Piecewise Linear Multicommodity Flow Problems

Bernard Gendron[1]

December 2016

LAAS, Toulouse

1 CIRRELT and DIRO, U. Montréal

Outline

Problem description

Extended model and polyhedral results

Structured Dantzig-Wolfe decomposition

Computational results

Multicommodity flow problem

- Directed network $G = (N, A)$ with node set N and arc set A
- Commodity set K: origins $O(k)$, destinations $D(k)$ and transshipment nodes $T(k)$ for each $k \in K$
- Demand d_i^k for commodity k at node i:

$$\begin{aligned} &> 0 \quad \text{if } i \in O(k) \\ &< 0 \quad \text{if } i \in D(k) \\ &= 0 \quad \text{if } i \in T(k) \end{aligned}$$

- Capacity u_a on each arc a

Mathematical formulation

x_a^k: flow of commodity k on arc a

$$\sum_{a \in F_i} x_a^k - \sum_{a \in B_i} x_a^k = d_i^k, \quad i \in N,\ k \in K, \tag{1}$$

$$x_a^k \geq 0, \quad a \in A,\ k \in K, \tag{2}$$

$$x_a \equiv \sum_{k \in K} x_a^k \leq u_a, \quad a \in A.$$

$$F_i = \{a \in A | a = (i,j),\, j \in N\},\ B_i = \{a \in A | a = (j,i),\, j \in N\}$$

Piecewise linear costs

- The objective is to minimize the sum of $|A|$ piecewise linear cost functions $g_a(x_a)$ such that:
  - $g_a(0) = 0$
  - $g_a(x_a)$ is lower semi-continuous
  - $g_a(x_a)$ is non-decreasing
- S_a: segments of $g_a(x_a)$
- $0 = b_a^0 < b_a^1 < \cdots < b_a^{|S_a|} = u_a$ (breakpoints)
- $c_a^s \geq 0$: linear cost (slope) of segment $s \in S_a$
- $f_a^s \geq 0$: fixed cost (intercept) of segment $s \in S_a$
- Applications in transportation:
  - Number of trucks used in TL (truckload) transportation
  - Economies of scale in LTL (less-than-truckload) transportation

Basic multiple choice model

x_a^s: total flow x_a, if it falls in $(b_a^{s-1}, b_a^s]$; 0, otherwise

y_a^s: 1, if x_a falls in $(b_a^{s-1}, b_a^s]$; 0, otherwise

$$v(B) = \min \sum_{a \in A} \sum_{s \in S_a} (c_a^s x_a^s + f_a^s y_a^s)$$

subject to (1), (2)

$$\sum_{k \in K} x_a^k = \sum_{s \in S_a} x_a^s, \quad a \in A,$$

$$b_a^{s-1} y_a^s \leq x_a^s \leq b_a^s y_a^s, \quad a \in A, s \in S_a,$$

$$\sum_{s \in S_a} y_a^s \leq 1, \quad a \in A,$$

$$y_a^s \in \{0, 1\}, \quad a \in A, s \in S_a.$$

Flow disaggregation

- M_a^k: upper bound on the flow of commodity k on arc a
- x_a^{ks}: the commodity flow x_a^k, if $x_a^s > 0$; 0, otherwise

$$\sum_{k \in K} x_a^{ks} = x_a^s, \quad a \in A, s \in S_a,$$

$$\sum_{s \in S_a} x_a^{ks} = x_a^k, \quad a \in A, k \in K.$$

- Add the extended forcing constraints:

$$x_a^{ks} \leq M_a^k y_a^s, \quad a \in A, k \in K, s \in S_a.$$

Extended multiple choice model

$$v(E) = \min \sum_{a \in A} \sum_{s \in S_a} \left(c_a^s \sum_{k \in K} x_a^{ks} + f_a^s y_a^s \right)$$

$$\sum_{a \in F_i} \sum_{s \in S_a} x_a^{ks} - \sum_{a \in B_i} \sum_{s \in S_a} x_a^{ks} = d_i^k, \quad i \in N, k \in K, \tag{3}$$

$$b_a^{s-1} y_a^s \leq \sum_{k \in K} x_a^{ks} \leq b_a^s y_a^s, \quad a \in A, s \in S_a,$$

$$0 \leq x_a^{ks} \leq M_a^k y_a^s, \quad a \in A, k \in K, s \in S_a,$$

$$\sum_{s \in S_a} y_a^s \leq 1, \quad a \in A, \tag{4}$$

$$y_a^s \in \{0, 1\}, \quad a \in A, s \in S_a. \tag{5}$$

Lagrangian relaxation

- Relax flow conservation equations (3) in a Lagrangian way (Balakrishnan, Graves 1989)
- In the Lagrangian subproblem: $\sum_{k \in K} x_a^{ks} > 0$ iff $y_a^s = 1$
- Hence, first solve for each arc a and segment s, the following continuous knapsack problem:

$$v(P_a^s(\pi)) = \min \sum_{k \in K} (c_a^s - \pi_{t(a)}^k + \pi_{h(a)}^k) x_a^{ks}$$

$$b_a^{s-1} \leq \sum_{k \in K} x_a^{ks} \leq b_a^s,$$

$$0 \leq x_a^{ks} \leq M_a^k, \quad k \in K.$$

- Then, solve a trivial IP subproblem in variables y_a^s only:

$$v(R(\pi)) = \min \sum_{a \in A} \sum_{s \in S_a} (v(P_a^s(\pi)) + f_a^s) y_a^s$$

subject to (4), (5).

Polyhedral results

- Notation:
  - $v(M)$: optimal value for model M
  - $F(M)$: feasible set for model M
  - $conv(F(M))$: convex hull of $F(M)$
  - $\overline{M}$: LP relaxation for model M
  - $R(\pi)$: Lagrangian subproblem for Lagrange multipliers π
  - D: Lagrangian dual
    $v(D) = \max_{\pi}\{v(R(\pi)) + \sum_{i \in N} \sum_{k \in K} \pi_i^k d_i^k\}$
- $F(\overline{R(\pi)}) = conv(F(R(\pi)))$
  (Croxton, Gendron, Magnanti 2007)
- $v(\overline{B}) \leq v(\overline{E}) = v(D) \leq v(B)$

Approaches for computing $v(\overline{E})$

- Use state-of-the-art LP/MIP solver (with its additional cuts): hopeless for large $|K|$
- Solve D by (subgradient optimization $+$) Dantzig-Wolfe decomposition ($+$ stabilization): interesting
- Use column-and-row generation to solve E directly: **structured Dantzig-Wolfe decomposition (SDW)** (Frangioni, Gendron 2013)

Reformulation and decomposition

- "Structured" MIP:

$$(P) \qquad \min_x \{ \; cx : \; Ax = b \; , \; x \in X \; \}$$

where $\quad (R(\alpha)) \qquad v(R(\alpha)) = \min_x \{ \; cx + \alpha(b - Ax) \; : \; x \in X \; \}$

is "significantly easier" than (P)

- Lagrangian dual:

$$(D) \max_\alpha \{ \; v(R(\alpha)) \; \} = \min_x \{ \; cx : \; Ax = b \; , \; x \in conv(X) \; \} (\overline{P})$$

- Reformulation:

$$conv(X) = \{ \; x = C\theta \; : \; \Gamma\theta \leq \gamma \; \}$$

- Examples:
  - Dantzig-Wolfe reformulation
  - Extended multiple choice model (E)

SDW assumptions

- Assumption 1 (*reformulation*):

$$conv(X) = \{ x = C\theta \ : \ \Gamma\theta \leq \gamma \}$$

- Assumption 2 (*padding with zeroes*):

$$\Gamma_{\mathcal{B}}\bar{\theta}_{\mathcal{B}} \leq \gamma_{\mathcal{B}} \quad \Rightarrow \Gamma \left[\bar{\theta}_{\mathcal{B}}, 0\right] \leq \gamma$$

$$\Rightarrow X_{\mathcal{B}} = \{ x = C_{\mathcal{B}}\theta_{\mathcal{B}} \ : \ \Gamma_{\mathcal{B}}\theta_{\mathcal{B}} \leq \gamma_{\mathcal{B}} \} \subseteq conv(X)$$

- Assumption 3 (*easy update of variables and constraints*):
  Given $\mathcal{B}$, $\bar{x} \in conv(X)$ s.t. $\bar{x} \notin X_{\mathcal{B}}$,
  it is "easy" to find $\mathcal{B}' \supset \mathcal{B}$ and $\Gamma_{\mathcal{B}'}$, $\gamma_{\mathcal{B}'}$ such that
  $\exists \ \mathcal{B}'' \supseteq \mathcal{B}'$ such that $\bar{x} \in X_{\mathcal{B}''}$.

SDW algorithm

$\langle$ initialize $\mathcal{B}$ $\rangle$;
repeat
    $\langle$ solve $(\overline{P}_{\mathcal{B}})$ for $\tilde{x}$, $\tilde{\alpha}$; $\tilde{v} = c\tilde{x}$ $\rangle$;
    $\bar{x} = \operatorname{argmin}_x\{ (c - \tilde{\alpha}A)x : x \in X \}$; /* $(R(\tilde{\alpha}))$ */
    if($\tilde{v} = c\bar{x} + \tilde{\alpha}(b - A\bar{x})$)
        then STOP; /* $\tilde{x}$ optimal */
        else $\langle$ update $\mathcal{B}$ as in Assumption 3 $\rangle$;
until $\sim$ **STOP**

- Finitely terminates with an optimal solution of $\overline{P}$
- ... even if (proper) removal of indices from $\mathcal{B}$ is allowed

Variant 1 of SDW for computing $v(\overline{E})$

- Initialization: solve $\overline{B}$; $v^{last} = v(\overline{B})$
- Column generation iterations:
  - For each arc a:
    1) let $s^* = \arg\min_{s \in S_a}\{v(P_a^s(\pi)) + f_a^s\}$;
    2) if $v(P_a^{s^*}(\pi)) + f_a^{s^*} < 0$, generate variable $y_a^{s^*}$ and, for each $k \in K$, variables $x_a^{ks^*}$
  - If columns have been generated, solve $\overline{E}_{\mathcal{B}}$ and iterate to generate more columns
- If $v(\overline{E}_{\mathcal{B}}) - v^{last} < \epsilon$, STOP; else, $v^{last} = v(\overline{E}_{\mathcal{B}})$
- Row generation iterations:
  - Add violated extended forcing constraints
  - If rows have been generated, solve $\overline{E}_{\mathcal{B}}$ and iterate to generate more rows
- Go to column generation iterations

Equivalent formulation for $\overline{E}$

$$v(\overline{E}) = \min \sum_{a \in A} \sum_{s \in S_a} \left(c_a^s \sum_{k \in K} x_a^{ks} + f_a^s y_a^s \right)$$

$$\sum_{a \in F_i} \sum_{s \in S_a} x_a^{ks} - \sum_{a \in B_i} \sum_{s \in S_a} x_a^{ks} = d_i^k, \quad i \in N, k \in K, \quad (\pi_i^k)$$

$$\sum_{k \in K} x_a^{ks} \leq b_a^s y_a^s, \quad a \in A, s \in S_a, \quad (\alpha_a^s)$$

$$0 \leq x_a^{ks} \leq M_a^k y_a^s, \quad a \in A, k \in K, s \in S_a,$$

$$\sum_{s \in S_a} y_a^s \leq 1, \quad a \in A, \quad (\gamma_a)$$

$$y_a^s \geq 0, \quad a \in A, s \in S_a.$$

Variant 2 of SDW for computing $v(\overline{E})$

- Identical to Variant 1, except for column generation iterations
- For each arc a:
  1) let $s^* = \arg\min_{s \in S_a}\{f_a^s(\pi, \alpha)\}$, where
  $f_a^s(\pi, \alpha) = f_a^s - b_a^s \alpha_a^s + \sum_{k \in K} c_a^{ks}(\pi, \alpha)$ and
  $c_a^{ks}(\pi, \alpha) = \min\{0, c_a^s - \pi_{t(a)}^k + \pi_{h(a)}^k + \alpha_a^s\}$;
  2) if $f_a^{s^*}(\pi, \alpha) < -\gamma_a$, then, for each k such that
  $c_a^{ks^*}(\pi, \alpha) < 0$, generate variables $x_a^{ks^*}$;
  3) for each s such that $y_a^s > 0$, for each k such that
  $c_a^{ks}(\pi, \alpha) < 0$, generate variables x_a^{ks}
- This column generation method corresponds to the application of SDW based on the **Lagrangian relaxation of flow conservation equations and capacity constraints**

Implementation and data instances

- **Network design** instances ($|K| = 40, 100, 200, 400$):
  - Commodity k = O-D pair with demand d^k
  - $b_a^s = sb_a$, $f_a^s = sf_a$, $s = 0, 1, \ldots, \lceil \sum_k d^k / b_a \rceil$
  - $c_a^s x_a^s$ replaced by $\sum_{k \in K} c_a^k x_a^k$
- All IPs and LPs solved with CPLEX 12.4
- B and E solved with CPLEX
- Variant 1 (Frangioni, Gendron 2009): solving the root relaxation, then freezing the formulation + CPLEX polishing for one hour
- Variant 2 (El Filali 2014): implemented with the B&P&C in SCIP 3.0.1 (Achterberg 2009)
- gap = final gap (%), cpu = time (limit = 3h)

Results at the root

Best known upper bound for each instance used to compute gaps

No lower bound after 3h: gap = 100%

$\|K\|$	B	E	SDW1	SDW2
40	2.4, 221	0.2, 14	0.4, 0	0.4, 3
100	16.3, 10751	25.6, 4095	1.8, 181	1.8, 229
200	22.4, 10781	88.2, 10782	5.7, 1711	5.7, 674
400	31.7, 9049	100.0, 10683	6.8, 10862	6.8, 7159

Table : Lower bounds at the root: gap (%), cpu (s)

Comparing the two SDW variants

No initial upper bound

Unlike SDW2, solved by B&P&C, frozen formulations in SDW1 implementation may **not** contain optimal solution

$\lvert K \rvert$	SDW1	SDW2
40	0.4, 0	0.0, 182
100	1.8, 1551	1.2, 9050
200	5.7, 5174	4.7, 10800
400	6.8, 11250	6.8, 10803

Table : SDW1 versus SDW2: gap (%), cpu (s)

Conclusions

- Frangioni, Gendron (2009) show SDW1 is more efficient than:
  - Solving D with an aggregated bundle method
  - Solving the special case of network design by a cutting-plane method based on residual capacity inequalities (Magnanti, Mirchandani, Vachani 1993; Atamtürk, Rajan 2002)
- SDW2 is as effective as SDW1 and more efficient when $|K|$ gets larger
- Forthcoming:
  - SDW3: generate both y and x variables simultaneously
  - Compare the three variants under the same B&P&C interface
  - Add stabilization techniques to speedup column generation (Frangioni, Gendron 2013)