DISCRETE OPTIMIZATION AND NETWORK DESIGN

Daniel Bienstock
Columbia University

Generic definition

Expand or **add capacity** to an existing network, or **build** a new network; subject to certain constraints

And in the resulting network, **route** a given set of multicommodity demands, subject to certain constraints

So as to maximize a given performance measure of the routing, or to deliver a required performance from the routing, at minimum cost.

Mixed-integer programming formulations

Expand or **add capacity** to an existing network, or **build** a new network; subject to certain constraints –

Use integer variables, sometimes 0/1. Constraints: e.g. topological. Example: the network must be a union of rings with ≤ 10 nodes each.

In the resulting network, **route** a given set of multicommodity demands, subject to certain constraints –

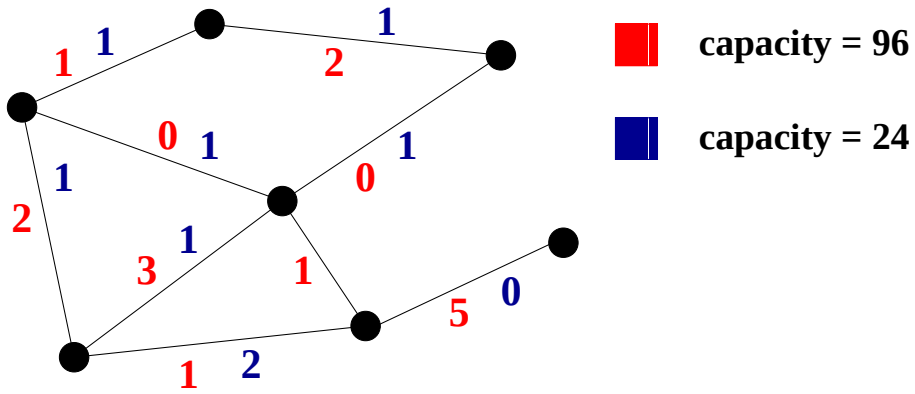
Usually modeled using continuous variables. The variables are used to model flows on arcs of the network, or path flows.

Optimize

Objective: minimize cost, or maximize throughput. Cost of the installed capacity, throughput = amount of traffic that is routed.

Many submodels

Different types of transmission systems. (Magnanti, Balakrishnan, Vachani; many others) e.g. on a link we may install an integer number of units of bandwidth 12, and an integer number of units of bandwidth 24, or 48, etc.



For each arc of the network we have a constraint of the form

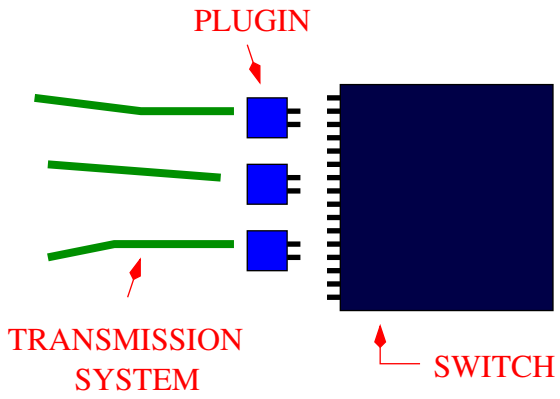
$$f_{ij} - \sum_t M^t x_{ij}^t \leq U_{ij}$$

where f_{ij} = total traffic on link (i, j), x_{ij}^t is the number of units of type t installed on the link; U_{ij} is the existing capacity

Related to knapsack polyhedra (Pochet and Wolsey, Günlük and Pochet)

Modeling node capacities

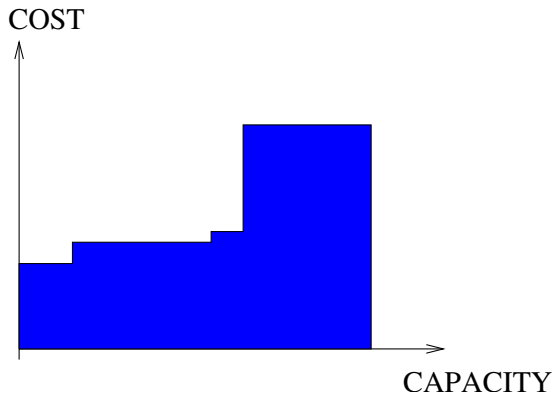
Example: ATM networks (Asynchronous Transfer Mode), B. and Saniee



- At each of a selected set of nodes, must decide whether to place a switch or not
- In addition to placing a switch, we must choose a set of adapters, or "plugins"
- Each plugin is traffic specific (e.g. video, packets, etc)
- Each plugin has a certain throughput and a number of "pins"
- The switch has a total number of pins available, plus capacity

The leased line cost model.

(Grötschel and Stoer, Alevras, others)



→ The cost of procuring capacity on a given arc is a step function – the function depends on the arc

For a given arc (i, j) we have a submodel of the form

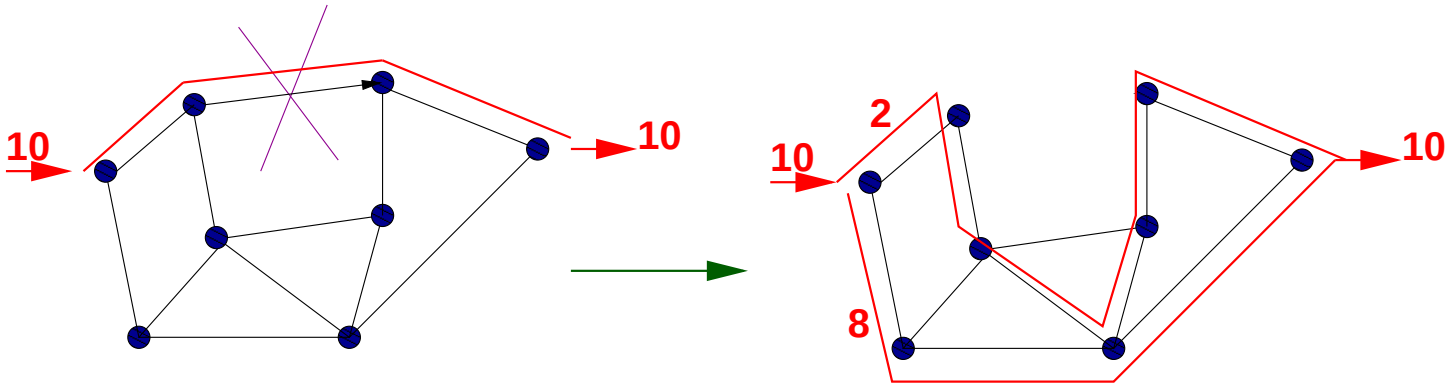
$$f_{ij} - \sum_t M_{ij}^t x_{ij}^t \leq U_{ij}$$

$$x_{ij}^t \leq x_{ij}^{t-1}$$

where $\boldsymbol{f_{ij}}$ = total traffic on link $\boldsymbol{(i,j)}$, $\boldsymbol{x_{ij}^t = 1}$ if we use "rung" $\boldsymbol{t}$; $\boldsymbol{M_{ij}^t}$ = incremental capacity of rung $\boldsymbol{t}$ over rung $\boldsymbol{t-1}$

Survivability requirements

If an edge (node) is removed, the remaining network should adapt seamlessly and carry all, or a prescribed fraction, of the demands



Many possible models: line restoration, capacity reservation, etc

Early versions (non-capacitated):
Grötschel, Monma, Shallcross; many others in the algorithms community

Modern versions (capacitated):
Magnanti and Sokol, Bienstock and Muratore, Wessäly et al

Even more models

Routing restrictions. Examples:

- **Internet Protocol Routing.** (A. Bley, M. Prytz, many others). Example: OSPF, MPLS protocols.

  $\longrightarrow$ Routing algorithm must produce, for each link (i, j) a positive integer π_{ij}.
  All traffic must be routed on **shortest paths** using the metric π.

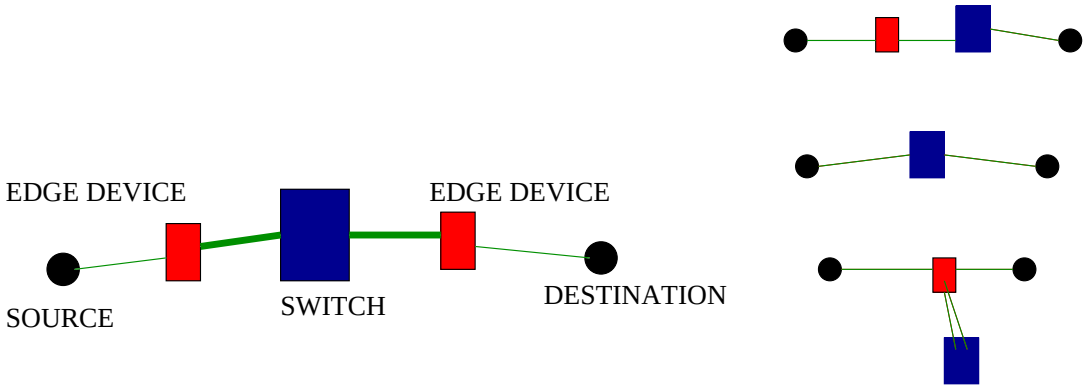
Even more models

Routing restrictions. Examples:

- **Internet Protocol Routing.** (A. Bley, M. Prytz, many others). Example: OSPF, MPLS protocols.

  $\longrightarrow$ Routing algorithm must produce, for each link (i, j) a positive integer π_{ij}.
  All traffic must be routed on **shortest paths** using the metric π.

- **ATM Routing.** Only certain paths may be used for routing, e.g.



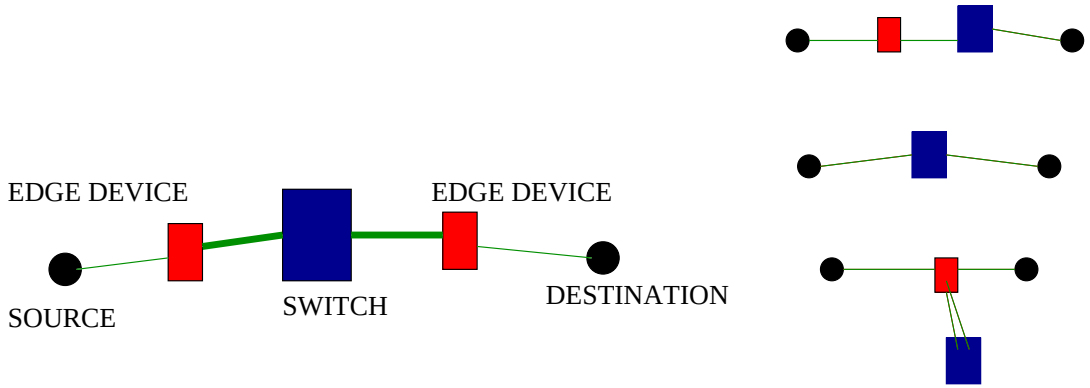
Even more models

Routing restrictions. Examples:

- **Internet Protocol Routing.** (A. Bley, M. Prytz, many others). Example: OSPF, MPLS protocols.

  → Routing algorithm must produce, for each link (i, j) a positive integer π_{ij}.
  All traffic must be routed on **shortest paths** using the metric π.

- **ATM Routing.** Only certain paths may be used for routing, e.g.



- **Wireless network design** (too many authors)

  → A. Eisenblaetter, frequency assignment in GSM networks

What is the track record?

- Specific problems or problem instances can **sometimes** be effectively solved
- Not infrequently, a lot of good mathematics can only provide an adequate solution.

An honest appraisal. Network design problems can be:

- Easy **(sometimes)**
- Hard **(usually)**
- Very hard **(not infrequently)**
- Impossible **(all too often)**

- They are also important

An old problem on optical networks (Bienstock and Günlük, 1990)

- Build a directed network to connect a set of nodes – the network will be used to route multicommodity demands
- Restriction: the nodes of the network can have up to $\boldsymbol{p}$ ingoing and outgoing links
- Route the demands as flows in the network

The objective of the problem is to simultaneously decide how to build the network and how to route the demands, so that the **maximum load is minimized**

$\Rightarrow$ Experimental observation: heuristics don't work

How about mixed-integer programming?

A mixed-integer programming formulation

- Variable: x_{ij} is a $0/1$ variable, $= 1$ when we use a link from i to j
- Variable: $f_{ij}^k \geq 0$ is the flow of commodity k on link (i, j).

(a) Constraint: for any node i, $\Sigma_j x_{ij} \leq p$ (or $= p$).

(b) Constraint: for any commodity k, and any arc (i, j), $f_{ij}^k \leq M x_{ij}$, where M is a large value

(c) Constraint: for any node i and commodity k, $\Sigma_j f_{ji}^k - \Sigma_j f_{ij}^k =$ demand of commodity k at i.

Problem:

Minimize z

Subject to:

$z \geq \Sigma_k f_{ij}^k$, for all (i, j)

(a), (b) and (c)

x is a $0/1$ vector and $f \geq 0$.

This is a hard problem

$\longrightarrow$ Instance **danoint** of MIPLIB III (p = 2; 8 nodes)

- 661 rows, 521 variables (56 0/1)
- Early 1990's: optimization problem can only be solved to within 4 % error, and only if special-purpose algorithms are used
- 2003: problem can now be solved with general-purpose mixed-integer solvers, in about **1 day CPU time**, enumerating about **1 million** branch-and-bound nodes

$\longrightarrow$ Modern general purpose MIP solver:
at least **one year CPU time** using 1993 hardware + LP solver

This is a hard problem

→ Instance **danoint** of MIPLIB III (p = 2; 8 nodes)

- 661 rows, 521 variables (56 0/1)
- Early 1990's: optimization problem can only be solved to within 4 % error, and only if special-purpose algorithms are used
- 2003: problem can now be solved with general-purpose mixed-integer solvers, in about **1 day CPU time**, enumerating about **1 million** branch-and-bound nodes

→ Larger instance **dano3mip**: 3202 constraints, 13873 variables (552 0/1)

- network has 20 nodes
- error gap is about 25 %
- beyond the reach of any current solver

The canonical problem – network loading

We are given a directed network, with

- A list of traffic demands to be routed: we must route d^k units of flow from node s^k to node t^k, $k = 1, 2, \ldots$,

- Each arc (i, j) has some *existing capacity* $u_{ij} \geq 0$,

- We can add an *integer* amount of capacity to each arc (i, j), at cost c_{ij} per unit.

Problem. Add capacity, at minimum cost, so that the demands can be feasibly routed.

Magnanti, Mirchandani, Vachani (1993), Bienstock, Chopra, Günlük, Tsai (1994), Barahona (1994), $\cdots$, Atamturk, Rajan (2002), Avella, Mattia, Sassano (2004)

Progress: 1994 - 2004

1994: Problem instances on 27 nodes, 102 arcs, 67 demands (from Bienstock, Chopra, Günlük, Tsai)
→ Cut & Branch on SUN Sparc 10, CPLEX 3.0

Name	lower	cut lower	best lower	upper	Gap (%)	Time (sec)
MS1	2753.4	2925.7	2933.2	2976.3	1.45	3600
MS2	2791.5	2943.1	2955.3	2978.2	0.77	3600
MS3	2829.5	2969.4	2978.9	2978.9	0.00	3600
MS4	3067.9	3225.9	3232.3	3256.8	0.75	3600
MS5	2997.6	3169.6	3174.8	3246.6	2.21	3600
MS6	2391.2	2563.4	2566.9	2592.4	0.98	3600

→ Cutting plane algorithm using

cut inequalities, 3-partitions (and a few other inequalities)

heuristics to find upper bounds

commercial branch-and-bound code

Progress: 1994 - 2004

1994: Problem instances on 27 nodes, 102 arcs, 67 demands (from Bienstock, Chopra, Günlük, Tsai)
→ Cut & Branch on SUN Sparc 10, CPLEX 3.0

Name	lower	cut lower	best lower	upper	Gap (%)	Time (sec)
MS1	2753.4	2925.7	2933.2	2976.3	1.45	3600
MS2	2791.5	2943.1	2955.3	2978.2	0.77	3600
MS3	2829.5	2969.4	2978.9	2978.9	0.00	3600
MS4	3067.9	3225.9	3232.3	3256.8	0.75	3600
MS5	2997.6	3169.6	3174.8	3246.6	2.21	3600
MS6	2391.2	2563.4	2566.9	2592.4	0.98	3600

2004: Avella, Mattia, Sassano
→ Branch & Cut on 1.7 GHz Pentium IV, CPLEX 8.0

Name	LB Flow	LB Cuts	BLB	UB Heur	BUB	Gap (%)	Nodes	Time (sec)
MS1	2738.78	2962.42	2962.42	2979.49	2962.42	0.00	5914	4523
MS2	2773.72	2976.24	2976.24	2980.71	2976.24	0.00	1928	1733
MS3	2824.11	2978.90	2978.90	2978.9	2978.90	0.00	46	7103
MS4	3038.38	3242.78	3242.78	3257.22	3242.78	0.00	426	1880
MS5	2979.25	3196.96	3196.96	3196.96	3196.96	0.00	4487	20464
MS6	2375.11	2585.00	2585.00	2609.87	2585.00	0.00	1591	8947

Progress: 1994 - 2004

1994: Problem instances on 27 nodes, 102 arcs, 67 demands (from Bienstock, Chopra, Günlük, Tsai)
→ Cut & Branch on SUN Sparc 10, CPLEX 3.0

Name	lower	cut lower	best lower	upper	Gap (%)	Time (sec)
MS1	2753.4	2925.7	2933.2	2976.3	1.45	3600
MS2	2791.5	2943.1	2955.3	2978.2	0.77	3600
MS3	2829.5	2969.4	2978.9	2978.9	0.00	3600
MS4	3067.9	3225.9	3232.3	3256.8	0.75	3600
MS5	2997.6	3169.6	3174.8	3246.6	2.21	3600
MS6	2391.2	2563.4	2566.9	2592.4	0.98	3600

2004: Avella, Mattia, Sassano
→ Branch & Cut on 1.7 GHz Pentium IV, CPLEX 8.0

Name	LB Flow	LB Cuts	BLB	UB Heur	BUB	Gap (%)	Nodes	Time (sec)
MS1	2738.78	2962.42	2962.42	2979.49	2962.42	0.00	5914	4523
MS2	2773.72	2976.24	2976.24	2980.71	2976.24	0.00	1928	1733
MS3	2824.11	2978.90	2978.90	2978.9	2978.90	0.00	46	7103
MS4	3038.38	3242.78	3242.78	3257.22	3242.78	0.00	426	1880
MS5	2979.25	3196.96	3196.96	3196.96	3196.96	0.00	4487	20464
MS6	2375.11	2585.00	2585.00	2609.87	2585.00	0.00	1591	8947

2005: CPLEX 9.0 on 2.8 GHz Xeon. After 6×10^5 seconds on MS5, gap $> 4\%$.

Formulation

$$\min \quad \textstyle\sum_{ij} c_{ij} x_{ij}$$

Subject to:

$$\sum_{(j,i)} f_{k,ji} - \sum_{(i,j)} f_{k,ij} = d_i^k \quad \forall k, i \tag{1}$$

$$\sum_{k} f_{k,ij} - x_{ij} \leq u_{ij} \quad \forall (i,j) \tag{2}$$

$$x_{ij} \in Z^+ \ \forall (i,j), \quad f_{k,ij} \geq 0 \quad \forall k, \ (i,j) \tag{3}$$

$\boldsymbol{d_i^k}$ = demand of commodity k at node i, $\boldsymbol{u_{ij}}$ = existing capacity in (i,j)
$\boldsymbol{f_{k,ij}}$ = flow of commodity k on (i,j); $\boldsymbol{x_{ij}}$ = capacity added to (i,j)

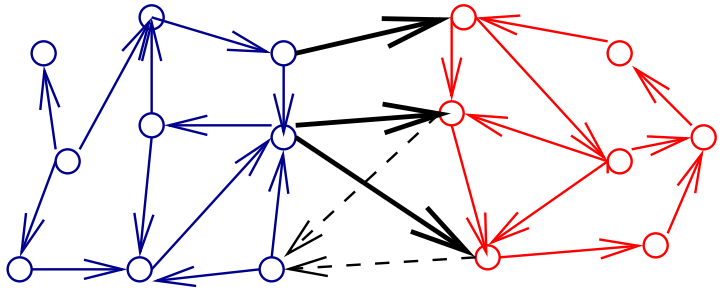
(Variations:

- Costs on the flows
- Different types ("modularities") of capacities
- Restriction on the allowable flows)

Cut-and-branch

Magnanti, Mirchandani, Vachani; Bienstock, Günlük

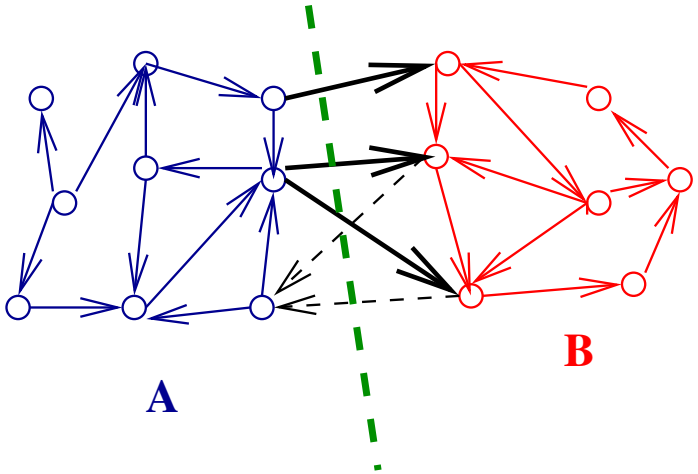
Given a partition $\boldsymbol{A}$, $\boldsymbol{B}$ of the nodes into 2 sets,



Cut-and-branch

Magnanti, Mirchandani, Vachani; Bienstock, Günlük

Given a partition $\boldsymbol{A}$, $\boldsymbol{B}$ of the nodes into 2 sets,



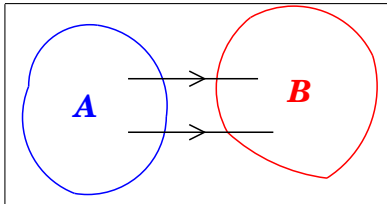
cut directed from A to B

$\rightarrow$ Capacity installed on arcs from $\boldsymbol{A}$ to $\boldsymbol{B}$, plus existing capacity, $\geq$ demand from $\boldsymbol{A}$ to $\boldsymbol{B}$

Cut-and-branch

Magnanti, Mirchandani, Vachani; Bienstock, Günlük

Given a partition $\boldsymbol{A}$, $\boldsymbol{B}$ of the nodes into 2 sets,



Cut inequality: $\boldsymbol{x(\delta_{A,B}) \geq \lceil D(A,B) - u(\delta_{A,B}) \rceil}$

Notation:

$\boldsymbol{x(\delta_{A,B})} = \Sigma_{i \in A, j \in B} x_{ij}$ = total capacity installed from $\boldsymbol{A}$ to $\boldsymbol{B}$

$\boldsymbol{u(\delta_{A,B})} = \Sigma_{i \in A, j \in B} u_{ij}$ = existing total capacity from $\boldsymbol{A}$ to $\boldsymbol{B}$

$\boldsymbol{D(A,B)}$ = total demand going from A to B

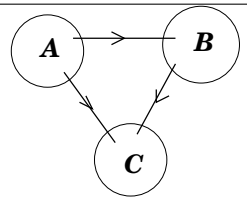
This is a strenghtening (Gomory rank-1) of: $\boldsymbol{x(\delta_{A,B}) \geq D(A,B) - u(\delta_{A,B})}$

Cut-and-branch

Magnanti, Mirchandani, Vachani; Bienstock, Günlük

Notation: $\boldsymbol{v(\delta_{A,B})} = \Sigma_{i\in A, j\in B}\, v_{ij}$; $\boldsymbol{D(A,B)}$ = total demand going from A to B

Given a partition $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$ of the nodes into 3 sets,



3-partition inequality:

$$\boldsymbol{x(\delta_{A,B}) + x(\delta_{B,C}) + x(\delta_{A,C}) \geq \lceil D(A,B) + D(A,C) + D(B,C) - u(\delta_{B,C}) - u(\delta_{B,C}) - u(\delta_{A,C})\rceil}$$

Also: k-partition inequalities, other configurations, mixed-integer rounding inequalities

→ Separating the cut inequalities is NP-hard (B. and Günlük)

→ So must rely on heuristics

What is the primary obstacle to running Cut-and-Branch?

→ The linear programs are remarkably hard in large cases (hundreds of nodes or more)

$$\begin{aligned} &\min \quad \Sigma_{ij}\, c_{ij} x_{ij} \\ &\text{Subject to:} \\ &\sum_{(j,i)} f_{k,ji} - \sum_{(i,j)} f_{k,ij} = d_i^k \quad \forall k, i \\ &\sum_{k} f_{k,ij} - x_{ij} \leq u_{ij} \quad \forall (i,j) \\ &x_{ij} \in Z^+ \ \forall (i,j), \quad f_{k,ij} \geq 0 \quad \forall k, (i,j) \end{aligned}$$

Nevertheless, the formulation is quite strong

Branch-and-cut using projection to x variables

Alevras, Grötschel, Stoer; Chopra, Tsai; Barahona; Avella, Mattia, Sassano

Key idea: only work with x variables; always maintain a "working formulation" $Ax \geq b$ of valid inequalities

Algorithm:

Step 1. Let x^* be an optimal solution to $\left\{\min c^T x \ : \ Ax \geq b,\ x \in Z_+\right\}$

Step 2. If there is a flow vector f^*, such that (x^*, f^*) are feasible for the network design problem, **STOP**.

Step 3. Else, find a valid inequality that is violated by x^*, add it to $Ax \geq b$, and go to **Step 1.**

Branch-and-cut using projection to x variables

Alevras, Grötschel, Stoer; Chopra, Tsai; Barahona; Avella, Mattia, Sassano

Key idea: only work with x variables; always maintain a "working formulation" $Ax \geq b$ of valid inequalities

Algorithm:

Step 1. Let x^* be an optimal solution to $\left\{\min c^T x \ : \ Ax \geq b,\ x \in Z_+\right\}$

Step 2. If there is a flow vector f^*, such that (x^*, f^*) are feasible for the network design problem, **STOP**.

Step 3. Else, find a valid inequality that is violated by x^*, add it to $Ax \geq b$, and go to **Step 1.**

Better Algorithm:

Step 1. Let x^* be an optimal solution to $\left\{\min c^T x \ : \ Ax \geq b,\ x \geq 0\right\}$

Step 2. If we can find a valid inequality that is violated by x^*, add it to $Ax \geq b$ and go to **Step 1.**

Step 3. Otherwise, if $x^* \in Z_+$ **STOP** – we have solved the problem.

Step 4. Otherwise, ***branch*** on some fractional variable x^*_j.

→ How do we implement **Step 2**?

One possibility: heuristics (Chopra and Tsai), separate cut inequalities by solving max-cut problem (Barahona)

Farkas' Lemma

Question: given a capacity vector $\boldsymbol{x^*}$, is there a flow vector $\boldsymbol{f}$ such that:

$$\begin{aligned} \sum_{(j,i)} f_{k,ji} - \sum_{(i,j)} f_{k,ij} &= d_i^k && \forall k, i \\ \sum_{k} f_{k,ij} &\le x^*_{ij} + u_{ij} && \forall (i,j) \\ f_{k,ij} &\ge 0 && \forall k,\ (i,j) \end{aligned}$$

Answer: there is ***no*** such vector, if and only if there are quantities $\boldsymbol{\pi_{ij} \ge 0}$ (for each arc $\boldsymbol{(i,j)}$) such that:

$$\boldsymbol{\Sigma_{ij}\, \pi_{ij}(x^*_{ij} + u_{ij}) \;<\; \Sigma_{i,k}\, L_i^k(\pi)\, d_i^k}$$

Here, $\boldsymbol{L_i^k(\pi)}$ = shortest path length, to node $\boldsymbol{i}$, from the source of commodity $\boldsymbol{k}$, using arc lengths $\boldsymbol{\pi}$.

A **metric** inequality: $\boldsymbol{\Sigma_{ij}\, \pi_{ij} x_{ij} \;\ge\; \Sigma_{i,k}\, L_i^k(\pi)\, d_i^k - \Sigma_{i,j} \pi_{ij} u_{ij}}$

(Lomonosov, Iri, Laurent, ...) Cut inequalities, partition inequalities, are all metric

Farkas' Lemma

Question: given a capacity vector $\boldsymbol{x^*}$, is there a flow vector $\boldsymbol{f}$ such that:

$$\begin{aligned} \sum_{(j,i)} f_{k,ji} - \sum_{(i,j)} f_{k,ij} &= d_i^k && \forall k, i \\ \sum_k f_{k,ij} &\le x_{ij}^* + u_{ij} && \forall (i,j) \\ f_{k,ij} &\ge 0 && \forall k,\ (i,j) \end{aligned}$$

Answer: there is ***no*** such vector, if and only if there are quantities $\boldsymbol{\pi_{ij} \ge 0}$ (for each arc $\boldsymbol{(i,j)}$) such that:

$$\boldsymbol{\Sigma_{ij}\, \pi_{ij}(x_{ij}^* + u_{ij}) \ < \ \Sigma_{i,k}\, L_i^k(\pi)\, d_i^k}$$

Here, $\boldsymbol{L_i^k(\pi)}$ = shortest path length, to node $\boldsymbol{i}$, from the source of commodity $\boldsymbol{k}$, using arc lengths $\boldsymbol{\pi}$.

Equivalently: solve the linear program

$$\min \quad \Sigma_{ij}(x_{ij}^* + u_{ij})\pi_{ij}$$

Subject to:

$$\begin{aligned} \Sigma_{(i,j)} L_j^k - L_i^k - \pi_{ij} &\le 0, \quad \forall k,\ (i,j) \\ \Sigma_{i,k}\, d_i^k L_i^k &= 1 \\ L \ge 0, \pi &\ge 0 \end{aligned}$$

If $\boldsymbol{\Sigma_{ij}(x_{ij}^* + u_{ij})\pi_{ij} < 1}$, violation

In that case, can usually tighten:

$$\boldsymbol{\Sigma_{ij} \lceil \pi_{ij} \rceil x_{ij} \ \ge \ \lceil 1 - \Sigma_{ij} \pi_{ij} u_{ij} \rceil}$$

(Avella, Mattia, Sassano)

Farkas' Lemma

Question: given a capacity vector $\boldsymbol{x^*}$, is there a flow vector $\boldsymbol{f}$ such that:

$$\begin{aligned} \sum_{(j,i)} f_{k,ji} - \sum_{(i,j)} f_{k,ij} &= d_i^k && \forall k, i \\ \sum_{k} f_{k,ij} &\le x_{ij}^* + u_{ij} && \forall (i,j) \\ f_{k,ij} &\ge 0 && \forall k,\ (i,j) \end{aligned}$$

Answer: there is ***no*** such vector, if and only if there are quantities $\boldsymbol{\pi_{ij} \ge 0}$ (for each arc $\boldsymbol{(i,j)}$) such that:

$$\boldsymbol{\Sigma_{ij}\, \pi_{ij}(x_{ij}^* + u_{ij}) \ < \ \Sigma_{i,k}\, L_i^k(\pi)\, d_i^k}$$

Here, $\boldsymbol{L_i^k(\pi)}$ = shortest path length, to node $\boldsymbol{i}$, from the source of commodity $\boldsymbol{k}$, using arc lengths $\boldsymbol{\pi}$.

Equivalently: solve the linear program

$$\min \quad \Sigma_{ij}(x_{ij}^* + u_{ij})\pi_{ij}$$

Subject to:

$$\begin{aligned} \Sigma_{(i,j)} L_j^k - L_i^k - \pi_{ij} &\le 0, \quad \forall k,\ (i,j) \\ \Sigma_{i,k}\, d_i^k L_i^k &= 1 \\ L \ge 0, \pi &\ge 0 \end{aligned}$$

If $\boldsymbol{\Sigma_{ij}(x_{ij}^* + u_{ij})\pi_{ij} < 1}$, violation

In that case, can usually tighten:

$$\boldsymbol{\Sigma_{ij} \lceil \pi_{ij} \rceil x_{ij} \ \ge \ \lceil 1 - \Sigma_{ij} \pi_{ij} u_{ij} \rceil}$$

(Avella, Mattia, Sassano)

$\Rightarrow$ Obstacle to this approach: **the bad LP (again)**

But is it bad? It is a **maximum concurrent flow** problem

The maximum concurrent flow problem – generic version

Definition: Given a network with capacities on the arcs, and multicommodity demands to be routed; find the maximum common fraction $\boldsymbol{\theta}$ of all demands that can be simultaneously routed.

$$\begin{aligned} & \max \quad \theta \\ \text{s.t.} \quad & \sum_{(j,i)} f_{k,ji} - \sum_{(i,j)} f_{k,ij} - \theta d_i^k = 0 \qquad \forall k, i \\ & \sum_{k} f_{k,ij} \leq u_{ij} \qquad \forall (i,j) \\ & f_{k,ij} \geq 0 \qquad \forall k, \, (i,j) \end{aligned}$$

Very difficult as a linear program

But many results regarding approximation algorithms

- **1971:** Fratta, Gerla and Kleinrock (flow deviation method)
- **1991:** Shahrokhi and Matula (exponential potential function). Relative error ϵ in time $O(\epsilon^{-5})$
- **1991-1994:** Grigoriadis and Khachiyan; Plotkin, Shmoys and Tardos (exponential potential function), $O(\epsilon^{-2})$
- **1998-1999:** Garg and Könemann, Fleischer (elaborations)
- **2003:** Nesterov (new ideas from non-differentiable optimization)
- **2004:** Bienstock and Iyengar: rel. error ϵ in time $O(\epsilon^{-1})$
- **2005:** Nemirovskii?

Approximation algorithms – rough outline

All algorithms resort to approximately solved related problem

$$\begin{aligned} \min \quad & \textstyle\sum_{(i,j)} \Phi\left(\frac{\sum_k f_{k,i,j}}{u_{ij}}\right) \\ \text{s.t.} \quad & \sum_{(j,i)} f_{k,ji} - \sum_{(i,j)} f_{k,ij} = d_i^k \qquad \forall k, i \\ & 0 \le f_{k,ij} \le u_{ij} \qquad \forall k\,(i,j) \end{aligned}$$

Where $\mathbf{\Phi}$ is an appropriate ***potential*** function

$\rightarrow$ How is $\mathbf{\Phi}$ approximately minimized?

Either:

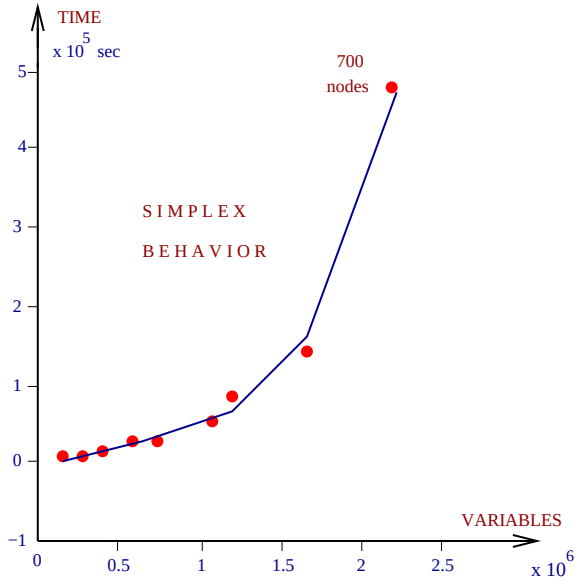
- First-order method: this gives rise to a sequence of **shortest-path** problems, or single-commodity linear min cost flow problems
- Quasi-Newton like methods: convex, separable single commodity min cost flow problems

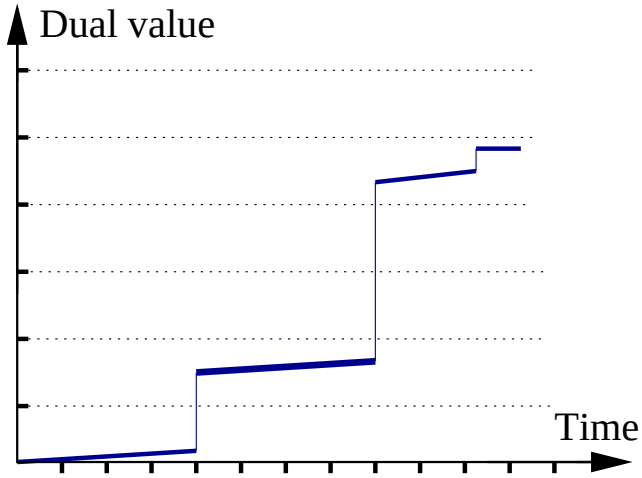
$\Rightarrow$ the methods are **scalable**

Approximation algorithms vs standard LP

Tests performed on a recent workstation

Name	nodes	arcs	comm.	Cplex Dual (sec)	$\mathbf{1.0e^{-04}}$ Approx (sec.)
RMF23	484	2123	5000	40368	746
RMF19	500	2200	5000	66112	1024
RMF25	600	2660	5000	137685	1303
RMF26	700	3120	5000	445541	2103
RMF27	1000	4500	10000	1734618	5487





Back to Farkas' Lemma

Question: given a capacity vector $\boldsymbol{x}^*$, is there a flow vector $\boldsymbol{f}$ such that:

$$\begin{aligned} \sum_{(j,i)} f_{k,ji} - \sum_{(i,j)} f_{k,ij} &= d_i^k && \forall k, i \\ \sum_{k} f_{k,ij} &\leq x_{ij}^* + u_{ij} && \forall (i,j) \\ f_{k,ij} &\geq 0 && \forall k,\ (i,j) \end{aligned}$$

Answer: there is ***no*** such vector, if and only if there are quantities $\boldsymbol{\pi_{ij} \geq 0}$ (for each arc $\boldsymbol{(i,j)}$) such that:

$$\boldsymbol{\Sigma_{ij}\, \pi_{ij}(x_{ij}^* + u_{ij}) < \Sigma_{i,k}\, L_i^k(\pi)\, d_i^k}$$

Here, $\boldsymbol{L_i^k(\pi)}$ = shortest path, to node $\boldsymbol{i}$, from the source of commodity $\boldsymbol{k}$, using arc lengths $\boldsymbol{\pi}$

Stronger valid inequalities for network design

(B. and Mattia)

Question: given a capacity vector x^*, is there a flow vector f for which the capacities $u_{ij} + x^*_{ij}$ suffice?

Let (p, q) be an arc with x^*_{pq} ***fractional***

Stronger question: are there vectors (x^1, f^1) and (x^2, f^2), each feasible for:

$$\begin{aligned} \sum_{(j,i)} f_{k,ji} - \sum_{(i,j)} f_{k,ij} &= d^k_i \quad \forall k, i \\ \sum_k f_{k,ij} - x_{ij} &\le u_{ij} \quad \forall (i,j) \\ x_{ij},\ f_{k,ij} &\ge 0 \quad \forall k, (i,j) \end{aligned}$$

and such that $x^1_{pq} = \lfloor x^*_{pq} \rfloor$ and $x^2_{pq} = \lceil x^*_{pq} \rceil$

and $x^* = \lambda\, x^1 + (1-\lambda)\, x^2$, for some $0 < \lambda < 1$?

Disjunctive programming, lift-and-project (Balas; Balas, Ceria, Cornuéjols)

$\Rightarrow$ The separation problem is twice as large

Stronger valid inequalities for network design

(B. and Mattia)

Question: given a capacity vector x^*, is there a flow vector f for which the capacities $u_{ij} + x^*_{ij}$ suffice?

Let (p, q) be an arc with x^*_{pq} ***fractional***

Stronger question: are there vectors (x^1, f^1) and (x^2, f^2), each feasible for:

$$\begin{aligned} \sum_{(j,i)} f_{k,ji} - \sum_{(i,j)} f_{k,ij} &= d^k_i \quad \forall k, i \\ \sum_k f_{k,ij} - x_{ij} &\le u_{ij} \quad \forall (i,j) \\ x_{ij},\ f_{k,ij} &\ge 0 \quad \forall k, (i,j) \end{aligned}$$

and such that $x^1_{pq} = \lfloor x^*_{pq} \rfloor$ and $x^2_{pq} = \lceil x^*_{pq} \rceil$

and $x^* = \lambda\, x^1 + (1-\lambda)\, x^2$, for some $0 < \lambda < 1$?

Disjunctive programming, lift-and-project (Balas; Balas, Ceria, Cornuéjols)

$\Rightarrow$ The separation problem is twice as large

Further strengthening is possible in the **0/1** case (**x a 0/1 vector**), e.g.

$$5x_a + 4x_b + x_c + x_d \ge 8 \quad \text{implies} \quad 2x_b + x_c + x_d \ge 2$$

(follows from $x_b + x_c \ge 1$, $x_b + x_d \ge 1$)

Stronger formulations for set covering-like problems

(B. and Zuckerberg, 2002)

Motivated by "lift-and-project" algorithms: Balas; Lovász and Schrijver, Sherali and Adams; Balas, Ceria, Cornuéjols, Lasserre

$\longrightarrow$ Expand the formulation of a **0/1**-integer program by adding new variables. The new variables express logical relationships between the other variables (and constraints).

Example: Suppose we have the inequalities:

$$\begin{aligned} x_1 + x_2 + x_3 + x_4 &\geq 1 \\ x_1 + x_2 + x_3 + x_5 &\geq 1 \end{aligned}$$

New variables:

- $\boldsymbol{\alpha} = 1$ when $\boldsymbol{x_1 = x_2 = x_3 = 0}$
- $\boldsymbol{\beta_1} = 1$ when $\boldsymbol{x_1 = 1,\ x_2 = x_3 = 0}$ (similarly, $\boldsymbol{\beta_2, \beta_3}$)
- $\boldsymbol{\nu} = 1$ when at least **two** of $\boldsymbol{x_1, x_2, x_3}$ equal $\mathbf{1}$
- $\boldsymbol{\alpha}^{(j)} = \boldsymbol{\alpha}\, \boldsymbol{x_j}$ (all j), $\boldsymbol{\nu}^{(j)} = \boldsymbol{\nu}\, \boldsymbol{x_j}$ (all j)

Then we can impose, e.g.:

$$\begin{aligned} \alpha + \sum_j \beta_j + \nu &= 1 \\ \sum_{j=1} \nu^{(j)} &\geq 2\nu \end{aligned}$$

Consequences

Theorem (B. and Zuckerberg, 2002): Given any set covering problem and any integer $k \geq 1$, in polynomial time we can generate a linear program each of whose solutions satisfies all valid inequalities with coefficients in $\{0, 1, 2, \ldots, k\}$.

$\rightarrow$ $k = 2$ requires **exponential** time using Sherali-Adams or Lovász-Schrijver

Theorem (B. and Zuckerberg, 2003): Given any set covering problem, an integer $r \geq 1$, and $0 < \epsilon < 1$, in polynomial time we can solve the rank- r Chvátal-Gomory relaxation to tolerance ϵ.

$\rightarrow$ $r = 1$ and $\epsilon = \frac{1}{2}$ requires **exponential** time using Sherali-Adams or Lovász-Schrijver.

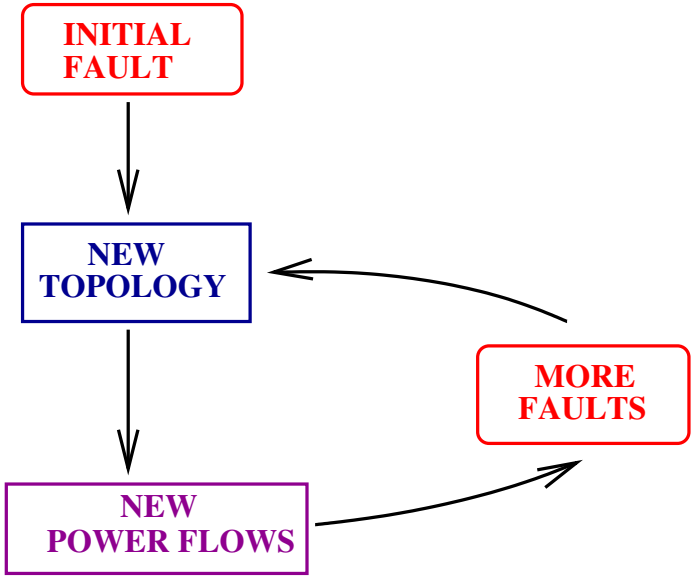
Ongoing work: network design for electrical transmission networks

(with S. Mattia)

Large-scale blackouts: Brazil (1999), North America (2003), France-Switzerland-Italy (2003)

The facts:

- In an electrical network, power flows obey the **laws of physics**
- If a fault occurs, the network topology is effectively changed
- In the new topology, the resulting power flows may overwhelm equipment
- This may trigger a new set of faults, or equipment shut-downs


INITIAL FAULT
NEW TOPOLOGY
MORE FAULTS
NEW POWER FLOWS

Ongoing work: network design for electrical transmission networks

Large-scale blackouts: Brazil (1999), North America (2003), France-Switzerland-Italy (2003)

The facts:

- In an electrical network, power flows obey the **laws of physics**
- If a fault occurs, the network topology is effectively changed
- In the new topology, the resulting power flows may overwhelm equipment
- This may trigger a new set of faults, or equipment shut-downs
- The process can **cascade**

Problem 1: how to upgrade a network, at low cost, so that the probability of a catastrophic cascade is low

Problem 2: how to react in real-time