Decomposition Methods for Network Design

Bernard Gendron*

May 12, 2010

Spring School on Supply Chain and Transportation Network Design

* CIRRELT and Département d’informatique et de recherche opérationnelle, Université de Montréal, Canada
Outline

Introduction to network design

Multicommodity capacitated fixed-charge network design
  Lagrangian relaxation
  Cutting-plane method

Structured Dantzig-Wolfe decomposition for network design
  Reformulations and polyhedral results
  Stabilized structured Dantzig-Wolfe decomposition
  Computational results

Conclusions
Network design

- Network with multiple commodities
- Each commodity flows between supply and demand points
- Minimization of a “complex” (non-convex) objective function
  - Tradeoff between transportation and investment costs
  - Transportation costs: not necessarily linear, can be piecewise linear
  - Investment costs: “fixed” cost for building, renting, operating “facilities” at nodes or arcs of the network
- Additional constraints: budget, capacity, topology, reliability,...
- Variants:
  - Centralized / Decentralized
  - Static / Dynamic
  - Determinist / Stochastic
  - Strategic / Tactical / Operational
Infrastructure network design: strategic planning

- Planning horizon: years
- Decisions: invest in building roads, warehouses, plants,...
- Typical assumptions:
  - Central control
  - Static network
  - Linear transportation costs
  - Fixed costs for investment decisions
  - Usually no capacities
  - Known demands based on average values
- Robustness is an issue: stochastic demands?
Service network design: tactical planning

- Planning horizon: months
- Decisions: establish or not “services” (vehicles moving between two points) + flows-inventories
- Dynamic network: space-time expansion
  - Node = location-period
  - Transportation arc = (location1-period1, location2-period2) = moving from location1 to location2 in time (period2-period1)
  - Inventory arc = (location-period, location-period+1) = holding inventory at location between two consecutive periods
- Typical assumptions:
  - Central control
  - Linear inventory-transportation costs
  - Fixed costs for service decisions
  - Service capacities
  - Known demands
Adaptive network design: operational planning

- Planning horizon: days
- Decisions: operate or not “facilities” (warehousing or parking space) for fast product delivery + how many vehicles to use on each arc
- Typical assumptions:
  - Central control
  - Dynamic network
  - Piecewise linear transportation costs
  - Fixed costs for facility decisions
  - Facility and vehicle capacities
  - Known demands
Directed network $G = (N, A)$, with node set $N$ and arc set $A$

Commodity set $K$: known demand $d^k$ between origin $O(k)$ and destination $D(k)$ for each $k \in K$

Unit transportation cost $c_{ij}$ on each arc $(i, j)$

Capacity $u_{ij}$ on each arc $(i, j)$

Cost $f_{ij}$ for each capacity unit installed on arc $(i, j)$
Problem formulation

\[ Z = \min \sum_{(i,j) \in A} \sum_{k \in K} c_{ij} d^k x^k_{ij} + \sum_{(i,j) \in A} f_{ij} y_{ij} \]

\[ \sum_{j \in N_i^+} x^k_{ij} - \sum_{j \in N_i^-} x^k_{ji} = \begin{cases} 1, & i = O(k) \\ 1, & i = D(k) \\ 0, & i \neq O(k), D(k) \end{cases} \quad i \in N, \ k \in K \]

\[ \sum_{k \in K} d^k x^k_{ij} \leq u_{ij} y_{ij} \quad (i,j) \in A \]

\[ 0 \leq x^k_{ij} \leq 1 \quad (i,j) \in A, \ k \in K \]

\[ y_{ij} \text{ integer} \quad (i,j) \in A \]
Extensions

- Fixed-charge: $0 \leq y_{ij} \leq 1 \quad (i, j) \in A$
- Asset-balance constraints: $\sum_{j \in N_i^+} y_{ij} - \sum_{j \in N_i^-} y_{ji} = 0 \quad i \in N$
- Non-bifurcated flows: $x_{ij}^k \text{ integer} \quad (i, j) \in A, \ k \in K$
- Piecewise linear arc flow costs
- Multifacility design: several facilities $t \in T_{ij}$ on each arc, each with capacity $u_{ij}^t$ and cost $f_{ij}^t$
Multicommodity capacitated fixed-charge network design

- Directed network $G = (N, A)$, with node set $N$ and arc set $A$
- Commodity set $K$: known demand $d^k$ between origin $O(k)$ and destination $D(k)$ for each $k \in K$
- Unit transportation cost $c_{ij}$ on each arc $(i, j)$
- Capacity $u_{ij}$ on each arc $(i, j)$
- Fixed charge $f_{ij}$ incurred whenever arc $(i, j)$ is used to transport some commodity units
Problem formulation (MCND)

\[
Z = \min \sum_{(i,j) \in A} \sum_{k \in K} c_{ij} x_{ij}^k + \sum_{(i,j) \in A} f_{ij} y_{ij}
\]

\[
\sum_{j \in N_i^+} x_{ij}^k - \sum_{j \in N_i^-} x_{ji}^k = \begin{cases} 
  -d^k, & i = O(k) \\
  d^k, & i = D(k) \\
  0, & i \neq O(k), D(k) 
\end{cases} \quad i \in N, \ k \in K
\]

\[
\sum_{k \in K} x_{ij}^k \leq u_{ij} y_{ij} \quad (i,j) \in A
\]

\[
x_{ij}^k \geq 0 \quad (i,j) \in A, \ k \in K
\]

\[
y_{ij} \in \{0, 1\} \quad (i,j) \in A
\]
Developing solution methods for MCND: why?

- Generic problem: methods can be adapted to many similar network design applications
- But why “develop solution methods”: simply use a black-box solver!
- Things are not so simple:
  - LP relaxations are weak (typically, more than 20% gap w.r.t. optimal value)
  - LP relaxations can be hard to solve when the number of commodities is large: degeneracy
  - Combinatorial explosion
  - Dominant factors in increasing the complexity of a problem: high fixed charges + tight capacities + large number of commodities
- Two main classes of methods:
  - Mathematical programming
  - Metaheuristics
Overview of solution methods for MCND

- **Mathematical programming**
  - Cutting-plane methods: Chouman, Crainic, Gendron 2009
  - Benders decomposition: Costa, Cordeau, Gendron 2009

- **Metaheuristics**
  - Tabu search: Crainic, Farvolden, Gendreau 2000; Crainic, Gendreau 2002; Crainic, Gendreau, Ghamlouche 2003, 2004

- **Hybrid algorithms**
  - Slope scaling with long-term memory: Crainic, Gendron, Hernu 2004
Strong formulation

\[ Z = \min \sum_{(i,j) \in A} \sum_{k \in K} c_{ij} x_{ij}^k + \sum_{(i,j) \in A} f_{ij} y_{ij} \]

\[ \sum_{j \in N_i^+} x_{ij}^k - \sum_{j \in N_i^-} x_{ji}^k = \begin{cases} 
- d^k, & i = O(k) \\
- d^k, & i = D(k) \\
0, & i \neq O(k), D(k) 
\end{cases} \quad i \in N, k \in K \quad (\pi_i^k) \]

\[ \sum_{k \in K} x_{ij}^k \leq u_{ij} y_{ij} \quad (i,j) \in A \quad (\alpha_{ij}) \]

\[ x_{ij}^k \leq b_{ij}^k y_{ij} \quad (i,j) \in A, k \in K \quad (\beta_{ij}^k) \]

\[ x_{ij}^k \geq 0 \quad (i,j) \in A, k \in K \]

\[ y_{ij} \in \{0, 1\} \quad (i,j) \in A \]
Shortest path relaxation

\[ Z(\alpha, \beta) = \min \sum_{(i,j) \in A} \sum_{k \in K} (c_{ij} + \alpha_{ij} + \beta_{ij}^k)x_{ij}^k \]

\[ + \sum_{(i,j) \in A} (f_{ij} - u_{ij}\alpha_{ij} - \sum_{k \in K} b_{ij}^k\beta_{ij}^k)y_{ij} \]

\[ \sum_{j \in N_i^+} x_{ij}^k - \sum_{j \in N_i^-} x_{ji}^k = \begin{cases} 
- d^k, & i = O(k) \\
- d^k, & i = D(k) \\
0, & i \neq O(k), D(k) \end{cases} \quad i \in N, \ k \in K \]

\[ y_{ij} \in \{0, 1\} \quad (i,j) \in A \]
Knapsack relaxation

\[ Z(\pi) = \min \sum_{(i,j) \in A} \sum_{k \in K} (c_{ij} + \pi_i^k - \pi_j^k) x_{ij}^k + \sum_{(i,j) \in A} f_{ij} y_{ij} + \sum_{k \in K} d_k^k (\pi_{D(k)}^k - \pi_{O(k)}^k) \]

\[ \sum_{k \in K} x_{ij}^k \leq u_{ij} y_{ij} \quad (i,j) \in A \]

\[ x_{ij}^k \leq b_{ij}^k y_{ij} \quad (i,j) \in A, \ k \in K \]

\[ x_{ij}^k \geq 0 \quad (i,j) \in A, \ k \in K \]

\[ y_{ij} \in \{0,1\} \quad (i,j) \in A \]
Theoretical and computational results

- Both Lagrangian relaxations provide the same lower bound as the strong LP relaxation
- Lower bound within 9% of optimality on average
- To find (near-)optimal Lagrangian multipliers, two classes of methods have been traditionally used:
  - Subgradient methods
  - Bundle methods
- Our computational results show that:
  - Bundle methods are much more robust
  - Bundle methods converge faster
  - Any of these two methods converge much faster than solving the strong LP relaxation with the simplex method
Cutting-plane method: motivations

- Starting with the weak LP relaxation, iteratively add violated valid inequalities:
  - To be more efficient: keep the problem size as small as possible
  - To be more effective: improve the lower bound
- But the black-box solver already does that, so why not simply use it?
- True, but we can be more efficient and more effective by exploiting the structure of MCND
- Five classes of valid inequalities:
  - Strong inequalities (SI)
  - Cover inequalities (CI)
  - Minimum cardinality inequalities (MCI)
  - Flow cover inequalities (FCI)
  - Flow pack inequalities (FPI)
Computational results

### Comparison with CPLEX

<table>
<thead>
<tr>
<th></th>
<th>CI gap</th>
<th>CI cpu</th>
<th>CI cuts</th>
<th>FCI gap</th>
<th>FCI cpu</th>
<th>FCI cuts</th>
<th>All gap</th>
<th>All cpu</th>
<th>All cuts</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPLEX</td>
<td>5.40%</td>
<td>0.6%</td>
<td>10</td>
<td>23.17%</td>
<td>70.9%</td>
<td>305</td>
<td>23.20%</td>
<td>72.5%</td>
<td>306</td>
</tr>
<tr>
<td>Cutting-Plane</td>
<td>8.54%</td>
<td>1.4%</td>
<td>27</td>
<td>26.25%</td>
<td>28.6%</td>
<td>766</td>
<td>27.98%</td>
<td>12.6%</td>
<td>1537</td>
</tr>
</tbody>
</table>
Computational results

- Comparison with CPLEX

<table>
<thead>
<tr>
<th></th>
<th>CI</th>
<th>FCI</th>
<th>All</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>gap</td>
<td>cpu</td>
<td>cuts</td>
</tr>
<tr>
<td>CPLEX</td>
<td>5.40%</td>
<td>0.6%</td>
<td>10</td>
</tr>
<tr>
<td>Cutting-Plane</td>
<td>8.54%</td>
<td>1.4%</td>
<td>27</td>
</tr>
</tbody>
</table>

- Comparison between valid inequalities

<table>
<thead>
<tr>
<th></th>
<th>None+</th>
<th>All-</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>gap</td>
<td>cpu</td>
</tr>
<tr>
<td>Ø</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>SI</td>
<td>26.53%</td>
<td>7.3%</td>
</tr>
<tr>
<td>CI</td>
<td>8.54%</td>
<td>1.4%</td>
</tr>
<tr>
<td>MCI</td>
<td>8.00%</td>
<td>1.4%</td>
</tr>
<tr>
<td>FCI</td>
<td>26.25%</td>
<td>28.6%</td>
</tr>
<tr>
<td>FPI</td>
<td>26.75%</td>
<td>32.9%</td>
</tr>
</tbody>
</table>
Branch-and-cut algorithm

- Apply the cutting-plane method at every node of the B&B tree
- Generate Benders feasibility cuts along the tree
- Add pre- and post-processing at every node to reduce the size of the solution space
- Apply a variant of strong branching
- The resulting B&C algorithm is:
  - Much better than CPLEX B&C using the weak LP relaxation
  - Much better than CPLEX B&C using the strong LP relaxation
  - Competitive with CPLEX B&C using the cutting-plane LP relaxation
- How does it compare with state-of-the-art Lagrangian-based B&B?
General integer formulation (I)

\[
\begin{align*}
\min & \quad \sum_{k \in K} \sum_{(i,j) \in A} d^k c_{ij} x_{ij}^k + \sum_{(i,j) \in A} f_{ij} y_{ij} \\
\sum_{j \in N} x_{ij}^k - \sum_{j \in N} x_{ji}^k & = \begin{cases} 
-1, & \text{if } i = O(k) \\
0, & \text{if } i = D(k) \\
1, & \text{if } i \neq O(k), D(k) 
\end{cases} \quad \forall \ i \in N, \ k \in K \\
\sum_{k \in K} d^k x_{ij}^k & \leq u_{ij} y_{ij} \quad \forall (i,j) \in A \\
0 & \leq x_{ij}^k \leq 1 \quad \forall (i,j) \in A, \ k \in K \\
y_{ij} & \geq 0 \quad \forall (i,j) \in A \\
y_{ij} & \text{ integer} \quad \forall (i,j) \in A
\end{align*}
\]
Lagrangian relaxation of flow conservation

\[
\begin{align*}
\min & \sum_{k \in K} \sum_{(i,j) \in A} (d^k c_{ij} - \pi_i^k + \pi_j^k) x_{ij}^k + \sum_{(i,j) \in A} f_{ij} y_{ij} + \sum_{k \in K} \pi_{O(k)}^k - \pi_{D(k)}^k \\
\sum_{k \in K} d^k x_{ij}^k & \leq u_{ij} y_{ij} \quad \forall (i,j) \in A \\
0 & \leq x_{ij}^k \leq 1 \quad \forall (i,j) \in A, \ k \in K \\
y_{ij} & \geq 0 \quad \forall (i,j) \in A \\
y_{ij} & \text{ integer} \quad \forall (i,j) \in A
\end{align*}
\]

- Lagrangian subproblem decomposes by arc
- Easy (≈ 2 continuous knapsack) but no integrality property
Residual capacity inequalities

- For any $P \subseteq K$, define $d^P = \sum_{k \in P} d^k$
- Then, for any $(i, j) \in A$, define

\[
\begin{align*}
a^P_{ij} &= \frac{d^P}{u_{ij}} \\
q^P_{ij} &= \left\lfloor a^P_{ij} \right\rfloor \\
r^P_{ij} &= a^P_{ij} - \left\lfloor a^P_{ij} \right\rfloor
\end{align*}
\]

- Residual capacity inequalities

\[
\sum_{k \in P} \{a^k_{ij}(1 - x^k_{ij})\} \geq r^P_{ij}(q^P_{ij} - y_{ij}) \quad \forall (i, j) \in A, \ P \subseteq K
\]

- Characterize the convex hull of solutions to the Lagrangian subproblem (Magnanti, Mirchandani, Vachani 1993)
- Separation can be performed in $O(|A||K|)$ (Atamtürk, Rajan 2002)
Multiple choice model

\[ y_{ij} \leq \left\lceil \frac{\sum_{k \in K} d^k}{u_{ij}} \right\rceil = T_{ij} \]

\[ S_{ij} = \{1, \ldots, T_{ij}\} \]

\[ y^s_{ij} = \begin{cases} 
1, & \text{if } y_{ij} = s \\
0, & \text{otherwise} 
\end{cases} \quad \forall s \in S_{ij} \]

\[ x^s_{ij} = \begin{cases} 
\sum_{k \in K} d^k x^k_{ij}, & \text{if } y_{ij} = s \\
0, & \text{otherwise} 
\end{cases} \quad \forall s \in S_{ij} \]
Binary formulation \((B)\)

\[
y_{ij} = \sum_{s \in S_{ij}} sy_{ij}^s \quad \forall (i, j) \in A
\]

\[
\sum_{k \in K} d^k x_{ij}^k = \sum_{s \in S_{ij}} x_{ij}^s \quad \forall (i, j) \in A
\]

\[(s - 1)u_{ij} y_{ij}^s \leq x_{ij}^s \leq su_{ij} y_{ij}^s \quad (i, j) \in A, s \in S_{ij}\]

\[
\sum_{s \in S_{ij}} y_{ij}^s \leq 1 \quad (i, j) \in A
\]

\[
y_{ij}^s \geq 0 \quad (i, j) \in A, s \in S_{ij}
\]

\[
y_{ij}^s \text{ integer} \quad (i, j) \in A, s \in S_{ij}
\]
Variable disaggregation and extended formulation ($B^+$)

- Extended auxiliary variables

$$x_{ks}^{ij} = \begin{cases} x_{ij}^k, & \text{if } y_{ij} = s \\ 0, & \text{otherwise} \end{cases} \quad \forall \ s \in S_{ij}$$

$$x_{ij}^k = \sum_{s \in S_{ij}} x_{ij}^{ks} \quad \forall (i, j) \in A, k \in K$$

$$x_{ij}^s = \sum_{k \in K} d^k x_{ij}^{ks} \quad \forall (i, j) \in A, s \in S_{ij}$$

- Extended linking inequalities

$$x_{ij}^{ks} \leq y_{ij}^s \quad \forall (i, j) \in A, k \in K, s \in S_{ij}$$
Polyhedral results: notation

- $F(M)$: feasible set for model $M$
- $\text{conv}(F(M))$: convex hull of $F(M)$
- $LP(M)$: LP relaxation for model $M$
- $LS(M)$: Lagrangian subproblem (relaxation of flow conservation constraints)
- $LD(M)$: Lagrangian dual for $LS(M)$
Polyhedral results

- $LD(I)$ and $LD(B^+)$ are equivalent
- $F(LP(LS(B^+))) = conv(F(LS(B^+)))$
  (Croxton, Gendron, Magnanti 2007)
- $LP(B^+)$ and $LD(B^+)$ are equivalent
- $LP(B^+)$ and $LD(I)$ are equivalent
- $I^+ = I + \text{residual capacity inequalities}$
- $LP(B^+)$ and $LP(I^+)$ are equivalent (Frangioni, Gendron 2009)
Reformulations and decomposition

- “Structured” MIP:

\[
(P) \quad \min_x \{ cx : Ax = b , \ x \in X \}
\]

where \((P_\alpha)\):

\[
Z(\alpha) = \min_x \{ cx + \alpha(b - Ax) : x \in X \}
\]

“significantly easier” than \((P)\)

- Lagrangian dual:

\[
(LD) \quad \max_\alpha \{ Z(\alpha) \} = \min_x \{ cx : Ax = b , \ x \in \text{conv}(X) \} (LP)
\]

- Reformulation:

\[
\text{conv}(X) = \{ x = C\theta : \Gamma \theta \leq \gamma \}
\]

- Examples:
  - Dantzig-Wolfe Reformulation \((DW)\)
  - Extended Formulation \((B^+)\)
Structured DW decomposition: assumptions

- **Assumption 1 (reformulation):**
  \[
  \text{conv}(X) = \{ x = C\theta : \Gamma \theta \leq \gamma \}
  \]

- **Assumption 2 (padding with zeroes):**
  \[
  \Gamma_B \theta_B \leq \gamma_B \implies \Gamma [\theta_B, 0] \leq \gamma
  \]
  \[
  \implies X_B = \{ x = C_B \theta_B : \Gamma_B \theta_B \leq \gamma_B \} \subseteq \text{conv}(X)
  \]

- **Assumption 3 (easy update of variables and constraints):**
  Given \( B, \bar{x} \in \text{conv}(X) \) s.t. \( \bar{x} \notin X_B \),
  it is “easy” to find \( B' \supseteq B \) and \( \Gamma_{B'}, \gamma_{B'} \) such that
  \( \exists B'' \supseteq B' \) such that \( \bar{x} \in X_{B''} \).
Structured DW decomposition: algorithm

\[
\langle \text{initialize } B \rangle; \\
\text{repeat} \\
\langle \text{solve } (LP_B) \text{ for } \tilde{x}, \tilde{\alpha} ; \tilde{v} = c\tilde{x} \rangle; \\
\tilde{x} = \text{argmin}_{x} \{ (c - \tilde{\alpha}A)x : x \in X \}; \quad /* (P_{\tilde{\alpha}}) */ \\
\text{if}( \tilde{v} = c\tilde{x} + \tilde{\alpha}(b - A\tilde{x}) ) \\
\quad \text{then STOP;} \quad /* \tilde{x} \text{ optimal */} \\
\quad \text{else } \langle \text{update } B \text{ as in Assumption 3 } \rangle; \\
\text{until } \sim \text{ STOP}
\]

- Finitely terminates with an optimal solution of \((LP)\)
- ... even if (proper) removal of indices from \(B\) is allowed
Structured DW and other decomposition methods

- Generalizes DW, whose unstructured model is identical for all applications (except when exploiting disaggregation)
- Substantially different from both RG (Row Generation) and DW
Stability issues in (structured)DW

- The next $\tilde{\alpha}$ can be very far from the current one
- In general, the sequence of $\tilde{\alpha}$ is unstable, has no locality properties and convergence speed does not improve near the optimum
- Counter-measure: use a Proximal Point method defined by a stabilizing term $D_t$, depending on the current $\tilde{\alpha}$ and proximal parameter(s) $t$
Some stabilizing terms

a penalty

\[ D_t = \frac{1}{2t} \| \cdot \|_2^2 \]
\[ D_t^* = \frac{1}{2} t \| \cdot \|_2^2 \]

a trust region

\[ D_t = l_{B_\infty}(t) \]
\[ D_t^* = t \| \cdot \|_1 \]

or both

\[ D_{\Gamma^\pm, \Delta^\pm, \varepsilon^\pm} = \ldots \]
\[ D_{\Gamma^*_{\Gamma^\pm, \Delta^\pm, \varepsilon^\pm}} = \ldots \]
Stabilized structured DW algorithm

- Exactly the same as stabilizing DW!
- Stabilized DW = Proximal Point + Column Generation (= Bundle, Frangioni 2002)
- Even simpler from the primal viewpoint:

\[
\min \{ cx - \tilde{\alpha}z + \mathcal{D}_t^*(-z) : z = Ax - b, \ x = C_B\theta_B, \ \Gamma_B\theta_B \leq \gamma_B \} 
\]

- With proper choice of \(\mathcal{D}_t^*\), this is still a linear program
- Dual optimal variables of “\(z = Ax - b\)” still give \(\tilde{\alpha}\)
- Convergence theory basically the same as in (Frangioni 2002)
Summary of Approaches

- $I^+$: Cutting-plane with exponential number of constraints, but easy separation
- $StabDW$: Bundle for $DW$ with exponential number of variables, but easy pricing
- $StructDW$: Structured DW for $LP(B^+)$ with pseudo-polynomial number of variables and constraints
- $S_2DW_2$: Stabilized Structured DW with quadratic penalty
- $S_2DW_1$: Stabilized Structured DW with trust region
- $S_2DW_1–ws^2$: Stabilized Structured DW with trust region and subgradient optimization warmstart
Computational experiments

- Large-scale instances ($|K| \in \{100, 200, 400\}$), very difficult
- $C = 1 \Rightarrow$ lightly capacitated, $C = 16 \Rightarrow$ tightly capacitated
- Solving the root relaxation, then freezing the formulation + CPLEX polishing for one hour
- Unlike $I^+$, frozen $B^+$ formulations may not contain optimal solution \( \Rightarrow \) final gap \( \approx \) quality of obtained formulation
- \( \text{imp} = \) lower bound improvement (equal for all)
  \( \text{gap} = \) final gap (\%), \( \text{cpu} = \) time, \( \text{it} = \) iterations
Sample computational results ($|K| = 100$)

| Problem | $|A|$ | $C$ | imp | $l_+$ | cpu | gap | it | StabDW | cpu | gap | it | StructDW | cpu | gap | it |
|---------|-----|-----|-----|-------|-----|-----|----|--------|-----|-----|----|----------|-----|-----|----|
| 517     | 1   | 187.00 | 348 | 5.78  | 26  | 4323 | 88144 | 296    | 6.94 | 55  |
|         | 4   | 138.22 | 362 | 6.42  | 25  | 3581 | 79390 | 312    | 7.48 | 44  |
|         | 8   | 100.08 | 305 | 6.12  | 21  | 4054 | 88807 | 633    | 6.11 | 61  |
|         | 16  | 60.49  | 249 | 6.20  | 21  | 3015 | 71651 | 1138   | 6.45 | 87  |
| 517     | 1   | 155.19 | 140 | 3.95  | 23  | 2899 | 69500 | 188    | 4.70 | 60  |
|         | 4   | 122.84 | 194 | 3.87  | 26  | 2799 | 65229 | 147    | 4.15 | 39  |
|         | 8   | 93.00  | 151 | 3.96  | 20  | 2824 | 66025 | 355    | 4.31 | 67  |
|         | 16  | 59.68  | 116 | 4.72  | 18  | 2172 | 56184 | 551    | 4.94 | 70  |
| 669     | 1   | 114.50 | 80  | 0.50  | 26  | 330  | 11273 | 36     | 0.46 | 32  |
|         | 4   | 97.32  | 78  | 0.46  | 22  | 327  | 10951 | 66     | 0.46 | 50  |
|         | 8   | 79.62  | 68  | 0.46  | 19  | 323  | 11173 | 55     | 0.46 | 33  |
|         | 16  | 56.19  | 58  | 0.74  | 19  | 275  | 9979  | 164    | 0.81 | 65  |
Sample computational results ($|K| = 200$)

| Problem | $|A|$ | $C$ | imp | $I^+$ | cpu | gap | it | StabDW | cpu | it | StructDW | cpu | gap | it |
|---------|------|-----|-----|-------|-----|------|---|--------|-----|---|-----------|-----|------|---|
|         |      |     |     |       |     |      |   |        |     |   |           |     |      |   |
| 229     | 1    | 205.67 | 49081 | 28.16 | 109 | 11748 | 154821 | 525 | 10.50 | 44 |
| 4       | 131.24 | 30899 | 25.40 | 91   | 9132 | 131674 | 807 | 13.58 | 45 |
| 8       | 84.61 | 16502 | 21.80 | 87   | 12682 | 162766 | 1593 | 10.17 | 44 |
| 16      | 42.78 | 2090 | 5.59  | 54   | 6541 | 97952 | 2630 | 9.20  | 73 |
| 229     | 1    | 185.17 | 18326 | 20.53 | 86  | 9261 | 132963 | 380 | 7.44  | 39 |
| 4       | 125.39 | 15537 | 18.81 | 80   | 11791 | 147879 | 612 | 9.36  | 49 |
| 8       | 85.31 | 9500 | 13.08 | 74   | 10702 | 146727 | 1647 | 8.87  | 68 |
| 16      | 46.09 | 1900 | 7.19  | 52   | 7268 | 107197 | 3167 | 7.99  | 108 |
| 287     | 1    | 198.87 | 14559 | 27.86 | 66  | 8815 | 120614 | 598 | 12.54 | 53 |
| 4       | 136.97 | 11934 | 22.52 | 62   | 8426 | 112308 | 603 | 15.07 | 37 |
| 8       | 92.94 | 9656 | 15.28 | 64   | 10098 | 130536 | 1221 | 10.38 | 41 |
| 16      | 53.45 | 3579 | 11.60 | 54   | 6801 | 98972 | 3515 | 9.06  | 99 |
Sample computational results ($|K| = 400$)

<table>
<thead>
<tr>
<th>Problem</th>
<th>StabDW</th>
<th>StructDW</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>A</td>
<td>$</td>
</tr>
<tr>
<td>519</td>
<td>1</td>
<td>100.83</td>
</tr>
<tr>
<td>4</td>
<td>92.54</td>
<td>88031</td>
</tr>
<tr>
<td>8</td>
<td>82.16</td>
<td>88918</td>
</tr>
<tr>
<td>16</td>
<td>65.53</td>
<td>85384</td>
</tr>
<tr>
<td>519</td>
<td>1</td>
<td>125.07</td>
</tr>
<tr>
<td>4</td>
<td>111.02</td>
<td>90573</td>
</tr>
<tr>
<td>8</td>
<td>94.82</td>
<td>93418</td>
</tr>
<tr>
<td>16</td>
<td>71.31</td>
<td>93567</td>
</tr>
<tr>
<td>668</td>
<td>1</td>
<td>126.02</td>
</tr>
<tr>
<td>4</td>
<td>115.29</td>
<td>99014</td>
</tr>
<tr>
<td>8</td>
<td>102.03</td>
<td>104481</td>
</tr>
<tr>
<td>16</td>
<td>80.96</td>
<td>103011</td>
</tr>
</tbody>
</table>
Some preliminary conclusions

- DW unbearably slow, and disaggregating does not help enough
- Stabilized DW ≡ bundle much better, but only aggregated
- SDW worsens as $C$ grows (tighter capacities), RG the converse
- SDW generally better, but times and gaps are still large ⇒ Stabilized SDW seems promising
Computational experiments on stabilized SDW

- No removal/aggregation for $B$, fixed $t$ (class-specific tuning)
- Different stabilizing terms: quadratic penalty vs trust region (QP vs LP)
- Different warm-start: “standard” MCF initialization (used for all) vs MCF + subgradient warm-start (few iterations, class-specific tuning)
- gap = final gap (%), cpu = time, it = iterations, ss = serious steps
Sample computational results ($|K| = 100$)

<table>
<thead>
<tr>
<th></th>
<th>StructDW</th>
<th>$S^2DW_2$</th>
<th>$S^2DW_1$</th>
<th>$S^2DW_1\text{–}ws^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>cpu</td>
<td>gap</td>
<td>it</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>296</td>
<td>6.94</td>
<td>55</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>312</td>
<td>7.48</td>
<td>44</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>633</td>
<td>6.11</td>
<td>61</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>1138</td>
<td>6.45</td>
<td>87</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>188</td>
<td>4.70</td>
<td>60</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>147</td>
<td>4.15</td>
<td>39</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>354</td>
<td>4.31</td>
<td>67</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>551</td>
<td>4.94</td>
<td>70</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>36</td>
<td>0.46</td>
<td>32</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>66</td>
<td>0.46</td>
<td>50</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>55</td>
<td>0.46</td>
<td>33</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>164</td>
<td>0.81</td>
<td>65</td>
<td></td>
</tr>
</tbody>
</table>

Note: The table shows computational results for different datasets with varying sizes. The columns represent the CPU time, gap, number of iterations (it), and success (ss) for each dataset. The rows correspond to different problem sizes (1, 4, 8, 16). The colors in the table indicate the performance of different algorithms: red for better performance and blue for worse performance.
Sample computational results ($|K| = 200$)

<table>
<thead>
<tr>
<th>C</th>
<th>StructDW</th>
<th>$S^2DW_2$</th>
<th>$S^2DW_1$</th>
<th>$S^2DW_1−ws^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>cpu</td>
<td>gap</td>
<td>it</td>
<td>cpu</td>
</tr>
<tr>
<td>1</td>
<td>525</td>
<td>10.50</td>
<td>44</td>
<td>1.8e4</td>
</tr>
<tr>
<td>4</td>
<td>807</td>
<td>13.58</td>
<td>45</td>
<td>2.7e4</td>
</tr>
<tr>
<td>8</td>
<td>1593</td>
<td>10.17</td>
<td>44</td>
<td>8.3e4</td>
</tr>
<tr>
<td>16</td>
<td>2630</td>
<td>9.20</td>
<td>73</td>
<td>1.1e5</td>
</tr>
<tr>
<td>1</td>
<td>380</td>
<td>7.44</td>
<td>39</td>
<td>1.0e4</td>
</tr>
<tr>
<td>4</td>
<td>612</td>
<td>9.36</td>
<td>49</td>
<td>1.3e4</td>
</tr>
<tr>
<td>8</td>
<td>1647</td>
<td>8.87</td>
<td>68</td>
<td>3.3e4</td>
</tr>
<tr>
<td>16</td>
<td>3167</td>
<td>7.99</td>
<td>108</td>
<td>7.0e4</td>
</tr>
<tr>
<td>1</td>
<td>598</td>
<td>12.54</td>
<td>53</td>
<td>2.1e4</td>
</tr>
<tr>
<td>4</td>
<td>603</td>
<td>15.07</td>
<td>37</td>
<td>1.8e4</td>
</tr>
<tr>
<td>8</td>
<td>1221</td>
<td>10.38</td>
<td>41</td>
<td>5.2e4</td>
</tr>
<tr>
<td>16</td>
<td>3515</td>
<td>9.06</td>
<td>99</td>
<td>1.3e5</td>
</tr>
</tbody>
</table>
Sample computational results ($|K| = 400$)

<table>
<thead>
<tr>
<th>$C$</th>
<th>StructDW</th>
<th>$S^2DW_1$</th>
<th>$S^2DW_1$−ws$^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>cpu gap it</td>
<td>cpu gap it ss</td>
</tr>
<tr>
<td>1</td>
<td>9839 9.96</td>
<td>157 2473 2.23 76 55</td>
<td>1857 2.31 53 38</td>
</tr>
<tr>
<td>4</td>
<td>9087 11.25</td>
<td>140 2140 2.33 68 54</td>
<td>2140 2.36 66 44</td>
</tr>
<tr>
<td>8</td>
<td>11613 8.47</td>
<td>143 2338 2.45 66 45</td>
<td>1813 2.30 52 30</td>
</tr>
<tr>
<td>16</td>
<td>38617 10.26</td>
<td>242 3403 2.66 77 39</td>
<td>2570 2.26 58 23</td>
</tr>
<tr>
<td>1</td>
<td>22246 14.90</td>
<td>165 4811 3.31 87 76</td>
<td>4668 3.06 66 55</td>
</tr>
<tr>
<td>4</td>
<td>17976 18.22</td>
<td>131 4324 2.57 77 64</td>
<td>4373 3.19 66 45</td>
</tr>
<tr>
<td>8</td>
<td>30460 18.18</td>
<td>159 5224 3.14 85 60</td>
<td>4209 2.86 57 36</td>
</tr>
<tr>
<td>16</td>
<td>74447 16.50</td>
<td>176 5532 3.14 67 46</td>
<td>5191 3.02 64 23</td>
</tr>
<tr>
<td>1</td>
<td>23771 11.89</td>
<td>149 9215 2.96 97 78</td>
<td>6815 3.01 69 56</td>
</tr>
<tr>
<td>4</td>
<td>28567 10.97</td>
<td>176 6766 2.99 79 63</td>
<td>6506 3.07 69 45</td>
</tr>
<tr>
<td>8</td>
<td>27871 12.07</td>
<td>130 7560 2.67 87 56</td>
<td>5765 2.78 61 37</td>
</tr>
<tr>
<td>16</td>
<td>58363 13.95</td>
<td>156 8626 3.14 83 45</td>
<td>3764 2.95 41 18</td>
</tr>
</tbody>
</table>
Current research and future trends

- Adaptive network design (Gendron, Semet 2009)
- Integrating uncertainty: stochastic programming (Crainic, Gendreau, Rei, Wallace 2009)
- Decentralized / collaborative network design
- On the methodological side:
  - Alternative formulations based on paths/circuits and (multi)-cutsets
  - Decomposition methods involving column and cut generation within B&B: B&C&P
  - Hybrid algorithms combining mathematical programming and metaheuristics
  - Parallel computing (especially for large-scale adaptive and stochastic network design)