsymbol{\xi}^{\alpha} = \lambda_{\alpha}\boldsymbol{\xi}^{\alpha} \text{ with } \alpha > 0.$$

Above means $\boldsymbol{M}$ has a simple tractable inverse in terms of λs and $\boldsymbol{\xi}$s. Also observe that

$$\boldsymbol{M} = \boldsymbol{L} + \boldsymbol{\Pi} \rightarrow \boldsymbol{M}\boldsymbol{\Pi} = \boldsymbol{L}\boldsymbol{\Pi} + \boldsymbol{\Pi}^2 = \boldsymbol{\Pi}^2 = \boldsymbol{\Pi} \rightarrow \boldsymbol{M}^{-1}\boldsymbol{\Pi} = \boldsymbol{\Pi} \qquad (***)$$

Derivation of random load, 2

Let us return to $\boldsymbol{L}\,\boldsymbol{r}^{kl} = \boldsymbol{v}^k - \boldsymbol{v}^l$ with $r_l^{kl} = 0$ (**).

Adding $\boldsymbol{\Pi}\,\boldsymbol{r}^{kl}$ to both sides of (**), we get

$$\boldsymbol{L}\,\boldsymbol{r}^{kl} + \boldsymbol{\Pi}\,\boldsymbol{r}^{kl} = (\boldsymbol{v}^k - \boldsymbol{v}^l) + \boldsymbol{\Pi}\,\boldsymbol{r}^{kl}$$

$$\boldsymbol{M} r^{kl} = (\boldsymbol{v}^k - \boldsymbol{v}^l) + \boldsymbol{\Pi}\,\boldsymbol{r}^{kl}$$

So at last

$$\begin{aligned}\boldsymbol{r}^{kl} &= \boldsymbol{M}^{-\mathbf{1}}[(\boldsymbol{v}^k - \boldsymbol{v}^l) + \boldsymbol{\Pi}\,\boldsymbol{r}^{kl}] = \boldsymbol{M}^{-\mathbf{1}}(\boldsymbol{v}^k - \boldsymbol{v}^l) + \mathbf{M}^{-\mathbf{1}}\boldsymbol{\Pi}\,\boldsymbol{r}^{kl} \\ &= \boldsymbol{M}^{-\mathbf{1}}(\boldsymbol{v}^k - \boldsymbol{v}^l) + \boldsymbol{\Pi}\,\boldsymbol{r}^{kl} \quad \text{(by (***))} \\ &= \boldsymbol{M}^{-\mathbf{1}}(\boldsymbol{v}^k - \boldsymbol{v}^l) + \theta(kl)\mathbf{1}_N\end{aligned}$$

And we can get $\theta(kl)$ from the boundary condition.

Derivation of random load, 3

$$0 = r_l^{kl} = [\boldsymbol{M}^{-\mathbf{1}}(\boldsymbol{v}^k - \boldsymbol{v}^l]_l \; + \theta(kl)$$
$$0 = [\boldsymbol{M}^{-\mathbf{1}}\boldsymbol{v}^k]_{\boldsymbol{l}} - [\boldsymbol{M}^{-\mathbf{1}}\boldsymbol{v}^l]_{\boldsymbol{l}} \; + \theta(kl)$$
$$0 = [\boldsymbol{M}^{-\mathbf{1}}]_{\boldsymbol{lk}} - [\boldsymbol{M}^{-\mathbf{1}}]_{\boldsymbol{ll}} + \theta(kl)$$

$$\theta(kl) = [\boldsymbol{M}^{-\mathbf{1}}]_{\boldsymbol{ll}} - [\boldsymbol{M}^{-\mathbf{1}}]_{\boldsymbol{lk}}$$

Now going back to r_j^{kl}

$$r_j^{kl} = [\boldsymbol{M}^{-1}]_{jk} - [\boldsymbol{M}^{-1}]_{jl} - [\boldsymbol{M}^{-1}]_{lk} + [\boldsymbol{M}^{-1}]_{ll} \qquad (****)$$

We can now get f_j^{kl} for t_{kl} units from (****)

Derivation of random load, 4 – Final Result

So flow f_j^{kl} for t_{kl} units of flow is just $t_{kl} r_j^{kl} d_j$.

If Λ_j denotes the total flow at node j for *all* commodities, then

$$\Lambda_j = \sum_k \sum_l t_{kl} \; f_j^{kl} + \sum_k t_{kj}$$
$$= d_j \sum_k \sum_l t_{kl} \left(r_j^{kl} = [\boldsymbol{M}^{-1}]_{jk} - [\boldsymbol{M}^{-1}]_{jl} - \; [\boldsymbol{M}^{-1}]_{lk} + [\boldsymbol{M}^{-1}]_{ll} \right) + T_j$$

And at last, if the demand matrix is symmetric (t_{kl})=(t_{lk})

$$\Lambda_j = d_j Tr[\boldsymbol{L}_T \; \boldsymbol{M}^{-1}] + T_j$$

$$\boldsymbol{\Lambda} = \boldsymbol{D} \; Tr[\boldsymbol{L}_T \; \boldsymbol{M}^{-1}] + \widetilde{\boldsymbol{T}}. \boldsymbol{1}_N \qquad (*****)$$

Derivation of random load, 5 – Case of All-to-All Uniform Flows

$If\ t_{kl} = 1\ for\ all\ k \neq l\ then$

$$\Lambda_j = d_j N \sum_{a>0} \frac{1}{\lambda_a} + (N-1)$$

This is seen more easily from re-writing (*****) in this form

$$\Lambda_j = d_j \sum_k \sum_l t_{kl} \sum_{\alpha>0} \frac{1}{\lambda_\alpha} \xi_l^\alpha (\xi_l^\alpha - \xi_k^\alpha) + T_j$$

Load due to Net MCF

In derivation of

$$\Lambda_j = (N-1) + (N\sum_{a\geq 1}\frac{1}{\lambda_a})\, d_j$$

we allowed RW to go through a transit node multiple times, each time contributing to load. We may define an alternative load, $\overline{\Lambda}_j$, which measures the $net\ flow$ through node. Thus for neighbors i & j net inflow to node i from neighbor j

$$= \frac{p_j^{kl}}{d_j} - \frac{p_i^{kl}}{d_i}$$

Thus in equilibrium

$$\overline{\Lambda}_j^{kl} = \frac{1}{2}\sum_{i\sim j}|r_i^{kl} - r_j^{kl}| + \frac{1}{2}(\delta_{ik} + \delta_{il})$$

Adding up for all (k, l)

$$\overline{\Lambda}_j = \frac{1}{2N(N-1)}\sum_{k\neq l}\sum_{i\sim j}|r_i^{kl} - r_j^{kl}| + \frac{1}{N}$$

Load due to Net MCF

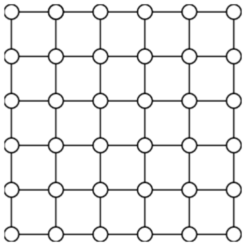
After substituting for r_j^{kl} from (3),

$$\overline{\Lambda}_j = \frac{1}{2N(N-1)} \sum_{k \neq l} \sum_{i \sim j} \sum_{a>0} |(\xi_k^a - \xi_l^a) \frac{1}{\lambda_a} (\xi_j^a - \xi_i^a)| + \frac{1}{N}$$

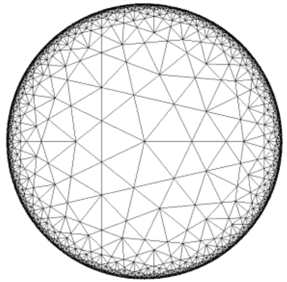
where as before $\{\lambda_a, \xi^a\} \in eigSys(\boldsymbol{L})$.

Maximum Load Λ_{d} for Some Prototypical Graphs Using Shortest Paths

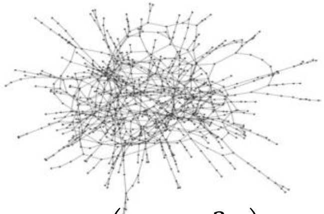
Square Lattice $\Lambda_{\mathrm{d}} \sim O(N^{\frac{3}{2}})\ for\ large\ N$



Hyperbolci grids $\Lambda_{\mathrm{d}} \sim N^2\ for\ large\ \ N$



For random graph, $\Lambda_{\mathrm{d}} \sim\ O\big(N\ log(N)\big)\ for\ large\ N$



$\mathrm{G}_{\mathrm{E-R}}\left(1000, \frac{2}{1000}\right)$

Dan’s software can express sentiments – Example of C code

```
struct module
       {
        char gross[3];
        int manures[3], stools[6], sh*ts[9];
        float zap, dunk, ditch;
       }
```

NOKIA

