Network Design Models for Distribution Systems

Anant Balakrishnan

March 2003

Outline

- Motivation
- Problem definition
- Network Design model
- Variants and special cases
- Overview of solution methods
  - Dual Ascent approach
  - Computational results
  - Importance of good model formulation
- Extensions

Based on joint work with Thomas Magnanti, Prakash Mirchandani, Richard Wong, and others
Distribution System Design example
Armstrong World Industries

- Market leader in flooring and ceiling products
- > $3 Billion sales in 2001
- Product-focused organization structure
  - **Flooring** products division versus **Ceiling** division
  - distinct manufacturing, sales, distribution organizations
- Traditional distribution system
  - independent flooring and ceiling distributors
  - responsible for sales, pricing, delivery, credit
- Big-Box retailers are now biggest customers

Big-Box Retailers

- Growing rapidly, increasing market power
  - **Home Depot**: $54 billion annual sales, 1400+ US stores and growing
- Formula for success: *wide variety at competitive prices under one roof*
  - negotiate low prices from manufacturers
  - maintain low store inventory, high variety
  - ensure high availability
- “Demanding” customers
  - need a single point of contact
  - frequent (weekly) deliveries of multiple items in small batches
  - 24-hour delivery lead time
Meeting Big-Box Retailers’ needs

- Set up **new** distribution network
  - Establish **Regional Distribution Centers (RDC)** for warehousing, distribution
  - Specify coverage region (assign stores) for each RDC
  - RDCs receive bulk shipments from factories
  - RDCs deliver orders to stores (small trucks) at scheduled times (weekly)

- **+ Organizational changes**

Distribution System Design issues

- **Decisions**
  - How many distribution centers?
  - Where?
  - Which customers (stores) to assign to each DC?
  - Which plants to supply each DC?

- **Problem scope**
  - 19 Flooring plants, 7 Ceiling plants in U.S. alone
  - Hundreds of SKUs
  - Over 2000+ customer locations (stores) nationwide
Distribution System Design problem

Cost tradeoffs

- **Cost components**
  - DC costs: Fixed investment + operating costs (fixed and/or throughput-dependent)
  - Inbound (plant-to-DC) transportation costs: including economies of scale
  - Outbound (DC-to-store) delivery cost

- **Basic tradeoff**
  - If we open more DCs ⇒ we can locate them closer to customers, but …
    - Higher total DC fixed costs
    - Higher total inbound transportation costs
    - Lower outbound delivery cost
Distribution System Design solution

Strategic Distribution System Design applications

- Yellow Freight – Powell et al. (1992)
- Digital Equipment Corp. - Brown et al. (1995)
- Procter & Gamble – Camm et al. (1997)
- UPS – Barnhart et al. (1999, 2002)
- Railroad Blocking – Barnhart et al. (2000)
- DHL Hong Kong – Cheung et al. (2001)
**Practice of Supply Chain Design**

- Models vary depending on application context
- Large problem sizes (‘00s of nodes, ‘000s of O-D pairs)
- Need specialized solution methodologies to exploit problem structure
- Reported savings of tens to hundreds of millions of dollars
- Optimal design capability supplemented with:
  - Detailed simulations to test optimal designs
  - What-if, sensitivity, and scenario analyses
- Commercial SC and ERP systems now provide APS modules with SC design optimization capability

---

**Network Design Problem definition**

- Origin, Destination nodes
- Transshipment points
- Commodity \( k \)
  - Origin \( O_k \)
  - Destination \( D_k \)
- Demand \( b_k \)
- Fixed cost \( F_{ij} \)
- Variable (flow) cost \( c_{ij} \)

Decisions: Select edges, route flows
Network Design Problem definition

- **Given**
  - **NODES**
    - Origins (plants), transshipment points (DC), destinations (stores)
  - **COMMODITIES**
    - Origin and destination
    - Demand
  - **EDGES**
    - Fixed cost
    - Variable (flow) cost

- **Required**
  - Select the edges to use
  - Route required flows

- **Objective**
  - Minimize total FIXED cost of design + FLOW costs

---

**Notation**

**Parameters**
- \( G(N,E) \) Given graph (directed or undirected)
- \( i,j \in N \) Nodes
- \( (i,j) \in N \) Edges (directed) or arcs (directed)
- \( A \) Node-arc incident matrix
- \( k \in K \) Commodities
- \( B_k \) Demand vector for commodity \( k \)
- \( F_{ij} \) Fixed cost of edge \( (i,j) \)
- \( c_{ij}^k \) Variable cost of comm. \( k \) from \( i \) to \( j \) on edge \( (i,j) \)

**Variables**
- \( f_{ij}^k \) Units of flow of comm. \( k \) from \( i \) to \( j \) on edge \( (i,j) \)
- \( z_{ij} \) Design variable;
  - \( = 1 \) if solution includes edge \( (i,j) \), 0 otherwise
**Arc flow formulation**

Minimize  \( Fz + \sum_{k \in K} c^k f^k \)

Flow conservation constraints  \( Af^k = B_k \quad \forall k \in K \)

Forcing constraints  \( f^k \leq u^k z \quad \forall k \in K \)

Nonnegativity, integrality  \( f^k \geq 0 \quad \forall k \in K, z \in \{0,1\}^m \)

Routing requirements (optional)  \( f^k \in P^k \quad \forall k \in K \)

Design restrictions (optional)  \( z \in Z(f) \)

---

**Classification of Network Design problems**

- **Demand** structure
  - Complete demand, single source, single destination, ..
  - With or without Steiner nodes

- **Network** structure
  - Directed vs. undirected
  - Tree, layered, general
  - With or without edge duplication
  - Multiple node types, facility types

- **Cost** structure
  - Only fixed costs, fixed + variable costs, step function, general concave costs

- **Design** and **Routing** restrictions
Types of additional constraints

- **Routing requirements**
  - Weight (e.g., delay) or hop constraints
  - Bifurcated vs. non-bifurcated flows
  - Alternate (edge-disjoint) paths
  - Commodity-dependent facility requirements

- **Design restrictions**
  - Capacity constraints—commodity-specific, bundled
  - Simultaneous vs. non-simultaneous usage
  - Degree, diameter constraints
  - Precedence, multiple choice constraints
  - Topological restrictions: tree, ring, ...

Other application contexts for Network Design models

- Supply chain design
- Less-than-truckload consolidation, hub location
- Public utilities (water, electricity, gas, waste) distribution planning
- Telecomm network design
  - Long-distance, local access, ring, wireless
- VLSI, circuit design
- Distributed database location
- Production planning, process design (e.g., chemical)
- Airline operations planning (e.g., fleet assignment)
- Railroad blocking
- Marketing models: Product positioning
- Biology, genetics?
Research perspectives

- Theory
  - New models
    - MLND, WCND, Cap. tree covering
  - Worst-case analysis
  - Polyhedral results
  - Decomposition methods
  - Problem reduction
  - Dual-ascent
  - Approximation/heuristics

- Applications
  - LTL consolidation
  - Distribution sys design
  - Local access network
  - Restoration planning

- Algorithms

Basic Network Design model

- No additional routing requirements or design restrictions
- Each commodity $k$ has single origin $O_k$ and single destination $D_k$
  - Scale all demands to 1 unit
  - Define $x_{ij}^k$ as fraction of commodity $k$'s demand flowing from node $i$ to node $j$
  - Define $c_{ij}^k$ as cost of routing all units of commodity $k$ from node $i$ to node $j$
- Assume directed graph
Arc flow formulation for Basic model

Minimize \[ \sum_{(i,j) \in E} F_{ij} z_{ij} + \sum_{k \in K} \sum_{(i,j)} c^k_{ij} x^k_{ij} \]

Flow conservation
\[ \sum_{j} x^k_{ij} - \sum_{j} x^k_{ji} = \begin{cases} +1 & \text{if } i = O_k \\ -1 & \text{if } i = D_k \\ 0 & \text{otherwise} \end{cases} \quad \forall k \in K \]

Forcing constraints
\[ x^k_{ij} \leq z_{ij} \quad \forall (i, j) \in E, k \in K \]

Nonnegativity
\[ x^k_{ij} \geq 0 \quad \forall (i, j) \in E, k \in K \]

Integrality
\[ z_{ij} \in \{0, 1\} \quad \forall (i, j) \in E \]

Some Model Extensions

- Node costs
- Piecewise-concave costs (economies of scale)
- Demand selection
- Uncertainty (limited versions)
Economies of scale (piecewise linear, concave costs)

\[ F_{ij}^1, F_{ij}^2, F_{ij}^3, c_{ij} \]

Demand selection

\[ \pi_k \] 
Commodity \( k \)  
Origin \( O_k \)  
Destination \( D_k \)  
Demand \( b_k \)  

Fixed cost \( \pi_k \) 
Variable) cost = 0
### DC-to-Store Delivery Fee Table

<table>
<thead>
<tr>
<th>Delivery Weight (pounds)</th>
<th>0-1000 lbs</th>
<th>1000-4000</th>
<th>4000-10000</th>
<th>&gt; 10,000 lbs</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-30 miles</td>
<td>$40</td>
<td>$70</td>
<td>$100</td>
<td>$140</td>
</tr>
<tr>
<td>30-70 miles</td>
<td>$60</td>
<td>$90</td>
<td>$120</td>
<td>$160</td>
</tr>
<tr>
<td>&gt; 70 miles</td>
<td>$70</td>
<td>$100</td>
<td>$140</td>
<td>$200</td>
</tr>
</tbody>
</table>

### Special Cases

- **Shortest Path problem**
  - Only one commodity
  - Fixed cost = arc length; variable cost = 0

- **Minimum Spanning Tree problem**
  - Single source (root node)
  - One commodity each to all other nodes
  - Fixed cost = arc length; variable cost = 0

- **Steiner Tree problem**
  - Same as MST except some nodes are transshipment nodes (no corresponding commodity)
Special cases (continued)

- Facility Location problem
- Dynamic Lot-sizing problem
- Traveling Salesman problem

Facility Location Problem

![Diagram of Facility Location Problem]

Bipartite network, single source

Transportation cost

Plant Fixed cost

Source

Plants

Customers
Problem complexity

- Model formulation size grows very rapidly with problem size
  - Problem with 100 nodes, 2000 arcs (sparse), 2000 commodities has 4 million variables

- Problem is NP-hard since it generalizes several known difficult problems

Solution approaches

- Exact methods
  - Decomposition (e.g., Lagrangian relaxation, Bender’s decomposition, column generation)
  - Polyhedral methods (branch-and-cut)

- Approximate methods
  - Optimization-based (e.g., dual ascent, primal-dual)
  - Solving restricted (easy) versions
  - Local search & improvement methods
Dual Ascent principle

- Approximately solve dual of LP relaxation
  - By iteratively adjusting the dual multipliers
  - Exploit special problem structure

- Dual solution provides
  - Lower bound on optimal value (performance guarantee)
  - Starting feasible solution for heuristic improvement procedure
  - Problem reduction opportunity

Dual Ascent success stories

- Assignment problem (Fisher)
- Steiner Tree problem (Wong)
- Uncapacitated Network Design (BMW)
- Multi-level Network Design (BMM)
- Survivable Network Design (Raghavan)
- Other optimization problems
LP relaxation of Basic model

Minimize \( \sum_{(i,j) \in E} F_{ij} z_{ij} + \sum_{k \in K} \sum_{(i,j) \in E} c_{ij}^k x_{ij}^k \)

Dual variables

Flow conservation \( \sum_j x_{ij}^k - \sum_j x_{ji}^k = \begin{cases} +1 & \text{if } i = O_k \\ -1 & \text{if } i = D_k \\ 0 & \text{otherwise} \end{cases} \forall i \in N, k \in K \quad v_i^k \)

Forcing constraints \( x_{ij}^k \leq z_{ij} \quad \forall (i,j) \in E, k \in K \quad w_{ij}^k \)

Nonnegativity \( x_{ij}^k, z_{ij} \geq 0 \quad \forall (i,j) \in E, k \in K \)

Integrality \( z_{ij} \in \{0, 1\} \quad \forall (i,j) \in E \)

Dual problem and its special structure

Define \( d_{ij}^k = c_{ij}^k + w_{ij}^k \)

Maximize \( \sum_{k \in K} v_{D_k}^k \)

Node potentials \( v_j^k, v_i^k, v_j^k d_{ij}^k w_{ij}^k \quad \forall (i,j) \in E, k \in K \)

Fixed cost allocation \( \sum_{k \in K} w_{ij}^k \leq F_{ij} \quad \forall (i,j) \in E \)

Non-negativity \( v_i^k, w_{ij}^k \geq 0 \quad \forall (i,j) \in E, k \in K \)

Given values of \( w_{ij}^k \) satisfying the FC allocation constraint, for each commodity \( k \), \( v_{D_k}^k \triangleq L_k \) is length of shortest path from \( O_k \) to \( D_k \) using \( d_{ij}^k = c_{ij}^k + w_{ij}^k \) as arc lengths
Dual Ascent idea

- Starting with $w = 0$
  - For each commodity $k$, selectively increase certain $w_{ij}^k$ values—by allocating the arc fixed cost $F_{ij}$—to increase the value of the shortest path length $L_k$
  - Stop when no more increases possible.

Definitions

- $R_{ij} = F_{ij} - \sum_k w_{ij}^k$ = Remaining fixed cost on edge $(i, j)$
- Edge $(i, j)$ is **tight** for comm. $k$ if it lies on the shortest O-D path for comm. $k$
- $S_{ij}^k = (v_j^k - v_i^k) - d_{ij}^k$ = Slack on edge $(i, j)$ for comm. $k$

Labeling procedure

- Similar to Dijkstra’s shortest path algorithm
- For each commodity:
  - Find current shortest paths (tight arcs)
  - Define a cutset defined by labeled nodes (incl. destination)
  - Increase shortest path length by minimum of
    - Remaining fixed cost among tight arcs in cutset
    - Slack among loose arcs in cutset
  - Update dual values, labels
    - Allocate fixed costs for tight arcs,
    - increase dual objective function value, and
    - label additional nodes (if remaining fixed cost is binding)
- Stop when origin node is labeled for all commodities
Dual Ascent example

Variable cost $c_{ij} = 1$ for all edges, all commodities

Final dual solution

Final dual value = 14
Properties of Dual Ascent procedure

- Generalizes
  - Edmond's directed spanning tree algorithm
  - Erlenkotter's DUALOC facility location procedure
  - Wong's Steiner Tree algorithm

- Pseudo-polynomial

- Property of final dual solution
  - At termination, network consisting of “Fully allocated arcs” is feasible for the original problem, i.e., this design is guaranteed to contain at least one O-D path for every commodity

Dual Ascent outputs

- Lower bound
  - Final dual value is a lower bound on optimal value of network design problem

- Heuristic solution
  - Using feasible design from dual ascent procedure as a starting solution, apply local improvement procedure (e.g., Add/Drop or interchange heuristic)

- Problem reduction
  - Eliminate edges based on dual solution
Problem reduction

- Let
  - $Z_D = \text{Final dual objective value}$
  - $Z^H = \text{Cost of heuristic solution}$
  - $R_{ij} = \text{Remaining fixed cost of edge } (i, j)$

- If $R_{ij} > (Z^H - Z_D)$, then edge $(i, j)$ cannot belong to any optimal solution
  - eliminate edge $(i, j)$, and re-apply dual ascent

Dual Ascent enhancements

- Use complementary-slackness conditions from final dual solution to fix $w$-values, and re-apply dual ascent procedure to possibly improve lower bound

- Test alternate commodity sequencing schemes

- Add methods to reallocate $w$-values (versus only increasing these values)

- Modify method to incorporate additional constraints
Test problems

Network Dimensions for Undirected Test Problems

<table>
<thead>
<tr>
<th>Problem Number</th>
<th>No. of Nodes</th>
<th>No. of Arteries</th>
<th>No. of Commodities</th>
<th>No. of Variables in Formulation P1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Integer</td>
</tr>
<tr>
<td>1</td>
<td>20</td>
<td>80</td>
<td>380</td>
<td>80</td>
</tr>
<tr>
<td>2</td>
<td>25</td>
<td>100</td>
<td>600</td>
<td>100</td>
</tr>
<tr>
<td>3</td>
<td>30</td>
<td>130</td>
<td>870</td>
<td>130</td>
</tr>
<tr>
<td>4</td>
<td>33</td>
<td>150</td>
<td>150</td>
<td>150</td>
</tr>
<tr>
<td>5</td>
<td>40</td>
<td>400</td>
<td>156</td>
<td>400</td>
</tr>
<tr>
<td>6</td>
<td>43</td>
<td>500</td>
<td>95</td>
<td>900</td>
</tr>
<tr>
<td>Complete Networks</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>15</td>
<td>195</td>
<td>210</td>
<td>105</td>
</tr>
<tr>
<td>8</td>
<td>20</td>
<td>190</td>
<td>300</td>
<td>100</td>
</tr>
<tr>
<td>9</td>
<td>25</td>
<td>500</td>
<td>600</td>
<td>300</td>
</tr>
<tr>
<td>10</td>
<td>30</td>
<td>435</td>
<td>870</td>
<td>435</td>
</tr>
<tr>
<td>11</td>
<td>35</td>
<td>595</td>
<td>1150</td>
<td>595</td>
</tr>
</tbody>
</table>

Computational results

Dual-Ascent Performance for Asymmetric Test Problems (all test problems have FC/VC ratio = 10,0)

<table>
<thead>
<tr>
<th>Problem Size</th>
<th>CPU Time</th>
<th>No. of Total Ascent</th>
<th>% Gap</th>
<th>% Ascent</th>
<th>No. of Ascent Deleted</th>
<th>No. of Ascent Cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.15</td>
<td>1.333</td>
<td>32.6</td>
<td>5</td>
<td>9</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>1.25</td>
<td>1.675</td>
<td>31.0</td>
<td>6</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>3.21</td>
<td>2.448</td>
<td>42.8</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>1.26</td>
<td>1.362</td>
<td>35.7</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>2.48</td>
<td>1.447</td>
<td>40.4</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>3.34</td>
<td>2.833</td>
<td>29.6</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>3.54</td>
<td>1.335</td>
<td>63.8</td>
<td>27</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>8</td>
<td>4.02</td>
<td>1.947</td>
<td>61.4</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>9</td>
<td>4.03</td>
<td>4.054</td>
<td>60.9</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>3.55</td>
<td>1.335</td>
<td>32.3</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>3.57</td>
<td>1.335</td>
<td>47.5</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

* % gap = (upper bound - lower bound)/lower bound.
* CPU time in seconds on an IBM 3093 (model BX).
* Properties of CPU time required for the ascent procedure.
Multi-level Network Design (MLND)

- Nodes are classified by level of importance into different types
- Correspondingly, we consider different facility (edge) types
- Higher-level nodes must be interconnected by higher grade facilities; this sub-network may optionally include lower-level nodes

- Higher grade facilities are more expensive
- Special case: Two-level Network Design (TLND)
  - Primary nodes $P$, Secondary nodes $S$
  - Two types of facilities: primary and secondary with fixed costs $a_{ij}$ and $b_{ij}$

Multi-level Network Design

[Diagram of multi-level network design]
**TLND Problem formulation**

Minimize \[ \sum_{(i,j) \in E} a_{ij} y_{ij} + \sum_{(i,j) \in k} b_{ij} z_{ij} + \sum_{k \in K} \sum_{(i,j)} c_{ij}^k f_{ij}^k \]

Flow conservation \[ \sum_{j} x_{ij}^k - \sum_{j} x_{ji}^k = \begin{cases} +1 & \text{if } i = O_k \\ -1 & \text{if } i = D_k \\ 0 & \text{otherwise} \end{cases} \ \forall k \in K \]

Primary forcing constraints \[ x_{ij}^k \leq y_{ij} \ \forall (i, j) \in E, k \in P \]

Secondary forcing constraints \[ x_{ij}^k \leq y_{ij} + z_{ij} \ \forall (i, j) \in E, k \in S \]

Nonnegativity \[ x_{ij}^k \geq 0 \ \forall (i, j) \in E, k \in K \]

Integrality \[ y_{ij}, z_{ij} \in \{0, 1\} \ \forall (i, j) \in E \]

---

**Special cases and complexity of TLND**

- **Steiner tree** problem
  - \( b_{ij} = 0 \) for all edges \((i,j)\)

- (Hierarchical) **Path-tree** problem
  - Number of primary nodes \(|P| = 2\)

- **Proportional-cost TLND** problem
  - \( a_{ij} = r b_{ij} \) for all edges \((i,j)\)

The TLND problem is NP-hard even if \(|P| = 2\) and either costs are proportional or \( a_{ij} = 1 \) and \( b_{ij} = 0 \) or 1
Illustrative Worst-case result for TLND problem

Consider two alternative heuristics

- **Secondary Extension (SE) method**
  - Connect primary nodes via Steiner tree
  - Extend this tree with secondary facilities to span remaining secondary nodes

- **Primary Upgrade (PU) method**
  - Connect all nodes via min spanning tree, with secondary facilities
  - Upgrade facilities on induced primary subtree

- SE method is near-optimal when secondary costs are small, whereas PU method is near-optimal when secondary costs are close to primary costs

- **Hybrid** method: select the better of the SE and PU solutions

**Theorem:** For the proportional-cost TLND problem, if $\rho$ denotes the worst-case ratio of the Steiner tree solution in the SE method, then the Hybrid method has worst-case ratio of $\frac{4}{4 - \rho}$, and the bound is tight

MLND Solution strategy

- Problem pre-processing to eliminate edges, flow variables
- Dual ascent to
  - Generate lower bound
  - Identify feasible solution
  - Reduce the problem (fix variables)
- Heuristic improvement of dual-based solution
Importance of good model formulation

- Dual ascent methods are successful when problem has special structure and *model formulation is tight*

**Tight problem formulations**

- Obtained by adding valid inequalities (ideally, facets) and expanding the set of variables
- May vastly increase problem size, but help to:
  - Generate good lower bounds
  - Better guarantees for heuristic solution quality
  - Improve algorithmic performance (lesser enumeration needed in exact algorithms)
  - Identify better heuristic solutions
- Use iterative (cutting plane, column generation) methods to cope with larger problem size
**Example: Forcing constraints in Network Design models**

Assume undirected edges

**Aggregate** forcing constraints

\[ \sum_{k \in K} x_{ij}^k \leq K | z_{ij} \quad \& \quad \sum_{k \in K} x_{ji}^k \leq K | z_{ji} \quad \forall (i, j) \in E \]

**Disaggregate** forcing constraints

\[ x_{ij}^k \leq z_{ij} \quad \& \quad x_{ji}^k \leq z_{ji} \quad \forall (i, j) \in E, k \in K \]

**Bidirectional** forcing constraints (when flow costs are same for all comm.)

Consider pairs of commodities \( k \) and \( h \) with same origin or destination

\[ x_{ij}^k + x_{ji}^h \leq z_{ij} \quad \forall (i, j) \in E; k, h \in K \] with \( O_k = O_h \) or \( D_k = D_h \)

---

**Example: Strength of forcing constraints**

Problem instance

- Fixed cost = 1,
- Variable cost = 0 for all edges

**Aggregate forcing constraint**

\[ Z_{ij} = \frac{1}{2} \]

LP value = 1

**Disaggregate forcing constraint**

\[ Z_{ij} = \frac{1}{2} \] for all edges

LP value = 1.5

**Bi-directional forcing constraint**

\[ Z_{ij} = 1 \]

LP value = 2 = optimal
Learnings

- Network design problems are **important** for strategic, tactical, and operational planning of distribution systems, but they are **challenging**
- Develop **tailored** algorithms that exploit the problem’s **special structure**
- **Strong problem formulations** are critically important
- Combine **multiple techniques**—problem preprocessing, decomposition, efficient subproblems, iterative model enhancement, problem reduction, heuristic search and improvement

Network Design extensions

<table>
<thead>
<tr>
<th>Network Design Model Variant</th>
<th>Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capacitated Network Design (CND)</td>
<td>Capacitated edges, no bifurcation of flows</td>
</tr>
<tr>
<td>Hop-constrained Network Design (HCND)</td>
<td># of edges on flow path must not exceed specified maximum</td>
</tr>
<tr>
<td>Weight-constrained Network Design (WCND)</td>
<td>Total weight (e.g. delay) on flow path must not exceed specified maximum</td>
</tr>
<tr>
<td>Network Loading (NL)</td>
<td>Discrete set of available edge capacities; no flow costs</td>
</tr>
<tr>
<td>Multi-level Network Design (MLND)</td>
<td>Multiple node types; higher level nodes require higher grade facilities (edges)</td>
</tr>
<tr>
<td>Survivable Network Design (SND)</td>
<td>Require disjoint alternate paths between node pairs</td>
</tr>
<tr>
<td>Network Restoration (NR)</td>
<td>Non-simultaneous flow; flow created by failure of edge</td>
</tr>
</tbody>
</table>