Network-Design Sensitivity Analysis

Paul Tune and Matthew Roughan
matthew.roughan@adelaide.edu.au

School of Mathematical Sciences
University of Adelaide

July 2, 2014

The Problem

Traffic Matrices

- simply a matrix of traffic from $A \rightarrow B$
- fundamental input for most network planning (invariant)

But good network data are notoriously hard to get, and inaccurate

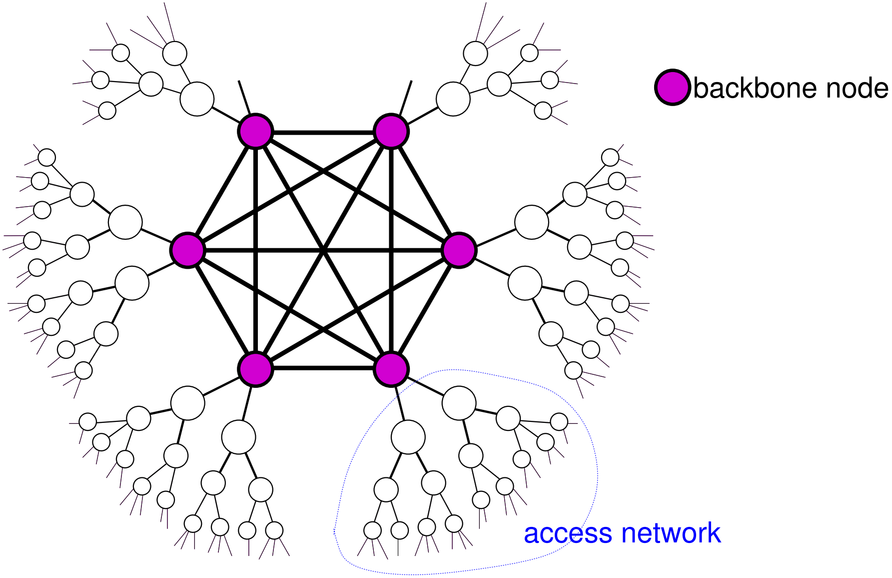
- measurements are an afterthought
  - often you don't get what you would like
  - measurements aren't calibrated
  - missing data is a big issue
- big data
  - sampling, sketching, ...
- prediction
  - planning needs predictions, which have errors
  - what about green fields planning?

Existing Network Planning Solutions

- Ignore the issue, and make a guess
- Make a guess, and then add 50%
- Oblivious routing
  - routing scheme that works for any traffic matrix
- Valiant network design
  - network design that works for any traffic matrix

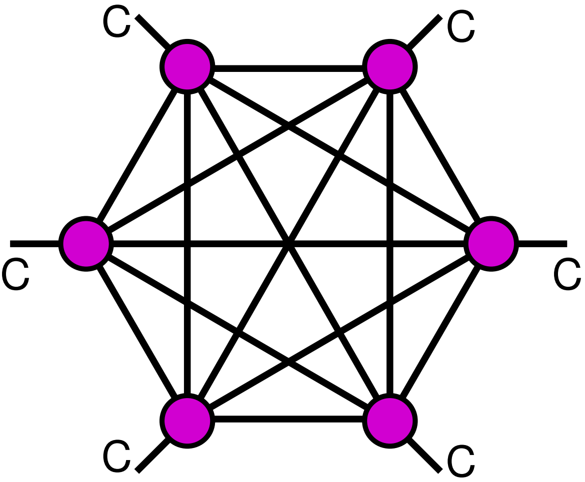
Are any of these used?

Valiant network design [Val82, ZSM04, ZSM05]



Valiant network design

Abstract the access network to have capacity C



Valiant network design

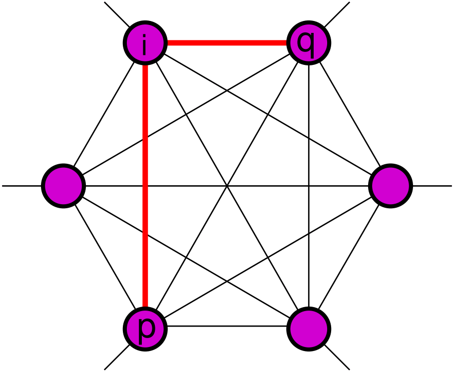
Simple case (which can be generalized)

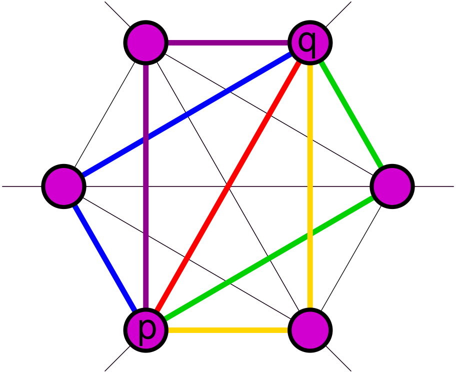
- don't know the traffic matrix t_{pq}
- assume access capacity C to each backbone node
- this limits traffic matrix

$$\sum_q t_{pq} \leq C \text{ and } \sum_p t_{pq} \leq C$$

- route traffic demand t_{pq} as follows
  - divide it into $|N|$ even groups
  - route group i as follows $p \to i \to q$
  - load balance across all of the possible 2 hop routes
  - do the same for all $p, q \in N$

Valiant network design





Valiant network design

- compare to direct routing
  - each packet traverses 2 hops
  - $2\times$ the bandwidth needed over optimal (a star)
- but it is *oblivious* to the traffic matrix
  - this design is provably the best oblivious network design [ZSM05] (given a certain cost model).
- it also has great advantages for survivability
  - can survive any combination of node failures
  - highly robust to link failures as well
  - only need marginal increases in link capacities

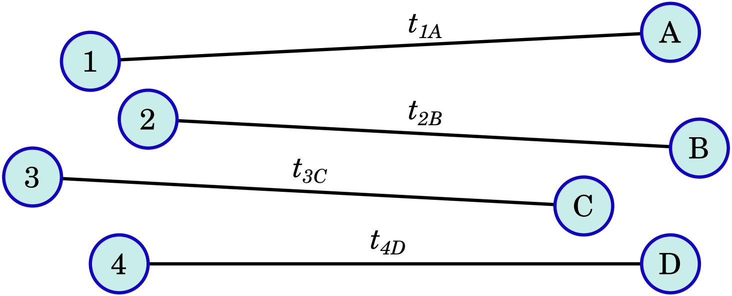
Better yet

- both of the above approaches assume we know don't know the traffic matrix
  - they are **oblivious**
  - but that has a cost in terms of efficiency
- but in reality we know something
  - e.g. SNMP measurements of traffic on links
  - e.g. partial netflow across network
- **can we design a network using the information we have, but taking into account the information we are missing?**
  - obviously we can, but how?

Mean-Risk Analysis in Finance

- Reduce volatility (and hence risk) of a portfolio by including multiple "uncorrelated" stocks
- Overall risk is reduced by balanced portfolio
  - no such thing as a free lunch
  - lowers returns if we knew the future
  - but in absence of predictions, we are overall better off

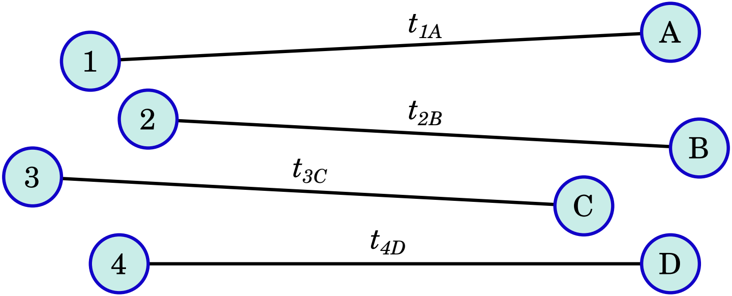
Imagine we need to carry the traffic $t_{i,j}$



- optimal capacities

$$c_{i,j} = t_{i,j}$$

Now assume we don't know the $t_{i,j}$ exactly



- assign capacities

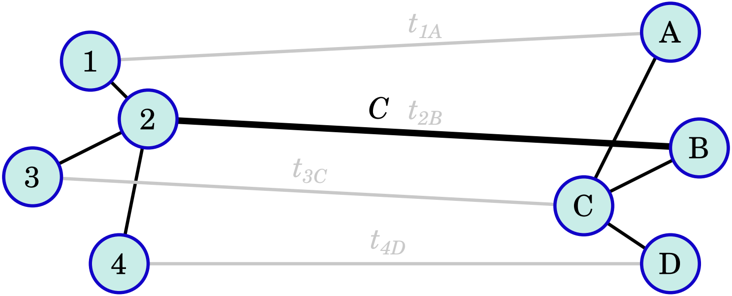
$$c_{i,j} = \hat{t}_{i,j} + \gamma \sigma_{i,j}$$

  - $\hat{t}_{i,j}$ is predicted traffic
  - $\sigma_{i,j}$ is some estimate of possible errors
  - γ is a over-build factor

Issues

- We can often get predictions $\hat{t}_{i,j}$
- Estimating errors $\sigma_{i,j}$ in predictions is harder
- Choosing γ is hard
  - it balances risk against efficiency
  - it's hard to choose because the balance is poor here

So lets build it more like this



- it's "less optimal" in one sense
  - we have to build more links
  - but shorter links are usually cheaper
- the one long link multiplexes the traffic from left to right
  - capacity on long link

$$C = \sum_{i,j} \hat{t}_{i,j} + \gamma\sigma$$

Assuming independent errors

$$\sigma \neq \sum_{i,j} \sigma_{i,j}$$

To determine σ we need an error model

- very typically, people use IID Gaussian

$$\sigma^2 = \sum_{i,j} \sigma_{i,j}^2$$

  e.g. if $\sigma_{i,j}$ above were 2

$$\sigma = \sqrt{4+4+4+4} = 4 = \frac{1}{2} \sum_{i,j} \sigma_{i,j}$$

  So when errors are large enough

$$C = \sum_{i,j} \hat{t}_{i,j} + \gamma\sigma < \sum_{i,j} c_{i,j}$$

Multiplexing gain

- The phenomena is called multiplexing gain
  - its been well known for a long time
  - but it doesn't seem to be used (explicitly) in IP network design?
- The analogy with finance is clear
  - a portfolio decreases risk by including shares whose risks are (hopefully) uncorrelated, so total risk is less
  - multiplexing does the same
- There is a cost for lack of knowledge
  - it doesn't have to be too bad
  - but you don't want to ignore data

Networks

Networks are more complicated than the above example but same deal applies

- optimal when traffic is known isn't robust (its sensitive)
- optimizing separately is a bad idea
- so some aggregation should happen

Tricky bits

The goal to balance risk with optimality

- What is risk here
  - is IID Gaussian a good model for errors?
  - how do we measure risk?
- What is optimal
  - lots of work on network design, so we will use a simple case
- How do you balance them?
  - stochastic optimization
  - but still need a hook?
  - we will do it using an ensemble of synthetic traffic matrices

Traffic Matrix Synthesis 101

- Simplest idea is IID, but Gaussian doesn't work
  - Log-normal [NST05]
    - reasonable match to observed distribution
    - doesn't have any structure
- Gravity model [Rou05], e.g.
  - generate "populations" p_i
  - traffic $t_{i,j}$

$$t_{i,j} \propto p_i p_j$$

  - matches some structure, and distribution
  - certainly isn't perfect
- Not a lot of other research on the topic
  - and we want to do something slightly different anyway
  - we don't want a completely random ensemble

Our goal

Generate an ensemble of TMs "like" a predicted matrix

- admissible
  - satisfies constraints
    - non-negative
    - imposed by network
- centered
  - their average centers on the predicted matrix
- controlled
  - variance around the predicted matrix can be controlled
  - linear parameter β
  - similar to the role of σ in Gaussian case

Typical methods

Form ensemble by adding noise $z_{i,j}$

- usually IID
- often Gaussian

$$\begin{aligned} \textbf{Additive}: &\quad y_{i,j} = t_{i,j} + \sigma z_{i,j}, \\ \textbf{Multiplicative}: &\quad y_{i,j} = t_{i,j}\left(1 + \sigma z_{i,j}\right), \end{aligned}$$

Both have problems:

- IID loses any structure
- Allows negative values
  - can truncate, but this introduces 0s, and de-centers
- Scaling
  - multiplexing means estimates of large elements should be relatively more accurate
  - neither of these have the correct scaling

Constraints

Start with admissibility: describe by constraints

- We use four sets of constraints
    1. *Non-negativity*: $t_{i,j} \geq 0$, $\forall i, j = 1, 2, \cdots, N$,
    2. *Row sums*: $\sum_j t_{i,j} = r_i$, $\forall i$,
    3. *Column sums*: $\sum_i t_{i,j} = c_j$, $\forall j$, and
    4. *Total traffic*: $\sum_{i,j} t_{i,j} = \sum_i r_i = \sum_j c_j = T$,
- Chosen to be exemplars
    - Easier to measure/predict total in/out traffic at a PoP
    - Matched to previous work on inference
- Could have any convex constraints

Spherically Additive Noise Model (SANM)

Let's enforce fundamental constraints by design

- Note that for non-negative traffic we can write it

$$t_{i,j} = a_{i,j}^2$$

- And total traffic constraints says

$$\sum_{i,j} a_{i,j}^2 = T,$$

- So traffic matrix sits on a N^2 dimensional hyper-sphere

Spherically Additive Noise Model (SANM)

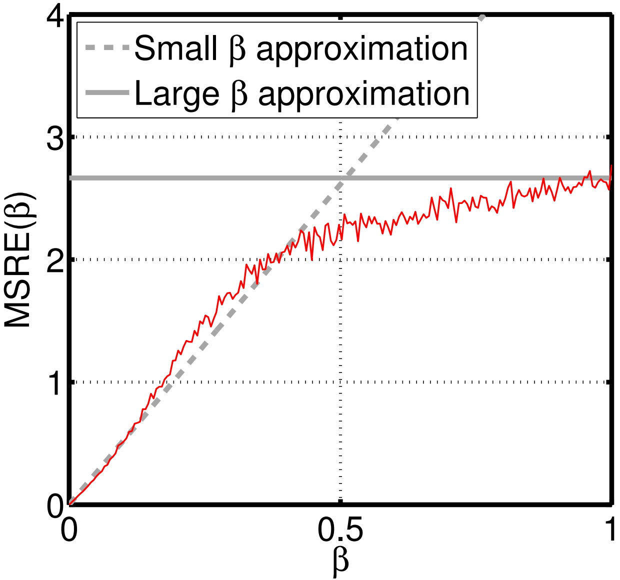
So, add noise in the N^2 dimensional space, along the hypersphere

- form new matrix

$$y_{i,j} = (a_{i,j} + \beta z_{i,j})^2$$

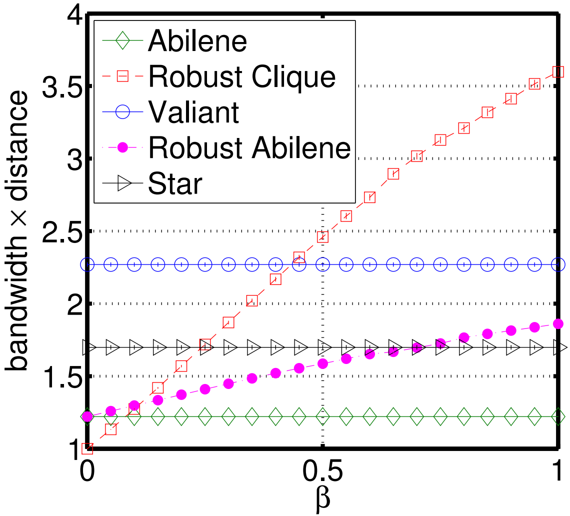
- then scale back to hypersphere, i.e., like normalizing
  - we are adding noise for a point on the hyper-sphere (hence the name)
- use Iterative Proportional Fitting (IPF)
  - finds "closest" TM on the hyper-sphere that fits the constraints

Synthesis Analysis



Network Design

We have looked at a few cases, but lets just take one here: redesign of Abilene



Conclusion

- Good design needs robustness to errors in predictions
  - extreme case is oblivious, but this is wasteful
  - using a little bit of information can improve things
- Mechanism to do so is to be able to generate synthetic traffic matrices
  - Spherically Additive Noise Model
  - nice properties
  - seems to work in practice

Further reading I

Antonia Nucci, Ashwin Sridharan, and Nina Taft, *The problem of synthetically generating IP traffic matrices: Initial recommendations*, ACM SIGCOMM Computer Communication Review **35** (2005), no. 3.

Matthew Roughan, *Simplifying the synthesis of Internet traffic matrices*, ACM SIGCOMM Computer Communications Review **35** (2005), no. 5, 93–96.

Leslie G. Valiant, *A scheme for fast parallel communication*, SIAM Journal on Computing **11** (1982), no. 2, 350–361.

Rui Zhang-Shen and Nick McKeown, *Designing a predictable Internet backbone*, HotNets III (San Diego, CA), November 2004, `http://tiny-tera.stanford.edu/~nickm/papers/index.html`.

Further reading II

———, *Designing a predictable Internet backbone with Valiant load-balancing*, Thirteenth International Workshop on Quality of Service (IWQoS) (Passau, Germany), June 2005, `http://tiny-tera.stanford.edu/~nickm/papers/index.html`.