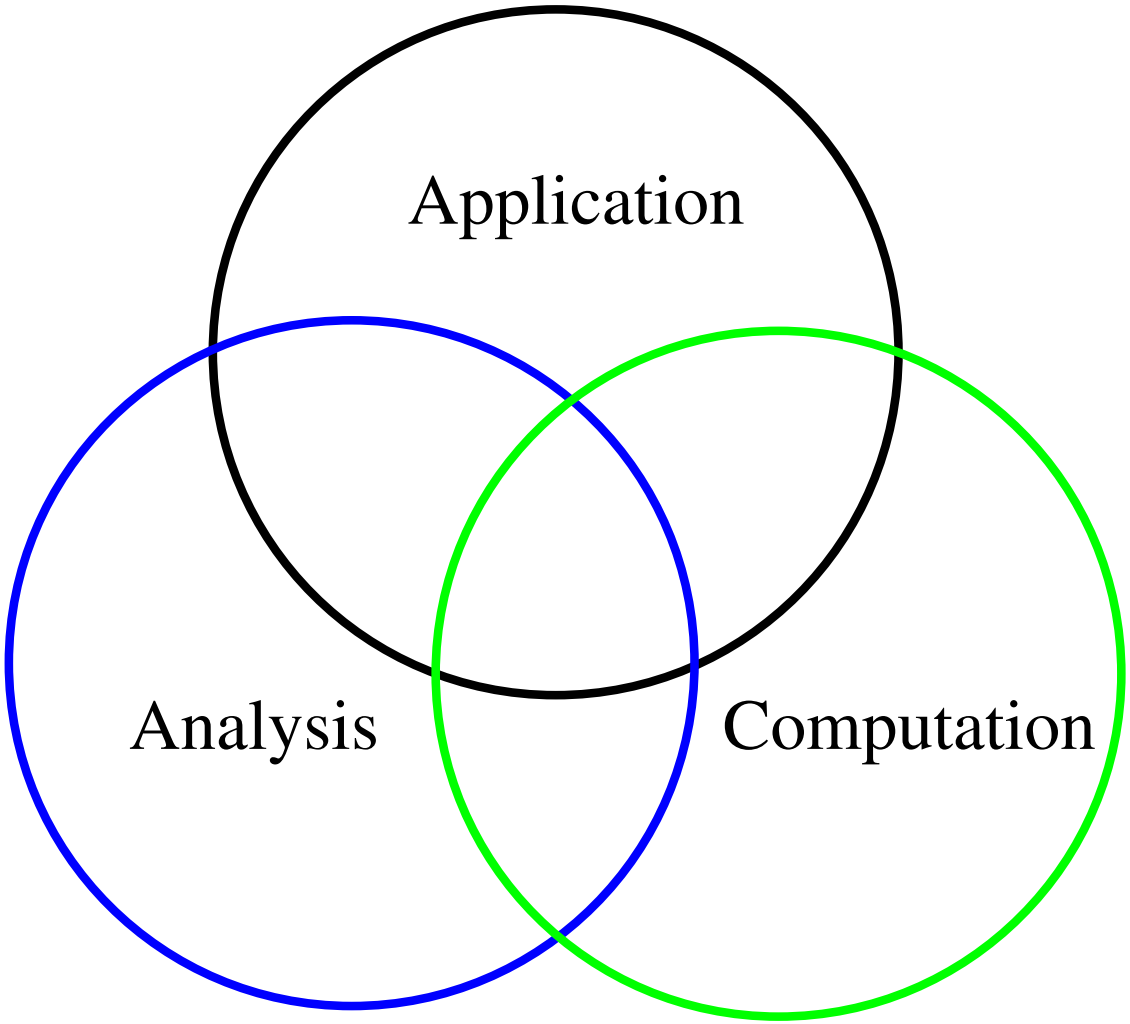
From Network Tomography to Power Networks?

David Bindel

Department of Computer Science
Cornell University

15 May 2012

The Computational Science & Engineering Picture



My interests

- Numerical methods
  - Structured and nonlinear eigenvalue solvers
  - Continuation for eigenvalue problems
  - Model reduction
  - Fast solvers for structured systems
  - Finite element methods
  - Network models
- Applications
- Software and computational infrastructure

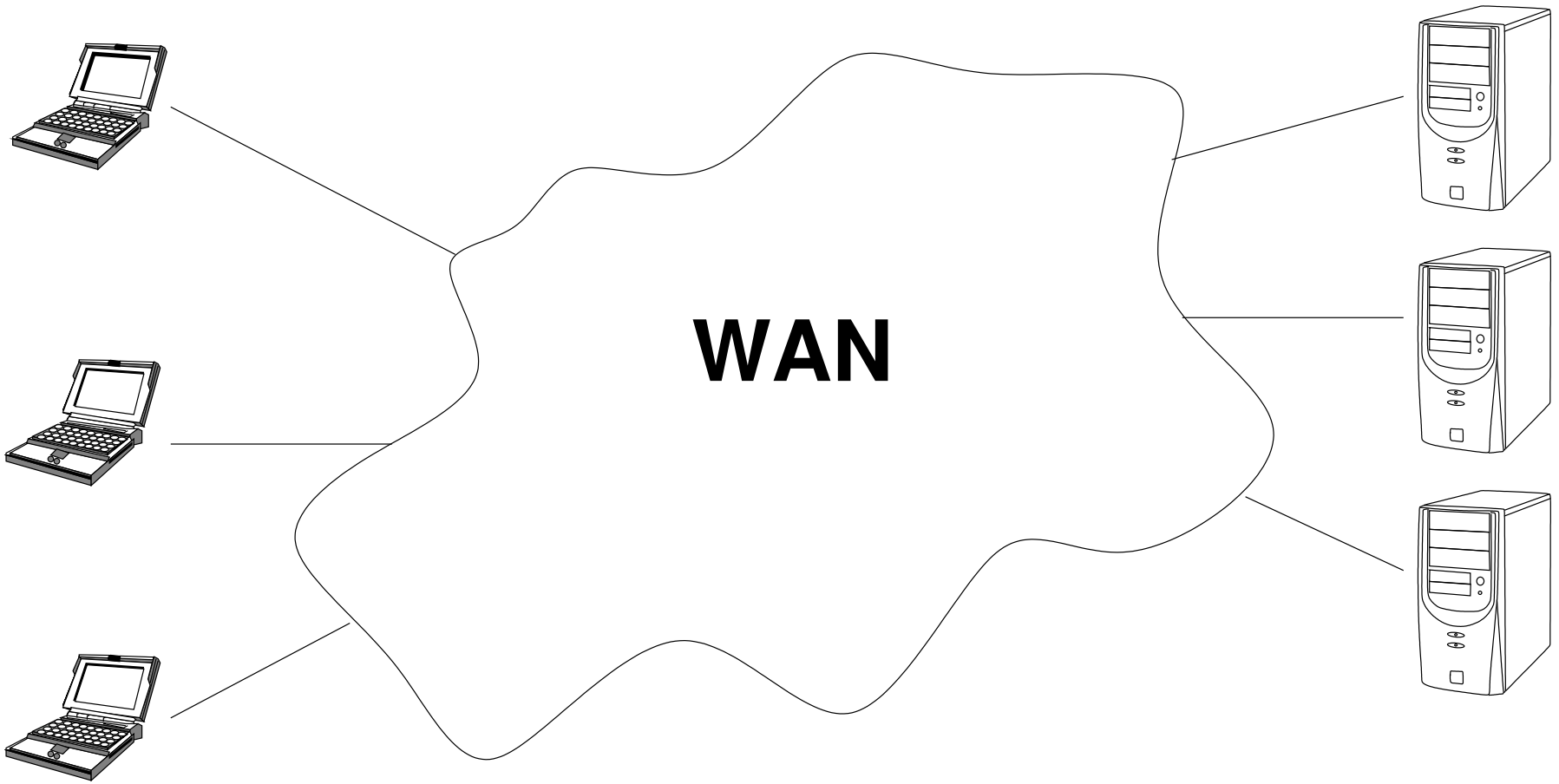
My interests

- Numerical methods
- Applications
  - MEMS simulation (system level and device level)
  - Computer network tomography
  - Social networks: community detection, opinion formation
  - Materials design
- Software and computational infrastructure

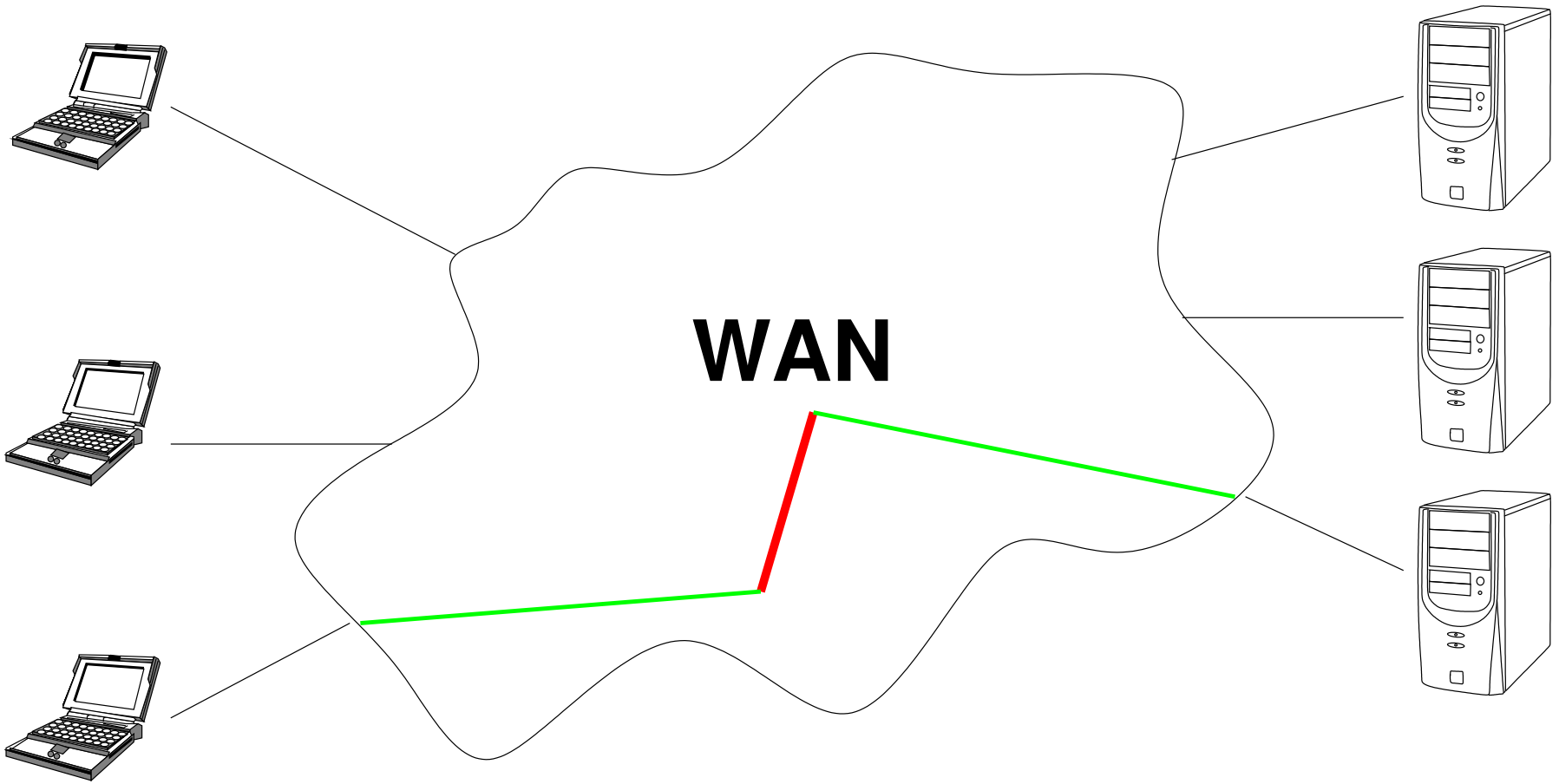
My interests

- Numerical methods
- Applications
- Software and computational infrastructure
  - Packages for individual algorithms
  - Simulators (SUGAR, HiQLab, MATFEAP, MatScat, deal2lab)
  - Infrastructure for cloud numerics (with Gehrke)

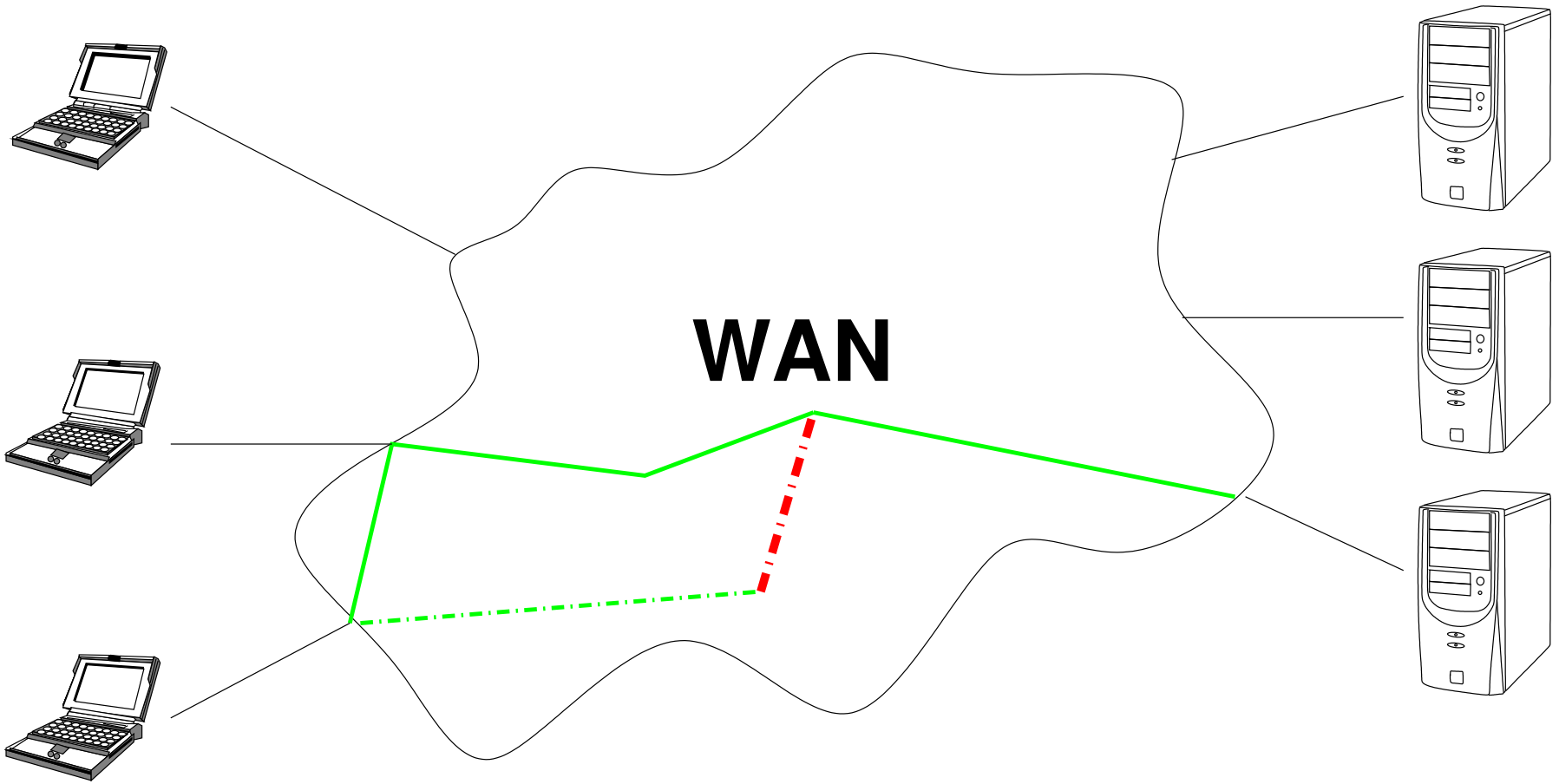
Application: Computer Network Tomography



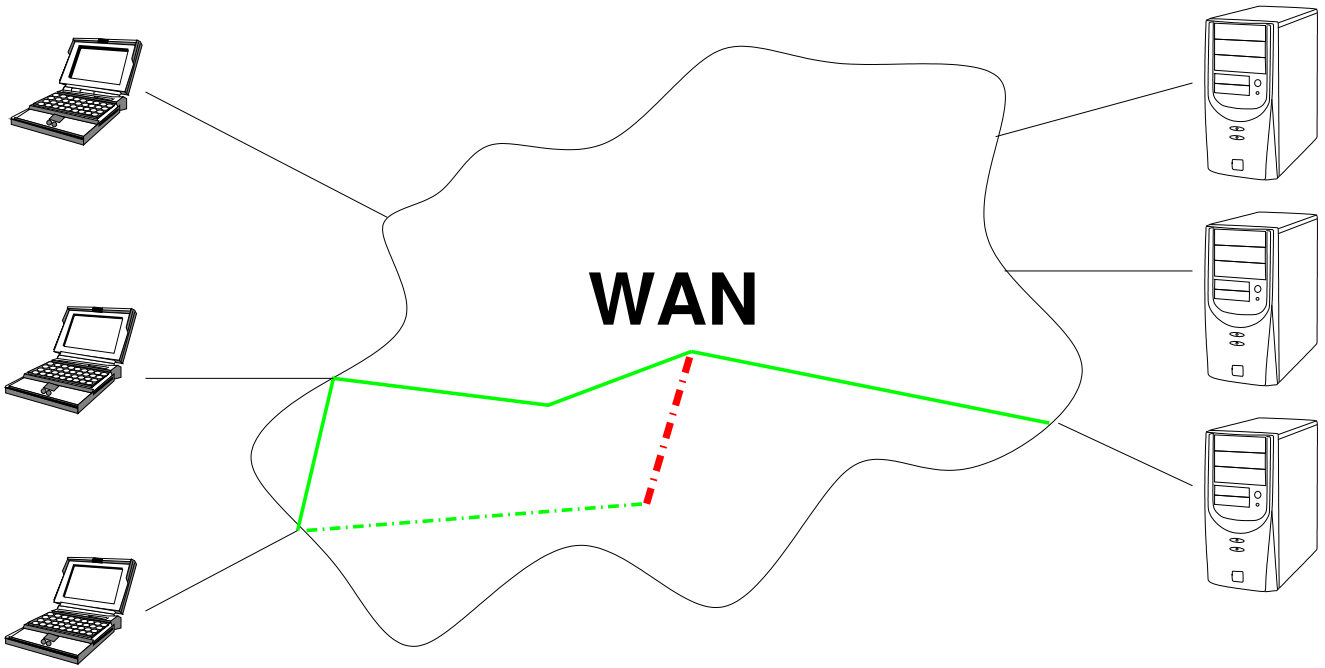
A Possible Problem



Find and Fix or Route Around?



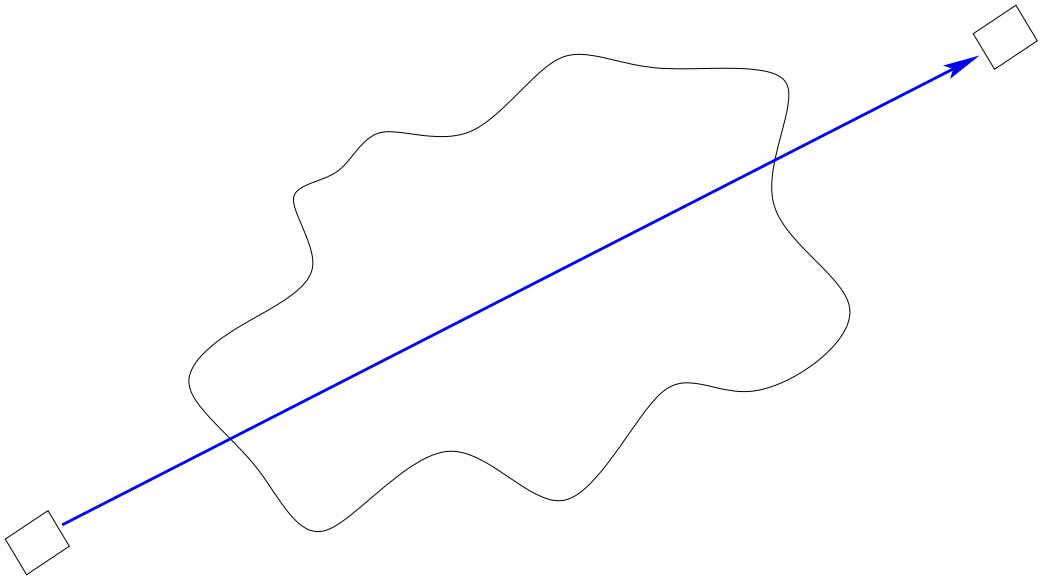
Overlays and measurement



Measure a few paths to infer:

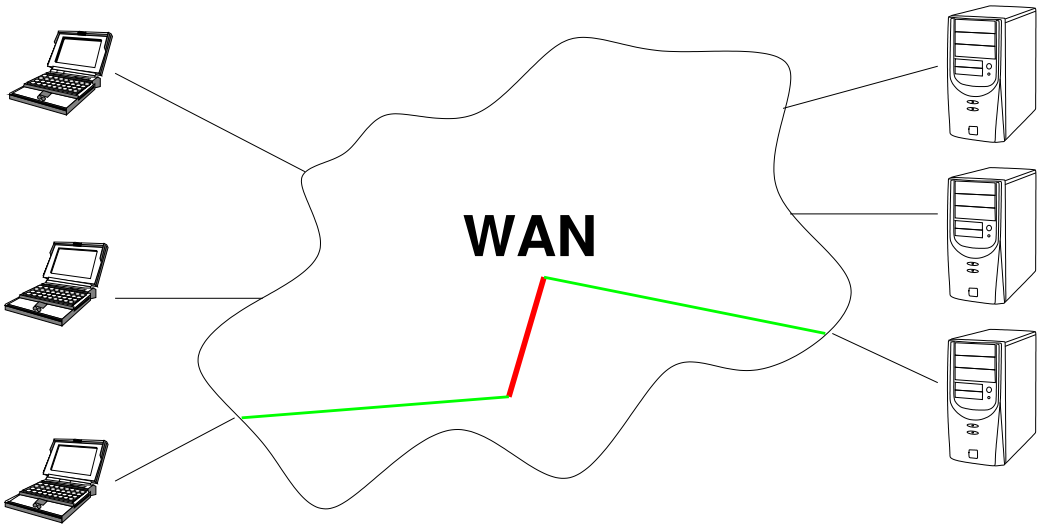
- *Path* properties (Chen, B., Song, Chavez, Katz: 2003, 2004, 2007)
- *Link* properties (Zhao, Chen, B.: 2006, 2009)
- Routing topology? (underway)

Discrete Radon transform



Radon transform:

$$(Ru)(L) = \int_L u(\mathbf{x})\,|d\mathbf{x}|$$



Discrete version:

$$(Gu)_i = \sum_{j \in \text{links}(i)} u_j$$

Additive metrics and path matrices

For additive metrics ($\log(P(\text{transmission}))$, latency, jitter, ...):

$$Gu = b$$

where

- b_i = property of ith end-to-end path
- u_j = property of link j
- $G_{ij} = \begin{cases} 1 & \text{if path } i \text{ uses link } j \\ 0 & \text{otherwise} \end{cases}$

Goal: Relate network structure to a structured decomposition of G

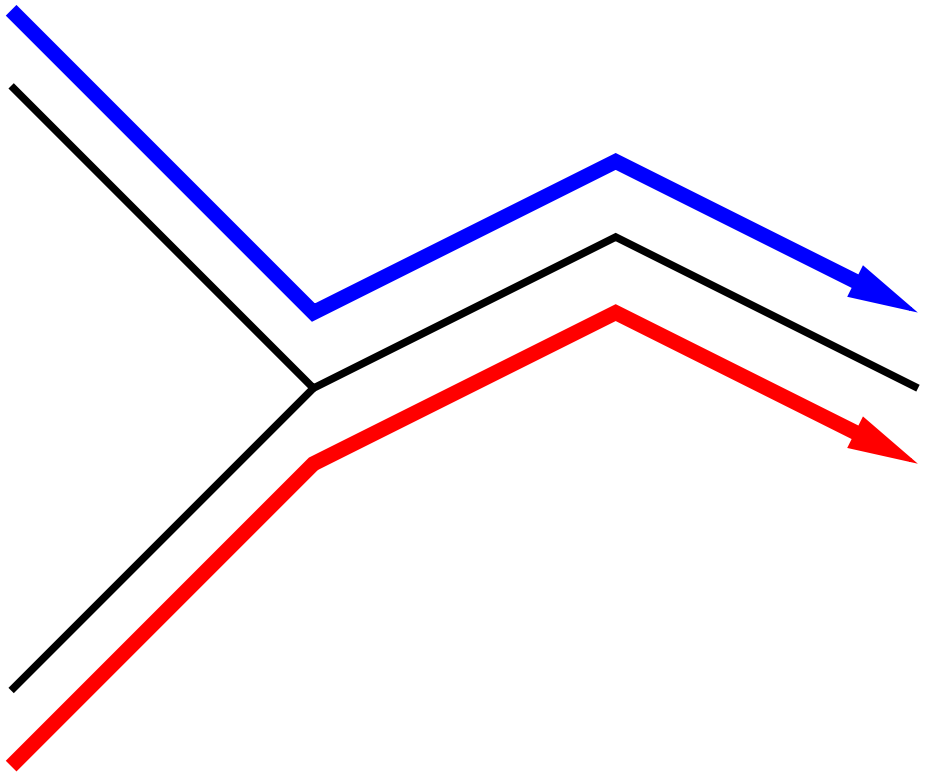
Properties of G

Network	Matrix G
Routing path	Row of G
Short paths	Sparse G
Routing table updates	Low rank updates to G

$k = \text{rank}(G) <$ # links $\ll$ # paths (for n sufficiently large).

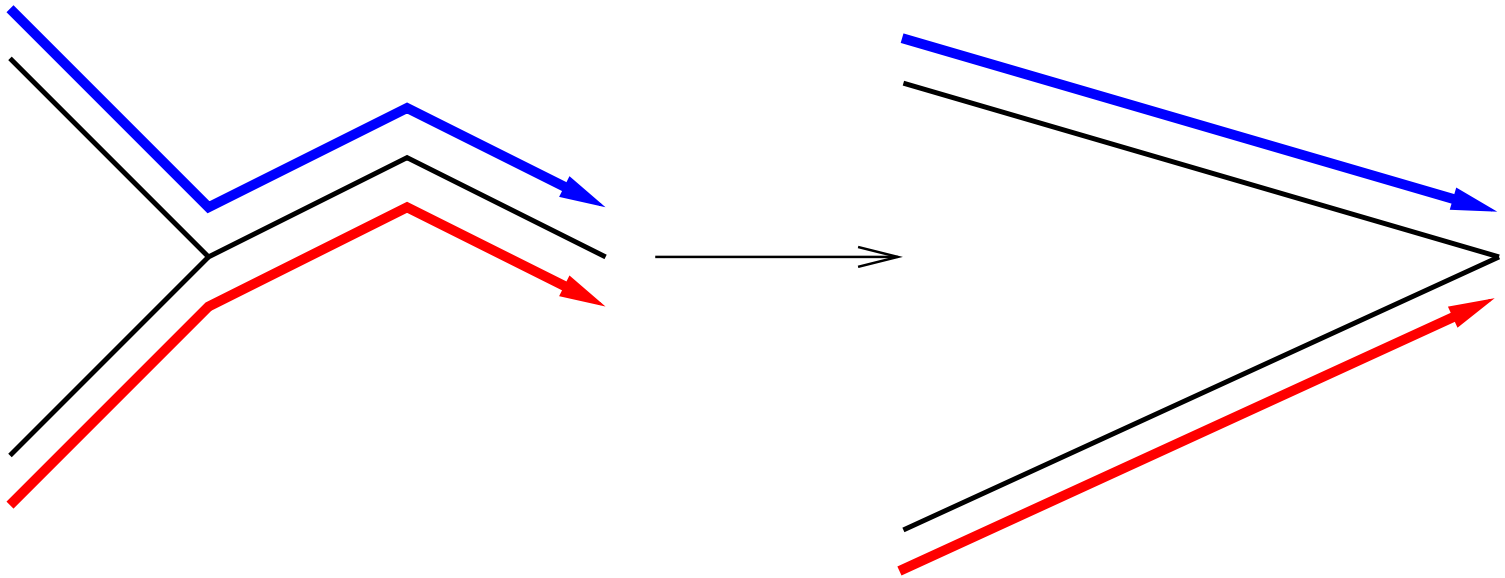
Ex: 500 nodes (of 100K), G is 249500×33237, $k = 9643$.

Identifiability



If both paths are flaky, what link is to blame?

Network virtualization and matrix factorization



Factor out a zero-one "virtualization matrix":

$$G(:, \text{fan}) = \begin{bmatrix} c_1 & c_2 & c_1 + c_2 & c_1 + c_2 \end{bmatrix} = \begin{bmatrix} c_1 & c_2 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix}$$

Even virtual links may be "unidentifiable."

Network virtualization and matrix factorization

Write network virtualization as

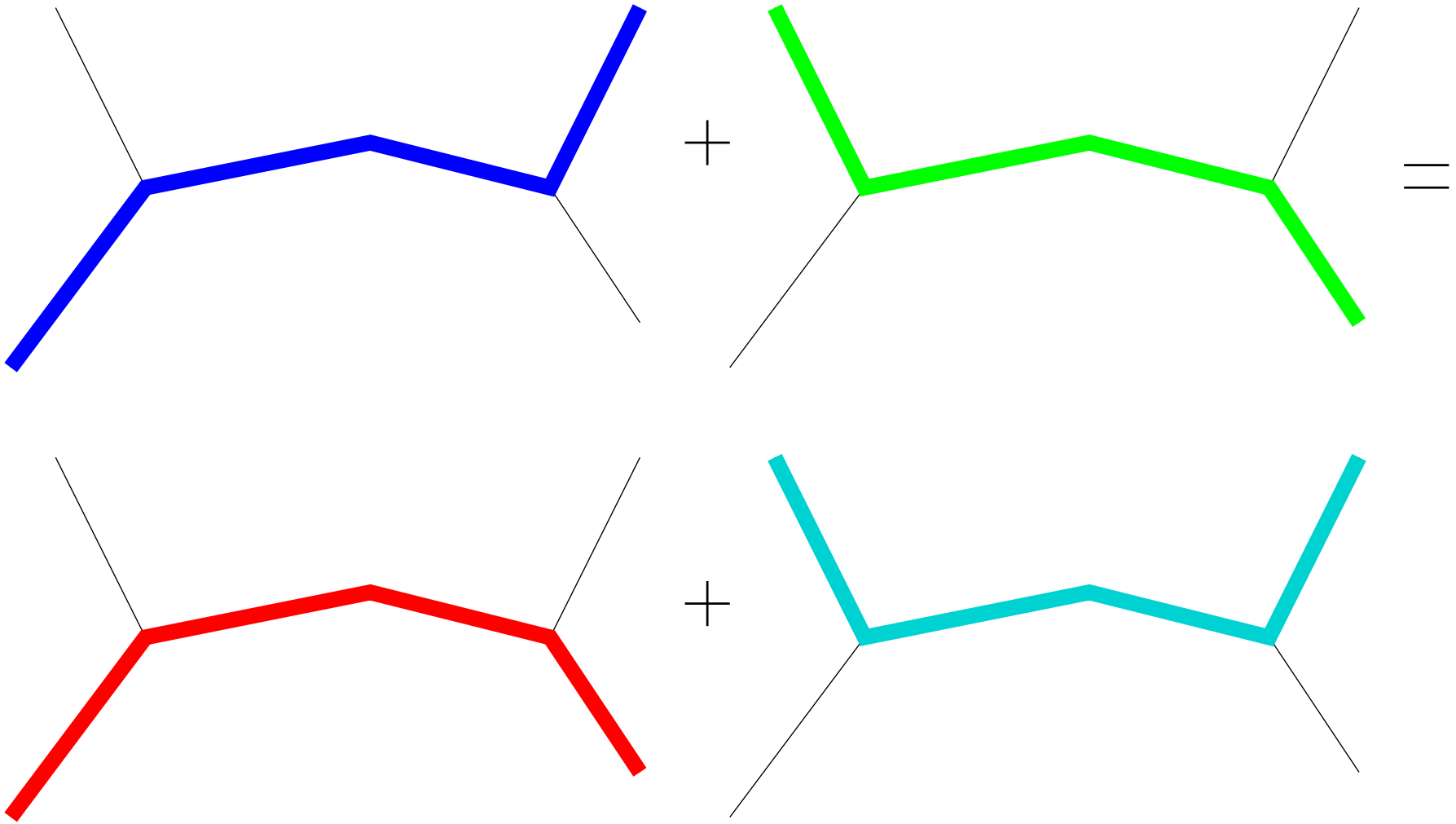
$$G = G^v S$$

where

- $G_{ij} = \begin{cases} 1 & \text{if path } i \text{ uses link } j \\ 0 & \text{otherwise} \end{cases}$
- $G^v_{ik} = \begin{cases} 1 & \text{if path } i \text{ uses virtual link } k \\ 0 & \text{otherwise} \end{cases}$
- $S_{kj} = \begin{cases} 1 & \text{if virtual link } k \text{ includes link } i \\ 0 & \text{otherwise} \end{cases}$

This handles (some) column dependencies. What about rows?

Linear Algebra of Paths



Matrix factorization perspective

Can combine with topology virtualization:

$$PG = \begin{bmatrix} I \\ T \end{bmatrix} \bar{G}^v S$$

where P is a permutation and S is a zero-one virtualization matrix.

Topics for offline discussion:

- How does one compute this factorization fast?
- How can you keep it up to date?
- How can you use it to infer
  - Path and link properties?
  - Missing topology?

Other connections?

- Domain decomposition and subdomain model reduction
  - What should an aggregator send to be most useful for SE?
- Stability monitoring and novel eigenvalue problems
  - Electrical network + out-of-band control = DDAE stability?
- Latency-tolerant infrastructure
  - How to program bulk-synchronous methods on cloud (efficiently)?
  - Right tradeoffs between bulk-synchronous and asynchronous?