On Solving Origin-Destination Multicommodity Network Flow(ODMCNF) Problem

I-Lin Wang

1.0 Multicommodity Problems:

TABLE 1. Multicommodit Problems

Problem	**Fleet Assignment**	**Crew Scehduling**	**Air Cargo Routing**
Commodity	A/C Types	Crews available at different Crew Bases, at different Times	O-D Shipments
Flow Balance	Conservation of A/C	Conservation of Crews	Conservation of shipments
Bundle Constraint	Each Flight Covered by Exactly 1 A/C Type	Each Flight Covered by Exactl 1 Crew	Flight capacity, Cargo Terminal Capacity

1.1 Linear MCNF Formulation:Given a network G(N,A), with |K| commodities

TABLE 2. Comparison of MCNF formulation

Node-Arc Oriented Formulation	**Path Oriented Formulation**
$min \sum_{k \in K} \sum_{a \in A} c_a^k x_a^k$ #vars: $\lvert A\rvert \lvert K\rvert$ s.t. $\sum_{k \in K} x_a^k \le u_a \quad \forall a \in A$ #rows: $\lvert A\rvert$ $N^k x^k = b^k \quad \forall k \in K$ #rows: $\lvert N\rvert \lvert K\rvert$ $x_a^k \ge 0 \quad \forall a \in A \ \forall k \in K$	$min \sum_{k \in K} \sum_{p \in P^k} PC_p^k f_p^k$ #vars: $\sum_k \lvert P^k \rvert$ s.t. $\sum_{k \in K} \sum_{p \in P^k} \delta_a^p f_p^k \cdot b^k \le u_a \quad \forall a \in A$ #rows: $\lvert A\rvert$ $\sum_{p \in P^k} f_p^k = 1 \quad \forall k \in K$ #rows: $\lvert K\rvert$ $f_p^k \ge 0 \quad \forall p \in P^k \ \forall k \in K$ $\delta_a^p = \begin{cases} 1 \text{ , if path p passes arc a} \\ 0 \text{ , o.w.} \end{cases}$, $PC_p^k = b^k \sum_{a \in A} AC_a^k \delta_a^p$
Subproblem: min-cost network flow problem	**Subproblem: shortest path problem**
Many constraints,	Fewer constraints, Much more columns, Easier subproblem

1.2 Solution Methods;

(a)Resource Directive (b)Price-Directive (c)Partitioning

- General Concept: try to decompose a hard problem into a set of easy problems

1. Price-Directive Decomposition (Lagrangian Relaxation vs D-W):

- **Lagragian Relaxation**
  - Relax Bundle Constraints, Using Subgradient Method to get the "right" Lagrangian Multiplier
  - Give a lower bound on the optimal objective value
  - No Monotonically Convergence on Multiplier or Objective, Worse Convergence Rate --Tailing Effect

- **Dantzig-Wolf Decomposition**
  - Find extreme Point(BFS) to each of the |K| subproblems
  - Each subproblem is a Single Commodity Network Flow Problem
  - Find Convex Combination of the Subproblem Solutions Satisfying Bundle Constraints
  - Too Many Columns(BFS), use Column Generation
  - Like LR, try to get "right" Dual Variable by solving a LP (Bundle Constraint)
  - Better Convergence Rate than LR, but each iteration does much more work
  - still has Tailing Effect--may caused by Degeneracy

2. Partitioning:

- Idea: By changing variables, try transform the original Basis into another matrix so that we can perform simplex method much more efficiently
-
- Specialization of Rosen's Partitioning Method for Angular & Dual-Angular problem
  - Determine Primal Feasible Bases
  - Relax Nonnegativity on these Basic Variables (make the problem size much smaller)
  - Eliminate All Basic Variables through Substitution
  - Solve the Reduced Problem. If the original Basic Variables are Nonnegative then we are done; Otherwise, Exchange Basis for those Basis corresponding to negative Basic Variables
  - Repeat until All Basic Variables are Nonnegative

1.3 Primal and Dual Formulation for Path-Oriented Formulation:commodities

TABLE 3. Primal & Dual PATH-Oriented Formulation

PATH Primal Formulation

$$min \sum_{k \in K} \sum_{p \in P^k} PC_p^k f_p^k + \sum_a M\alpha_a \quad ....(1)$$

Primal Var (f_p^k, α_a)

s.t.

Dual Var

$$\sum_{k \in K} \sum_{p \in P^k} \delta_a^p f_p^k \cdot b^k - \alpha_a \le u_a \quad \forall a \in A \; ....(2) \qquad (-\pi_a \ge 0)$$

$$\sum_{p \in P^k} f_p^k = 1 \qquad \forall k \in K \; ....(3) \qquad \sigma_k \text{ free}$$

$$f_p^k \ge 0 \quad \forall p \in P^k \quad \forall k \in K \;,\; \alpha_a \ge 0 \quad \forall a \in A$$

$$\delta_a^p = \begin{cases} 1 \text{ , if path p passes arc a} \\ 0 \text{ , o.w.} \end{cases} \;,\; PC_p^k = b^k \cdot \sum_{a \in A} AC_a^k \delta_a^p$$

PATH Dual Formulation

$$max \sum_a (-\pi_a) u_a + \sum_k \sigma_k \quad ....(4)$$

s.t.

$$\sum_a (-\pi_a)\delta_a^p \cdot b^k + \sigma_k \le PC_p^k \quad \forall p \in P^k \quad \forall k \in K \; ....(5)$$

$$\pi_a \le M \quad , \quad -\pi_a \ge 0 \qquad \forall a \in A$$

$$\sigma_k \quad \text{free} \qquad \forall k \in K$$

Define ${}^{\pi}PC_p^k = PC_p^k + \sum_a \pi_a \delta_a^p \cdot b^k = b^k \cdot \sum_a (AC_a^k + \pi_a)\delta_a^p$, that is, we can think of ${}^{\pi}PC_p^k$ as a new path length computed by the new arc cost $(AC_a^k - \pi_a) \ge 0$ for each arc a.

Then, (4) can be rewritten as ${}^{\pi}PC_p^k \ge \sigma_k \quad (4^*)$

1.4 Dantzig-Wolfe Decomposition & Column Generation:

- Let $-\pi_a \geq 0$ be the dual variable associated with each arc in (2); σ_k be the free dual variable associated with each commodity in (3); Also, we add surplus $\alpha_a \geq 0$ for each arc so that (2) will always be satisfied. Then we also add a big cost M for each slack in the objective function.
- Assume all arc cost are nonnegative. In the first iteration, for each commodity, choose the shortest path to be the basic variable, thus we construct the first Restricted Master Problem(RMP) by only using the columns associated with these basic variables and the surplus columns. Using any LP solver to solve it, we can obtain the optimal dual variables (π_a^*, δ_k^*).
- For each commodity, run the shortest path algorithm using the new arc length as $(AC_a^k + \pi_a^*)$. Then we compare the shortest path length ${}^{\pi}PC_k^p$ with δ_k^*. If ${}^{\pi}PC_k^p < \delta_k^*$, we will add that column(path) to the RMP.
- After adding all the necessary paths for all the commodities, we solve the RMP by some LP solver to obtain a new set of optimal dual variables (π_a^*, δ_k^*). We repeat last step and this step until no columns could be generated. Thus we achieve the dual feasibility while maintaining the primal feasibiility & C.S. That is, it is optimal.

1.5 Improve the Original DW/Column Generation by Rosen's Algorithm:

- When the problem has many OD pairs (worst case: $|K|=|N|^2$), it will be painful to solve the RMP.
- Relax convexity constraints by Rosen's algorithm. Thus we only need to solve the Bundle Constraints in the RMP
- First, we reformulate PATH as CYCLE by eliminating all the basic variables(key paths) from the problem. Then, we relax the convexit constraints in CYCLE to get a new reduced problem RELAX.
- For each commodity *k*(an OD pair), we select a path, *key(k)*. Let $\sum_k PC_{key(k)} = CT$.
- The CYCLE formulation is like we choose these key paths to send flow through, then any other path with the same OD will form several cycles with the key path. We then shift flow over the cycles.
- The CYCLE formulation is exactly the same as the PATH formulation except it is obtained by some row operations from the PATH formulation by eliminating the key path variables.

TABLE 4. Primal & Dual CYCLE Formulation

CYCLE Primal Formulation
$min \sum_{k \in K} \sum_{p \in P^k} (PC_p^k - PC_{key(k)}) x_p^k + \sum_a M\alpha_a$ ....(6)   s.t. $\sum_{k \in K} \sum_{p \in P^k} (\delta_a^p - \delta_a^{key(k)}) x_p^k \cdot b^k - \alpha_a \le u_a - \sum_k b^k \delta_a^{key(k)} \quad \forall a \in A$ ....(7) $\sum_{p \in P^k} x_p^k = 1 \quad \forall k \in K$ ....(8) $x_p^k \ge 0 \quad \forall p \in P^k \quad \forall k \in K \; , \alpha_a \ge 0 \quad \forall a \in A$ $\delta_a^p = \begin{cases} 1 \text{ , if path p passes arc a} \\ 0 \text{ , o.w.} \end{cases} \; , PC_p^k = b^k \cdot \sum_{a \in A} AC_a^k \delta_a^p$
CYCLE Dual Formulation
$max \sum_a (-\pi_a) \left(u_a - \sum_k b^k \delta_a^{key(k)} \right) + \sum_k \sigma_k$ ....(9)   s.t. $\sum_a (-\pi_a)(\delta_a^p - \delta_a^{key(k)}) \cdot b^k + \sigma_k \le PC_p^k - PC_{key(k)} \quad \forall p \in P^k \quad \forall k \in K$ ....(10) $\pi_a \le M \quad , \quad -\pi_a \ge 0 \quad \forall a \in A$ $\sigma_k \quad \text{free} \quad \forall k \in K$
Define $CC_p^k = PC_p^k - PC_{key(k)}$ as the cycle cost(cost difference between *p* and key path *key(k)*)   By using the definition of ${}^{\pi}PC_p^k$, (10) can be rewritten as ${}^{\pi}PC_p^k - {}^{\pi}PC_{key(k)} \ge \sigma_k$ (10^*)

- By (10^*), we generate a column when its corresponding path has length shorter than its keypath's plus σ_k.
- To further reduce the problem size, we relax the convexity constraints (8), and construct a new formulation RELAX(i), where i denodes the iteration index.

TABLE 5. Primal & Dual CYCLE-RELAX Formulation

RELAX(i) Primal Formulation

$$min \sum_{k \in K} \sum_{p \in P^k} CC_p^{k,i} x_p^{k,i} + \sum_a M\alpha_a^i \quad ....(11)$$

s.t.

$$\sum_{k \in K} \sum_{p \in P^k} (\delta_a^p - \delta_a^{key(k,i)}) x_p^{k,i} \cdot b^k - \alpha_a^i \le u_a - \sum_k b^k \delta_a^{key(k,i)} \quad \forall a \in A \quad ....(12)$$

$$x_p^{k,i} \ge 0 \quad \forall p \in P^k - key(k,i) \quad \forall k \in K \quad , \alpha_a \ge 0 \quad \forall a \in A$$

$x_{key(k,i)}$ free

$$\delta_a^p = \begin{cases} 1 \text{ , if path p passes arc a} \\ 0 \text{ , o.w.} \end{cases} \quad , PC_p^k = b^k \cdot \sum_{a \in A} AC_a^k \delta_a^p$$

RELAX(i) Dual Formulation

$$max \sum_a (-\pi_a^i) \left(u_a - \sum_k b^k \delta_a^{key(k,i)} \right) \quad ....(13)$$

s.t.

$$^{\pi}PC_p^{k,i} - {}^{\pi}PC_{key(k,i)} \ge 0 \quad \forall p \in P^k - key(k,i) \quad \forall k \in K \quad ....(14)$$

$$\pi_a \le M \quad , \quad -\pi_a \ge 0 \qquad \forall a \in A$$

Note that in this RELAX formulation, all the convexity constraints and their associated dual variable σ_k are all gone.

1.6 Rosen's Algorithm:

STEP0: START i=0

STEP1: i=i+1

STEP2: Select Keypaths for each commodity

STEP3: Construct RELAX(i)

STEP4: Solve RELAX(i) (using Column Generation here)

STEP5: Are all keypath has nonnegative flow? If yes, we are DONE. Otherwise, GOTO Step 2.

Step 3: Select Keypath for each Commodity:

- In the first iteration, we simply apply choose the shortest path using the original arc length AC_a. In the next iteration, if the keypath has nonnegative flow, then we keep it. Otherwise, we choose a new keypath.
- First we calculate the path flow for commodity k in the i+1th iteration:

$$x_p^{i+1} = \begin{cases} 1 - \sum\limits_{p \neq key(k,i)} x_p^{i*} \\ x_p^{i*} , \forall p \neq key(k,i) \end{cases} \quad , \ \alpha_a^{i+1} = \alpha_a^{i*}$$

- Since we only relax the nonnegativity constraint for the keypath in the previous iteration, those paths are the only variables possiblly carrying negative flows. That is, for the same OD pair, there must exist some other path carrying positive flow. To maintain the C.S of RELAX(i+1), we can choose any such path as a new key-path.
- If there are multiple choices of the new keypath, intuitively, we will choose the one carrying the biggest amount of flow because it looks more likely to be in basis.

Step 4: Construct RELAX(i) :

- We eliminate the variables corresponding to the kaypath by some column operations. In the first iteration, for the constraint coefficient matrix, we will get |K| zero columns and a negative identity matrix with dimension |A| corresponding to those surplus variables. Similarly, the objective coefficient will be |K| zero elements and |A| elements of big surplus cost M.
- In general, after we have determine the new keypath for each commodity, we have to update the constraint coefficient matrix and the objective cost vector by making those columns corresponding to the new keypaths to be zero.

Step 5: Solve RELAX(i) by Column Generation

- After we constructed the RELAX(i), we can use any LP solver to solve it, obtain the optimal dual variable π_a^{i*} for each arc. We then apply any shortest path algorithm using new arc length as $AC_a^k + \pi_a^{i*}$ and get the shortest path q^*_k with path length ${}^{\pi}PC_q^{k,i}$ for commodity k.
- Then we compare ${}^{\pi}PC_q^{k,i}$ with ${}^{\pi}PC_{ke(k,i)}$. If ${}^{\pi}PC_q^{k,i} \leq {}^{\pi}PC_{ke(k,i)}$ then we generate the column corresponding to path q because that column has negative reduced cost. Otherwise, we just leave the current keypath alone.
- We generate column for each commodity, then we solve the new RMP again to get a new set of optimal dual variables.
- We keep generating columns and solving RMP until no more columns could be generated. THat is, we are at optimality of RELAX(i).

1.7 Finiteness of Rosen's Algorithm:

- Optimal dual solution to RELAX(i) is a b.f.s. for the DRELAX(i+1), and CYCLE.
- Use dual simplex arguement, these dual b.f.s will have associated objective values that strictl increase at each iteration. Using some method to avoid the degeneracy, Rosen's algorithm will finish in finite number of steps.

Claim1: Optimal solution to DRELAX(i) is a b.f.s. for DRELAX(i+1)

Claim2: $Z^*_{\text{DRELAX(i)}} + \text{CT(i)} = Z_{\text{DRELAX(i+1)}} + \text{CT(i+1)}$

Claim3: Dual b.f.s. π^{i+1} for DRELAX(i+1) & Primal b.s. (x^{i+1}, α^{i+1}) satisfy C.S.

Claim4: $Z^*_{\text{DRELAX(i+1)}} + \text{CT(i+1)} > Z^*_{\text{DRELAX(i)}} + \text{CT(i)}$

1.8 Further Improvement:

- Generate More Simple Paths (symmetric difference of path q with keypath)
- Trying to avoid Degeneracy (by NNLS??)