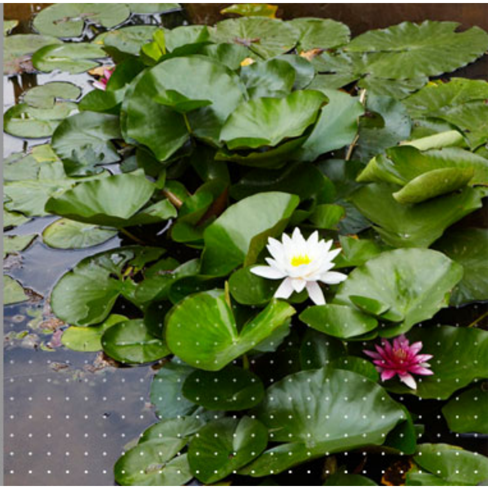
Minimum-Cost Network Design with (Dis)economies of Scale



Spyridon Antonakopoulos

Joint work with **Matthew Andrews** and **Lisa Zhang**

Athens Colloquium on Algorithms and Complexity, August 27th 2010

Overview

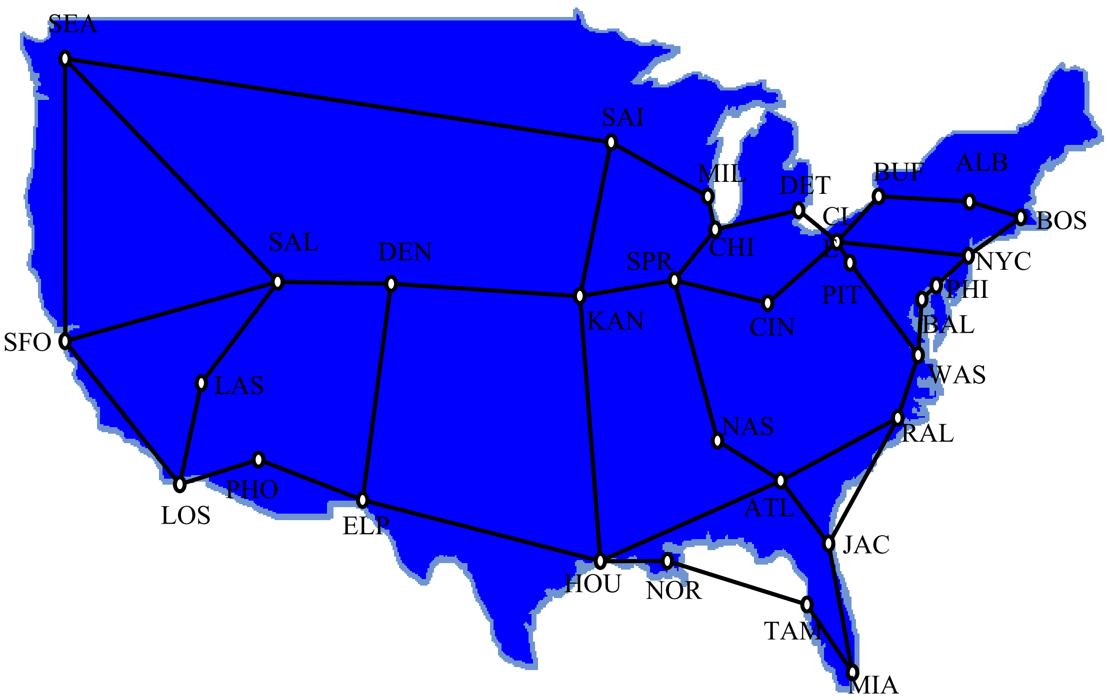
1. Background
   - Motivation
   - Relation to buy-at-bulk network design
2. Achieving a polylogarithmic approximation
   - Cost discretization
   - Well-cut-linked flow decomposition
   - Construction of expander virtual topology
   - Edge-disjoint routing
3. Concluding remarks

1 Background

Network design

Network: carries people and/or commodities (oil, data, etc.) between a number of locations.

Example: optical-core communications network covering the United States.



traffic demands

source	destination	bandwidth
NYC	PIT	10 Gbps
BOS	SEA	5 Gbps
ATL	SFO	8 Gbps
CHI	DEN	7 Gbps
...		

Alcatel·Lucent

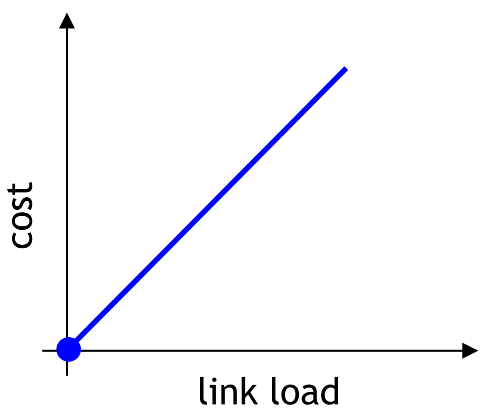
Network design (contd.)

Typical formulation of minimum-cost network design problem:

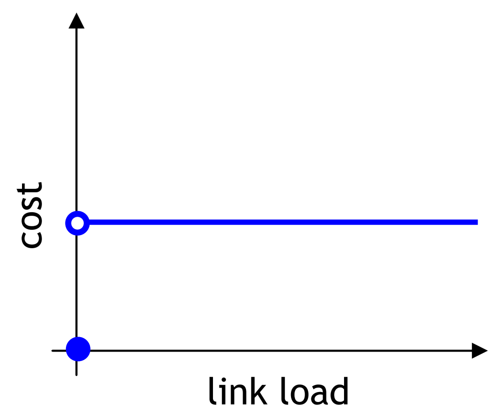
- Available network topology
  - Tree, ring, general graph, …
- Set of end-to-end traffic demands
  - Single sink, all pairs, multi-commodity, …
- A function representing the cost of a network element in relation to the traffic carried by that element
  - Uniform vs. non-uniform
- Problem-specific requirements, if any
  - Latency, fault tolerance, “hard” capacity constraints, …
- **Goal:** determine a minimum-cost network that can serve all traffic demands (and respect any additional requirements)

Some simple cost functions

linear



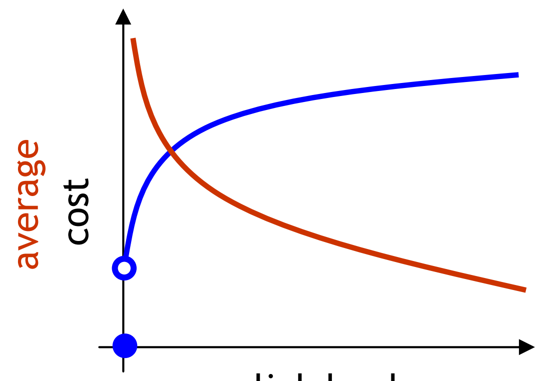
step-function ("flat")



- Linear cost function $\Rightarrow$ decomposes to a shortest-paths problem for each traffic demand
  - Easy – solvable in polynomial time [Dijkstra]
- Step-function ("flat cost") $\Rightarrow$ Steiner forest problem
  - Somewhat easy – we can find a solution with at most twice the optimal cost in polynomial time (i.e. approximation ratio 2) [Agrawal-Klein-Ravi, Goemans-Williamson]

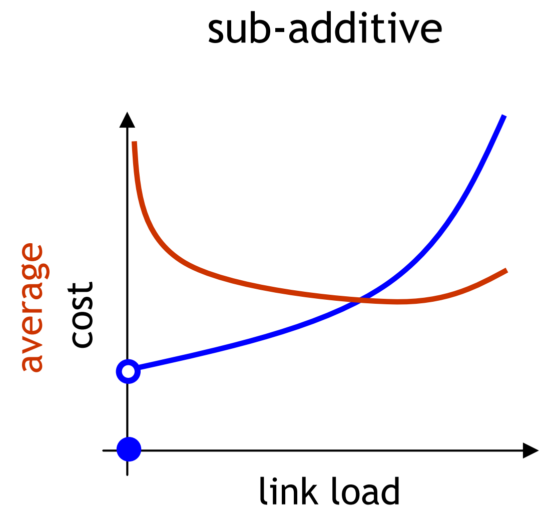
Buy-at-bulk cost functions

sub-additive



- Sub-additive cost function (continuous or discontinuous) $\Rightarrow$ buy-at-bulk network design problem
  - **Sub-additivity:** $f(x_1 + x_2) \leq f(x_1) + f(x_2)$
  - Models **economies of scale**
  - Manageable – we can find a solution with at most $O(\log n)$ times or polylog(n) times the optimal cost in polynomial time (uniform/non-uniform version resp.) [Awerbuch-Azar, Chekuri-Hajiaghayi-Kortsarz-Salavatipour]

Energy costs and (dis)economies of scale

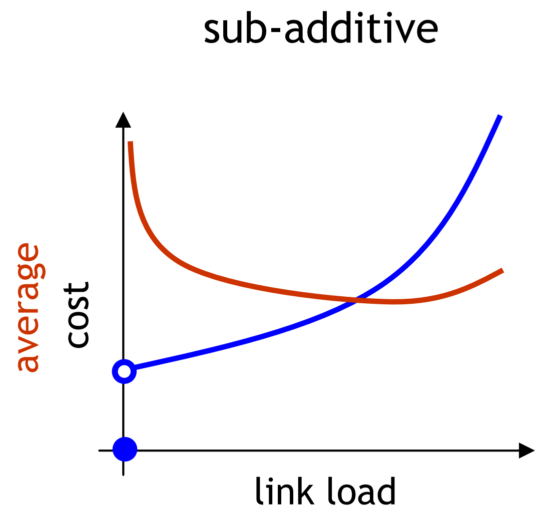


What if the cost function has the form $f(x) = \sigma + x^{\alpha}$ for x > 0, f(0) = 0?

- **Motivation:** describes power consumption of CMOS circuits with speed scaling
- Reflects a combination of economies and **diseconomies of scale**
  - Similarly-shaped cost curves commonly encountered in many industries $\Rightarrow$ potential for wide model applicability

Alcatel·Lucent

Energy costs and (dis)economies of scale (contd.)



- **Bad news:** a lower bound on the approximation ratio is exponentially dependent on α [A, Andrews-Fernandez-Zhang-Zhao]
  - **Good(?) news:** in case of CMOS power, $\alpha \leq 3$ and thus may be considered a constant
- **More good news:** for the uniform version, we can find a solution with at most polylog(n) times the optimal cost in polynomial time

α determines degree of polynomial

2 Achieving a polylogarithmic approximation

Algorithm outline

- Partition traffic demands by bandwidth, in buckets [1, 2), [2, 4), [4, 8), …
- Discretize cost function
- For each bucket, while there exist unrouted demands:
  - Solve LP relaxation and decompose fractional solution into well-cut-linked flows
  - Construct an expander graph as virtual network topology
  - Route (at least) some of the demands via edge-disjoint paths on the virtual topology
- Output the union of all partial routings

We need to ensure that each partial routing **(i)** serves at least a polylogarithmic fraction of the demands, and **(ii)** has cost at most polylog(n) times the optimum.

- Recall the analysis of the greedy set-cover algorithm…
  - However, a low ratio of cost / (#demands served) for every partial solution does not suffice in our case. We must bound the overall number of partial solutions as well.

Bucketing demands

- Place demands with bandwidth $[2^{j-1}, 2^j)$ in bucket j.
- Round up the bandwidth of all demands in bucket j to 2^j.
  - Lose only a factor of 2 in the approximation.
- Henceforth, we deal with the demands in one bucket at a time.
  - All such demands have the same bandwidth (convenient).
  - W.l.o.g. we also assume that each demand has distinct endpoints (terminals) from other demands in the same bucket.
- For a bucket j such that $2^j \geq \sigma^{1/\alpha}$, replace the cost function with $f^*(x) = 2x^\alpha$ at a loss of another factor 2 in the approximation.
  - Then, apply a CP-rounding algorithm from [Andrews-Fernandez-Zhang-Zhao] to route the demands in that bucket. ✓
- For a bucket j such that $2^j < \sigma^{1/\alpha}$, aggregate demands.
  - For simplicity, let's forget about aggregation; assume $\sigma^{1/\alpha} = 1$ and unit demands…

Discretizing the cost function

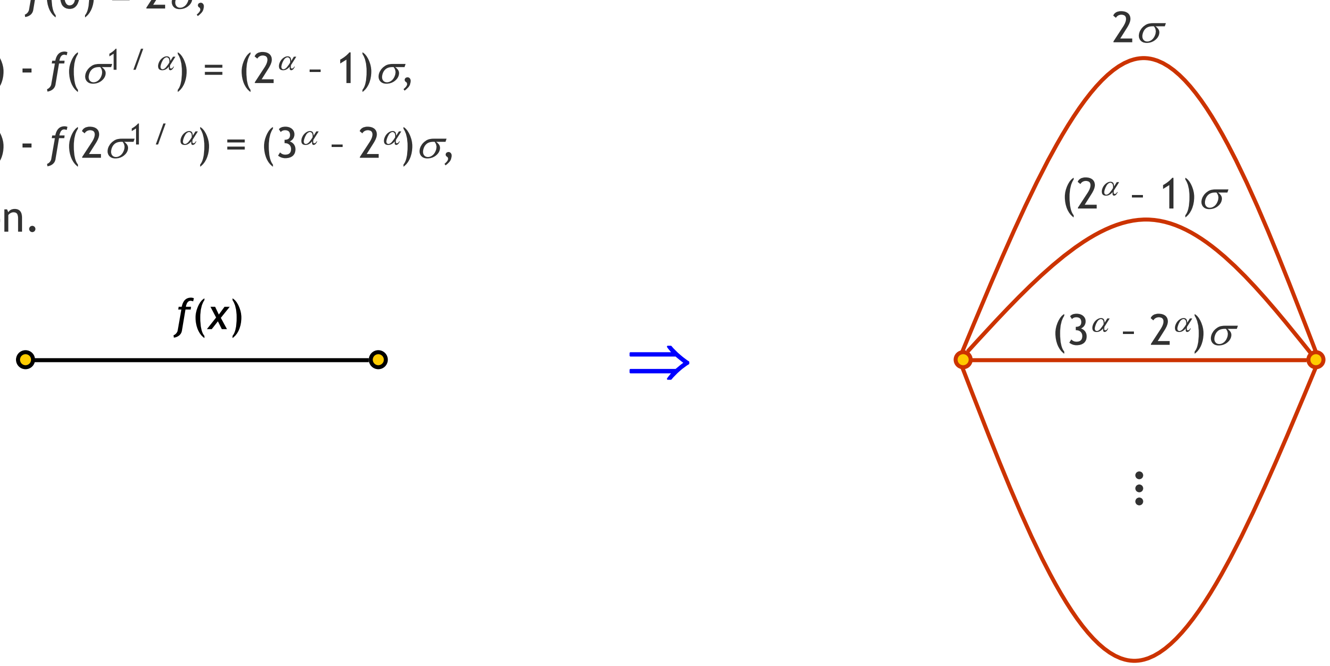
- Replace each link by a collection of parallel edges, with a fixed **capacity** equal to $\sigma^{1/\alpha}$, and flat (step-function) costs:

$f(\sigma^{1/\alpha}) - f(0) = 2\sigma,$

$f(2\sigma^{1/\alpha}) - f(\sigma^{1/\alpha}) = (2^\alpha - 1)\sigma,$

$f(3\sigma^{1/\alpha}) - f(2\sigma^{1/\alpha}) = (3^\alpha - 2^\alpha)\sigma,$

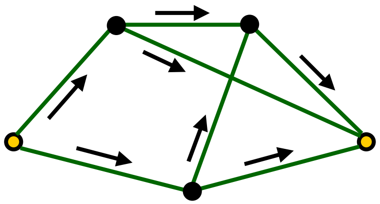
and so on.



- Clearly, cheap edges will be used before expensive ones.
- **Note:** this transformation does **not** make the problem equivalent to Steiner forest, because of edge capacities.

LP relaxation and well-cut-linked flows

- We formulate and solve an LP relaxation of the problem instance (including only the demands in the current bucket).
- In the (fractional) solution, each demand may be routed along more than one paths, each carrying only part of the demand's bandwidth.
  - These paths constitute a **flow** associated with the demand.
  - Not acceptable as a solution to our problem.



- Decompose into well-cut-linked terminals [Chekuri-Khanna-Shepherd]
  - Create node-disjoint subgraphs of original graph.
    - Discard demands with terminals in different subgraphs.
    - At least a certain fraction of demands survives.
  - **Salient property:** In order to cut one such subgraph in two parts, we would have to remove edges carrying a substantial total amount of flow (in the above fractional solution), proportional to the smaller total remaining demand in either part.

Constructing an expander topology

- When is a graph $G' = (V', E')$ an expander?
  - Regular
  - Expansion: for any $S \subset V'$ with $|S| \subset |V'| / 2$, the number of edges in the cut $(S, V' - S)$ is at least $c|S|$, for a constant $c > 0$.
- Very similar the property of well-cut-linked terminals we saw earlier.
  - But how to use it for expander construction?
  - Suppose we are given a routine that for any balanced partition (A, B) of a node set V^* produces a perfect matching. Then, we can construct an expander by calling this routine $O(\log^2 |V^*|)$ times. [Khandekar-Rao-Vazirani]
- We build an expander on the terminals of each subgraph of the decomposition.
  - Here, perfect matchings consist of entire **paths** joining terminals, not just edges.
  - Well-cut-linkedness ensures the existence of such a path-matching.
  - **Result:** a virtual expander topology that uses each edge of the real topology at most a polylogarithmic number of times

crucial for bounding routing cost

Edge-disjoint routing in the virtual topology

- Given an expander and a set of node pairs (with each node belonging to at most one pair), we can route at least a polylogarithmic fraction of those pairs via edge-disjoint paths. [Rao-Zhou]
  - Expander graphs tend to have many short paths…
- Thus, in the real topology we can route at least a polylogarithmic fraction of the demands, while the load on every edge is at most polylogarithmically larger compared to the solution of the LP relaxation.
- We apply the same process on the remaining demands. No more than a polylogarithmic number of iterations required.

Putting it all together:

- **Theorem.** Uniform network design with (dis)economies of scale is polylog(n)-approximable.

3 Concluding remarks

Concluding remarks

- Applicability extends to more general cost functions (not necessarily polynomial) as long as they increase at least at a linear rate – but not too quickly, of course.
  - Asymptotically linear concave functions are also covered. In general, though, concave cost functions are better handled in the buy-at-bulk framework.

Open questions

- What is the (in)approximability of the non-uniform version?
- The intermediate capacitated problem may be viewed as a special case of Fixed Charge Network Flow, which has been well-studied from a heuristics perspective. Can our algorithm be adapted for the latter problem too?

Thank you!