Basic Network Design: Single Commodity Flows with Uncertain Demands

Valentina Cacchiani[1] Michael Jünger[3] Frauke Liers[2]
Andrea Lodi[1] Daniel R. Schmidt[3]

[1]DEI, Università di Bologna

[2]Department Mathematik, Universität Erlangen-Nürnberg

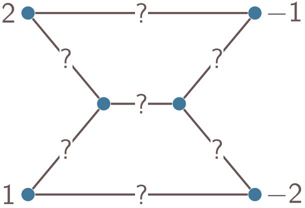
[3]Institut für Informatik, Universität zu Köln

Aussois, 10.01.2014

First: A non-robust Model

Input: network, supplies and demands, cost function

$G = (V, E)$

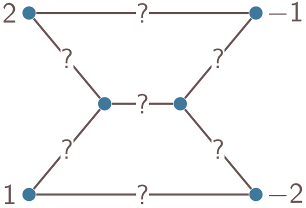


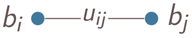
$\sum_{i \in V} b_i = 0$

First: A non-robust Model

Input: network, supplies and demands, cost function

$G = (V, E)$



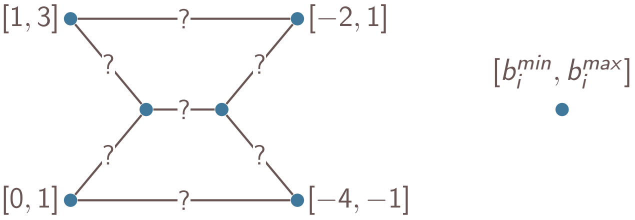


$\sum_{i \in V} b_i = 0$

Task: Find integer capacities such that all supplies/demands can be balanced with single-commodity b-flow.

Single Commodity Robust Network Design (sRND)

Input: network, uncertainty set $\mathcal{D}$, each *scenario* $b \in \mathcal{D}$ defines supplies/demands

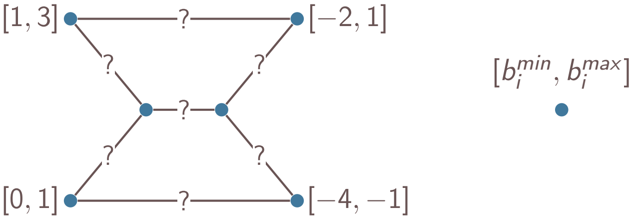


Here: $\mathcal{D} := \left\{ b \in \mathbb{R}^V \;\middle|\; \forall\, i \in V : b_i \in [b_i^{min}, b_i^{max}] \wedge \sum_{i \in V} b_i = 0 \right\}$

$:= \mathcal{H}(b^{min}, b^{max})$

Single Commodity Robust Network Design (sRND)

Input: network, uncertainty set $\mathcal{D}$, each *scenario* $b \in \mathcal{D}$ defines supplies/demands



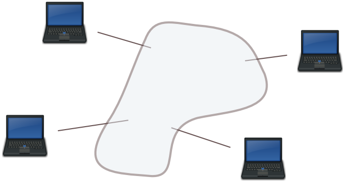
Here: $\mathcal{D} := \left\{ b \in \mathbb{R}^V \;\middle|\; \forall\, i \in V : b_i \in [b_i^{min}, b_i^{max}] \wedge \sum_{i \in V} b_i = 0 \right\}$

$:= \mathcal{H}(b^{min}, b^{max})$

Task: Find cheapest integer capacities such, that in all scenarios all supplies/demands can be balanced!

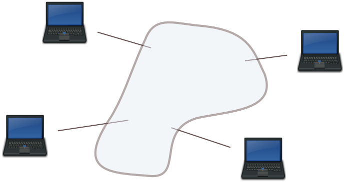
What is special here?

Theme: "Design a minimum cost network such that all traffic requests can be routed!"



What is special here?

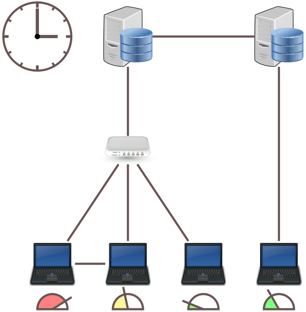
Theme: "Design a minimum cost network such that all traffic requests can be routed!"



The usual model

- users send point-to-point data *through* network
- pairwise communication
- Here: users get data *from* network

What is special here!



Model [Buchheim, Liers, Sanità 2011]

- servers with identical data ("supply")
- users request data ("demand")
- all servers may answer user requests

Applications

Can be used for...

- Streaming (Netflix, Maxdome...)
- Cloud storage
- Software distribution from mirrors
- (Energy networks)
- ...

Cut-Set formulation

Theorem (ACDJLLPS 2012)

Any instance $(G, c, \mathcal{D})$ of the sRND can be written as the following integer linear program.

$$\min \sum_{\{i,j\} \in E} c_{ij} u_{ij}$$

$$\sum_{\{i,j\} \in \delta(S)} u_{ij} \geq \max_{b \in \mathcal{D}} \left| \sum_{i \in S} b_i \right| \qquad \text{for all } S \subseteq V$$

$$u_{ij} \in \mathbb{Z}_{\geq 0} \qquad \text{for all } \{i,j\} \in E$$

How to solve (robust) LP relaxation? First ideas.

$$\min \sum_{\{i,j\} \in E} c_{ij} u_{ij}$$

$$\sum_{\{i,j\} \in \delta(S)} u_{ij} \geq \max_{b \in \mathcal{H}(b^{min}, b^{max})} \left| \sum_{i \in S} b_i \right| \quad \text{for all } S \subseteq V$$

$$u_{ij} \in \mathbb{R}_{\geq 0} \quad \text{for all } \{i,j\} \in E$$

How to solve (robust) LP relaxation? First ideas.

$$\min \sum_{\{i,j\}\in E} c_{ij} u_{ij}$$

$$\sum_{\{i,j\}\in\delta(S)} u_{ij} \geq \max_{b\in\mathcal{H}(b^{min},b^{max})} \left|\sum_{i\in S} b_i\right| \quad \text{for all } S \subseteq V$$

$$u_{ij} \in \mathbb{R}_{\geq 0} \quad \text{for all } \{i,j\} \in E$$

Approach I: Dualize the right hand side.

How to solve (robust) LP relaxation? First ideas.

$$\min \sum_{\{i,j\}\in E} c_{ij} u_{ij}$$

$$\sum_{\{i,j\}\in\delta(S)} u_{ij} \geq \max_{b\in\mathcal{H}(b^{min},b^{max})} \left|\sum_{i\in S} b_i\right| \quad \text{for all } S \subseteq V$$

$$u_{ij} \in \mathbb{R}_{\geq 0} \quad \text{for all } \{i,j\} \in E$$

Approach I: Dualize the right hand side.

[AABP2007] Static routing, yields linear program + separation

How to solve (robust) LP relaxation? First ideas.

$$\min \sum_{\{i,j\}\in E} c_{ij} u_{ij}$$

$$\sum_{\{i,j\}\in\delta(S)} u_{ij} \geq \max_{b\in\mathcal{H}(b^{min},b^{max})} \left|\sum_{i\in S} b_i\right| \quad \text{for all } S \subseteq V$$

$$u_{ij} \in \mathbb{R}_{\geq 0} \quad \text{for all } \{i,j\} \in E$$

Approach I: Dualize the right hand side.

[AABP2007] Static routing, yields linear program + separation

[Mattia2010] Dynamic routing, yields non-convex quadratic program

How to solve (robust) LP relaxation? First ideas.

$$\min \sum_{\{i,j\} \in E} c_{ij} u_{ij}$$

$$\sum_{\{i,j\} \in \delta(S)} u_{ij} \geq \max_{b \in \mathcal{H}(b^{min}, b^{max})} \left| \sum_{i \in S} b_i \right| \quad \text{for all } S \subseteq V$$

$$u_{ij} \in \mathbb{R}_{\geq 0} \quad \text{for all } \{i,j\} \in E$$

Approach I: Dualize the right hand side.

[AABP2007] Static routing, yields linear program + separation

[Mattia2010] Dynamic routing, yields non-convex quadratic program

Here: Quadratic, non-convex program or exponentially many variables

How to solve (robust) LP relaxation? First ideas.

$$\min \sum_{\{i,j\}\in E} c_{ij} u_{ij}$$

$$\sum_{\{i,j\}\in\delta(S)} u_{ij} \geq \max_{b\in\mathcal{H}(b^{min},b^{max})} \left|\sum_{i\in S} b_i\right| \quad \text{for all } S \subseteq V$$

$$u_{ij} \in \mathbb{R}_{\geq 0} \quad \text{for all } \{i,j\} \in E$$

Approach II: General technique for robust LPs.

- Due to [BTN99], works for *separable* $\mathcal{D}$

How to solve (robust) LP relaxation? First ideas.

$$\min \sum_{\{i,j\} \in E} c_{ij} u_{ij}$$

$$\sum_{\{i,j\} \in \delta(S)} u_{ij} \geq \max_{b \in \mathcal{H}(b^{min}, b^{max})} \left| \sum_{i \in S} b_i \right| \quad \text{for all } S \subseteq V$$

$$u_{ij} \in \mathbb{R}_{\geq 0} \quad \text{for all } \{i,j\} \in E$$

Approach II: General technique for robust LPs.

- Due to [BTN99], works for *separable* $\mathcal{D}$
- Solve auxilliary LP for each constraint

How to solve (robust) LP relaxation? First ideas.

$$\min \sum_{\{i,j\}\in E} c_{ij} u_{ij}$$

$$\sum_{\{i,j\}\in\delta(S)} u_{ij} \geq \max_{b\in\mathcal{H}(b^{min},b^{max})} \left|\sum_{i\in S} b_i\right| \quad \text{for all } S \subseteq V$$

$$u_{ij} \in \mathbb{R}_{\geq 0} \quad \text{for all } \{i,j\} \in E$$

Approach II: General technique for robust LPs.

- Due to [BTN99], works for *separable* $\mathcal{D}$
- Solve auxilliary LP for each constraint
- Algorithm is polynomial in size of deterministic formulation

How to solve (robust) LP relaxation? First ideas.

$$\min \sum_{\{i,j\}\in E} c_{ij} u_{ij}$$

$$\sum_{\{i,j\}\in\delta(S)} u_{ij} \geq \max_{b\in\mathcal{H}(b^{min},b^{max})} \left|\sum_{i\in S} b_i\right| \quad \text{for all } S \subseteq V$$

$$u_{ij} \in \mathbb{R}_{\geq 0} \quad \text{for all } \{i,j\} \in E$$

Approach II: General technique for robust LPs.

- Due to [BTN99], works for *separable* $\mathcal{D}$
- Solve auxilliary LP for each constraint
- Algorithm is polynomial in size of deterministic formulation
- Here: Doesn't help.

The separation problem

Current solution u^* violates cut-set inequality if and only if

$$\min_{S \subseteq V} \Big[\sum_{e \in \delta(S)} u^*_e - \max\Big\{ \Big| \sum_{i \in S} b_i \Big| : b \in \mathcal{H}(b^{min}, b^{max}) \Big\} \Big] < 0$$

The separation problem

Current solution u^* violates cut-set inequality if and only if

$$\min_{S \subseteq V} \Big[\sum_{e \in \delta(S)} u^*_e - \max\Big\{ \Big| \sum_{i \in S} b_i \Big| : b \in \mathcal{H}(b^{min}, b^{max}) \Big\} \Big] < 0$$

- Problem is NP-hard (solves Equicut)
- Straight-forward formulation as integer program is quadratic

The separation problem

Current solution u^* violates cut-set inequality if and only if

$$\min_{S \subseteq V} \Big[\sum_{e \in \delta(S)} u_e^* - \max\Big\{ \Big| \sum_{i \in S} b_i \Big| : b \in \mathcal{H}(b^{min}, b^{max}) \Big\} \Big] < 0$$

- Problem is NP-hard (solves Equicut)
- Straight-forward formulation as integer program is quadratic

Our approach.

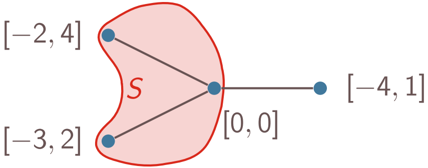
- Fix $S \subseteq V$ and solve

$$\max\Big\{ \Big| \sum_{i \in S} b_i \Big| \; \Big| \; b_i^{min} \leq b_i \leq b_i^{max} \; \forall \, i \in V, \; \sum_{i \in V} b_i = 0 \Big\}$$

- Integrate solution in integer program
- Make sure it becomes *linear*

Intuition: How to attain the maximum.

$$\max\Big\{\big|\sum_{i\in S} b_i\big| \;\Big|\; b_i^{min} \le b_i \le b_i^{max}\ \forall\, i \in V,\ \sum_{i\in V} b_i = 0\Big\}$$

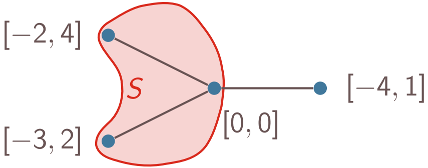


Two possible cases for some optimum b^*:

A) S is a source set: $\sum_{i\in S} b^* \ge 0$ *or*
B) S is a sink set: $\sum_{i\in S} b^* \le 0$.

Intuition: How to attain the maximum.

$$\max\Big\{\Big|\sum_{i\in S} b_i\Big| \;\Big|\; b_i^{min} \leq b_i \leq b_i^{max}\ \forall\, i \in V,\ \sum_{i\in V} b_i = 0\Big\}$$

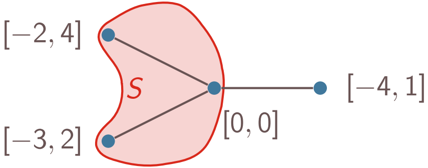


Two possible cases for some optimum b^*:

A) S is a source set: $\sum_{i\in S} b^* \geq 0$ *or*
B) S is a sink set: $\sum_{i\in S} b^* \leq 0$.

S is source set $\iff$ $V \setminus S$ is sink set *of the same value!*
w.l.o.g. Only consider source sets.

When S is a source set...

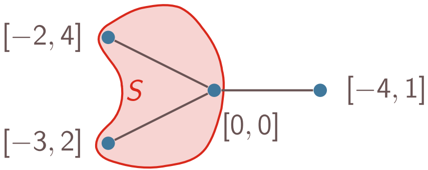


For any optimum b^*:

$$b_i^* = b_i^{max} \quad \text{for all } i \in S \qquad \Rightarrow \left|\sum_{i\in S} b_i^*\right| = \left|\sum_{i \in S} b_i^{max}\right|$$

$$\text{or} \quad b_i^* = b_i^{min} \quad \text{for all } i \in V \setminus S \qquad \Rightarrow \left|\sum_{i\in S} b_i^*\right| = \left|\sum_{i \in V\setminus S} b_i^{min}\right|$$

When S is a source set...



For any optimum b^*:

$$b_i^* = b_i^{max} \quad \text{for all } i \in S \qquad \Rightarrow \left|\sum_{i\in S} b_i^*\right| = \left|\sum_{i\in S} b_i^{max}\right|$$

$$\text{or} \quad b_i^* = b_i^{min} \quad \text{for all } i \in V \setminus S \quad \Rightarrow \left|\sum_{i\in S} b_i^*\right| = \left|\sum_{i\in V\setminus S} b_i^{min}\right|$$

The limiting set determines the objective value!

$$\sum_{i\in S} b_i^* = \min\left\{\sum_{i\in S} b_i^{max}, -\sum_{i\in V\setminus S} b_i^{min}\right\}$$

Putting all pieces together...

Starting from...

$$\min_{S \subseteq V} \left[\sum_{e \in \delta(S)} u_e^* - \max\left\{ \left| \sum_{i \in S} b_i \right| : b \in \mathcal{H}(b^{min}, b^{max}) \right\} \right]$$

Putting all pieces together...

Starting from...

$$\min_{S\subseteq V}\Big[\sum_{e\in\delta(S)} u_e^* - \max\Big\{\Big|\sum_{i\in S} b_i\Big| : b\in\mathcal{H}(b^{min}, b^{max})\Big\}\Big]$$

...we restrict to source sets...

$$= \min_{S\subseteq V}\Big[\sum_{e\in\delta(S)} u_e^* - \max\Big\{\sum_{i\in S} b_i : b\in\mathcal{H}(b^{min}, b^{max}), \sum_{i\in S} b_i \geq 0\Big\}\Big]$$

Putting all pieces together...

Starting from...

$$\min_{S \subseteq V} \Big[\sum_{e \in \delta(S)} u_e^* - \max \Big\{ \Big| \sum_{i \in S} b_i \Big| : b \in \mathcal{H}(b^{min}, b^{max}) \Big\} \Big]$$

...we restrict to source sets...

$$= \min_{S \subseteq V} \Big[\sum_{e \in \delta(S)} u_e^* - \max \Big\{ \sum_{i \in S} b_i : b \in \mathcal{H}(b^{min}, b^{max}), \sum_{i \in S} b_i \geq 0 \Big\} \Big]$$

...we characterize the optimum...

$$= \min_{S \subseteq V} \Big[\sum_{e \in \delta(S)} u_e^* - \min \Big\{ \sum_{i \in S} b_i^{max}, - \sum_{i \in V \setminus S} b_i^{min} \Big\} \Big]$$

And now: Our MIP

We rewrite

$$\min_{S \subseteq V} \Big[\sum_{e \in \delta(S)} u_e^* - \min\Big\{ \sum_{i \in S} b_i^{max}, -\sum_{i \in V \setminus S} b_i^{min} \Big\} \Big]$$

as

$$\min \sum_{\{i,j\} \in E} u_{ij}^* y_{ij} - B$$

$$B \leq \sum_{i \in V} x_i b_i^{max}$$

$$B \leq -\sum_{i \in V} (1 - x_i) b_i^{min}$$

$$y_{ij} \geq x_i - x_j \qquad \text{for all } \{i,j\} \in E$$

$$y_{ij} \geq x_j - x_i \qquad \text{for all } \{i,j\} \in E$$

$$x_i \in \{0,1\} \qquad \text{for all } i \in V$$

$$y_{ij} \in \{0,1\} \qquad \text{for all } \{i,j\} \in E$$

More ingredients

Speeding up the separation

- For finite $\mathcal{D}$ [ACDJLLPS 2012] separate cs-inequalities in time

$$O(|\mathcal{D}| \cdot T_{mincut})$$

- $\rightsquigarrow$ Keep list of non-routable scenarios
- $\rightsquigarrow$ Call polynomial separation on list first.

Non-routable scenarios

How to prove insuffiency of S?

- Cut-set inequality defined by cut S and scenario b
- Separation only gives us S and value of b
- Want: Actual scenario that cannot be routed through S

Non-routable scenarios

How to prove insuffiency of S?

- Cut-set inequality defined by cut S and scenario b
- Separation only gives us S and value of b
- Want: Actual scenario that cannot be routed through S

Linear time greedy algorithm (assuming S is limiting)

1. Set $b_i = b_i^{max}$ for all $i \in S$
2. Set b_i to legal value closest to zero for all $i \in V \setminus S$
3. Make b_i balanced by decreasing b_i for $i \in V \setminus S$

Non-routable scenarios

How to prove insuffiency of S?

- Cut-set inequality defined by cut S and scenario b
- Separation only gives us S and value of b
- Want: Actual scenario that cannot be routed through S

Linear time greedy algorithm (assuming S is limiting)

1. Set $b_i = b_i^{max}$ for all $i \in S$
2. Set b_i to legal value closest to zero for all $i \in V \setminus S$
3. Make b_i balanced by decreasing b_i for $i \in V \setminus S$

Nice property: Can yield several scenarios if called repeatedly.

The *Bound* in Branch-And-Cut

Improving the bounds

- Primal rounding heuristics (Simple, SP)
- 3-partition inequalities
- Zero-Half-Cut implementation by [ACF]

Setting

Environment

- 2.33 Ghz Intel Xeon, 3gb RAM
- ABACUS 3.2
- CPLEX 12.1
- time limit: 4 hours

Setting

Environment

- 2.33 Ghz Intel Xeon, 3gb RAM
- ABACUS 3.2
- CPLEX 12.1
- time limit: 4 hours

Instances

- Randomly generated values for b^{min}, b^{max}
  - uniform distribution
  - geometric distribution
  - 0/1 supplies and demands
- Built on existing network topologies from the SNDLib

SNDLIB: Overall computation

	pdh	newyork	france	ta1	norway	cost266	germany50	ta2
$\|V\|$	11	16	21	24	27	37	50	59
$\|E\|$	34	49	39	55	51	57	88	101

number of solved instances

geom	31/31	35/35	34/34	36/36	35/36	37/40	34/40	34/40
uni	36/36	37/38	39/40	38/39	37/40	36/40	28/40	33/40
0/1	40/40	40/40	40/40	40/40	40/40	40/40	39/40	40/40

SNDLIB: Overall computation

	pdh	newyork	france	ta1	norway	cost266	germany50	ta2
$\|V\|$	11	16	21	24	27	37	50	59
$\|E\|$	34	49	39	55	51	57	88	101

number of solved instances

geom	31/31	35/35	34/34	36/36	35/36	37/40	34/40	34/40
uni	36/36	37/38	39/40	38/39	37/40	36/40	28/40	33/40
0/1	40/40	40/40	40/40	40/40	40/40	40/40	39/40	40/40

average cpu time in seconds, solved instances only

geom	7	2	7	23	355	52	174	35
uni	1	53	50	89	307	217	402	148
0/1	2	0	0	0	0	8	15	1

CPU time for separation

	pdh	newyork	france	ta1	norway	cost266	germany50	ta2
$\lvert V\rvert$	11	16	21	24	27	37	50	59
$\lvert E\rvert$	34	49	39	55	51	57	88	101

average cpu time in seconds, solved instances only

geom	7	2	7	23	355	52	174	35
uni	1	53	50	89	307	217	402	148
0/1	2	0	0	0	0	8	15	1

average time spent in separation (in %)

geo	71.9	78.2	77.8	71.2	59.0	82.7	78.9	66.4
uni	87.5	67.3	70.2	70.1	59.2	82.2	81.3	79.3

Conclusions & Outlook

What does it mean?

- Tractable model for basic network design problem
- Changing the routing scheme is not the only way to easier separation

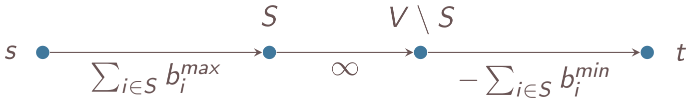
Conclusions & Outlook

What does it mean?

- Tractable model for basic network design problem
- Changing the routing scheme is not the only way to easier separation

How to continue?

- Meaningful extension to Γ-robustness model
- Probably won't work in multi-commodity case



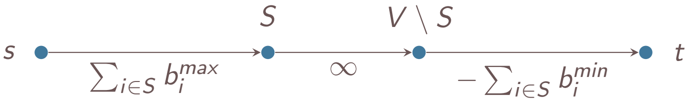
Conclusions & Outlook

What does it mean?

- Tractable model for basic network design problem
- Changing the routing scheme is not the only way to easier separation

How to continue?

- Meaningful extension to Γ-robustness model
- Probably won't work in multi-commodity case



— The End —