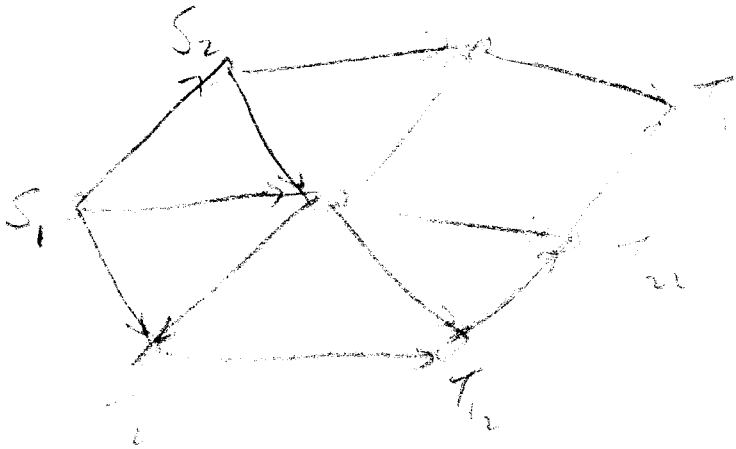


MULTICOMMODITY NETWORK FLOW PROBLEMS

(Goldfarb et al, ~~Annals of OR, 1996~~)

(An example of the use of the Lagrangian relaxation)

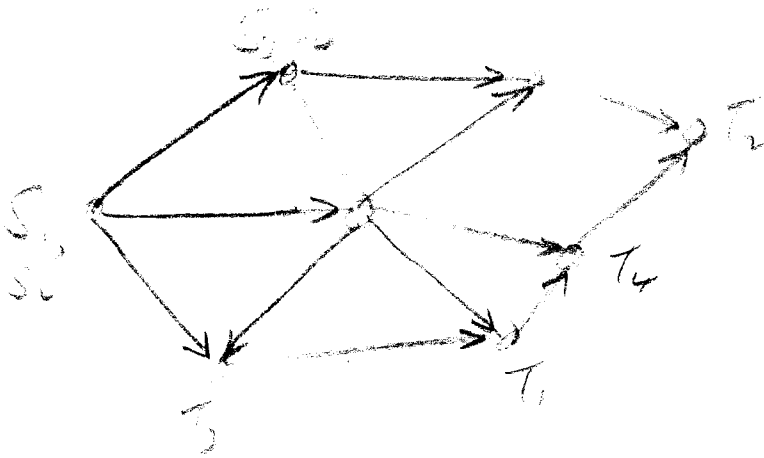


$S_i$ : source for commodity  $i$

$T_j$ : sinks for commodity  $i$ .

SOMD: single origin, multiple destination.

Break into SOSD: single origin, single destination, by splitting flows of each commodity:



$I$  = set of commodities

Eg: telephone calls.

Have capacity  $\gamma_a$  on each arc  $a$ .  
 $A$  = set of arcs

Need flow  $p_i$  from source  $i$  to sink  $i$ .

Let  $x_a^i$  = flow from source  $i$  to sink  $i$ . ( $x_a^i$  = flow on arc  $a$ )

Then  $x^i$  satisfies 
$$\sum_a x_a^i = \begin{cases} 0 & \leftarrow \text{source for flow } i \\ -p_i & \leftarrow \text{sink for flow } i \\ p_i & \\ 0 & \end{cases}$$

node-arc incidence matrix

This flow has a cost  $c_a^i$  on arc  $a$  for each unit used.

So: 
$$\min \sum_{i \in I} \sum_{a \in A} c_a^i x_a^i \quad (1)$$

s.t. 
$$N x^i = b^i \quad \forall i \in I \quad (2)$$

$$\sum_{i \in I} x_a^i \leq \gamma_a \quad \forall a \in A \quad (3)$$

$$x_a^i \geq 0 \quad \forall i \in I, \forall a \in A. \quad (4)$$

~~Convex, so~~

LI, so convex, so if we take a Lagrangian dual then we will get no duality gap.

Separable without the ~~constraint~~ constraint (3).

So put that into the Lagrangian:

$$L(x, u) = \sum_i \sum_a c_a^i x_a^i + \sum_{a \in A} u_a \left( \sum_{i \in I} x_a^i - \gamma_a \right)$$

Then 
$$\theta(u) = \min L(x, u)$$

s.t. 
$$\begin{aligned} N x^i &= b^i & \forall i \in I \\ x_a^i &\geq 0 & \forall i \in I, \forall a \in A. \end{aligned}$$

(LD(u))

Separable: requires solving  $|I|$  shortest path problems.

Let  $\bar{x}$  solve  $(LD(\bar{u}))$ .

Then  $\xi = \sum_{i \in I} x_i^i - y_a$  is the a.k.a component of a subgradient of  $\mathcal{D}(\bar{u})$ .

So: Could move in the direction  $\xi$ , i.e., update  $\bar{u}$  to  $\bar{u} + \alpha \xi$ .  
 How far to move in this direction?

One possibility: Step of length  $\frac{1}{k}$  at iteration  $k$ .

If any  $u_i$  becomes negative, make it zero.

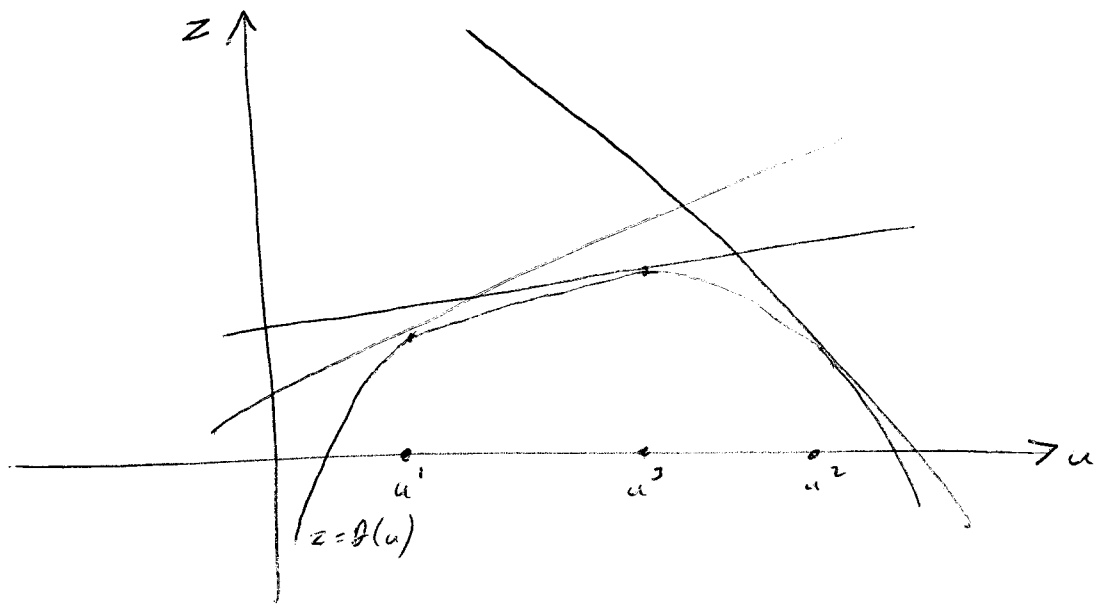
~~(S, y)~~

Alternative:

Know  $\mathcal{D}(u) \leq \mathcal{D}(\bar{u}) + \xi^T(u - \bar{u})$  since  $\xi$  is a subgradient.

Actually, get subgradient  $\xi^k$  at each iteration.

So:  $\max_{z, u} z$  s.t.  $z \leq \mathcal{D}(u^k) + \xi^T(u - u^k)$  (MP)



Algorithm:

- (1) Solve (MP) to get a dual point  $\bar{u}$ , and an upper bound  $z^{UB}$
- (2) Solve  $(LD(\bar{u}))$  to generate a new subgradient,  $\xi^k$ , and possibly update the lower bound  $z^{LB} = \max\{\theta(\bar{u}^{k+1}), z^{LB}\}$ .
- (3) If  $z^{UB} - z^{LB}$  small enough, STOP.  
Else, return to (1).

In practice, can improve the algorithm by DISAGGREGATION of the subgradients:

Have  $\xi_a = \sum_{i \in I} \bar{x}_a^i - \gamma_a$ ,  $\theta(u) = \sum \theta^i(\bar{x}, u)$   
 where  $\theta^i(\bar{x}, u) = \min_{x_a^i} \sum_a c_a^i x_a^i$

and  $z \leq \theta(\bar{u}) + \xi^T(u - \bar{u})$ .

Instead:

~~Let  $z = \sum_{i \in I} z_i$~~

~~and  $z_i \leq \theta^i(\bar{u}) + \bar{x}^i{}^T(u - \bar{u})$~~

Break up Lagrangian as

$$L(x, u) = \sum_i \sum_a c_a^i x_a^i + \sum_{a \in A} u_a \left( \sum_{i \in I} x_a^i - \gamma_a \right)$$

$$= - \sum_{a \in A} u_a \gamma_a + \sum_{i \in I} \left( \sum_{a \in A} (c_a^i + u_a) x_a^i \right)$$

$$= - u^T \gamma + \sum_{i \in I} L_i(x^i, u)$$

$$\text{So } \theta(u) = \min -u^T \gamma + \sum_{i \in I} L_i(x^i, u)$$

$$\text{s.t. } Nx^i = b^i \quad i \in I$$

$$x^i \geq 0 \quad i \in I$$

$$= -u^T \gamma + \sum_{i \in I} \left\{ \begin{array}{l} \min L_i(x^i, u) \\ \text{s.t. } Nx^i = b^i \\ x^i \geq 0 \end{array} \right\}$$

$$= -u^T \gamma + \theta_{\frac{b^i}{N}}^i(u)$$

So get subgradients for each  $\theta_{\frac{b^i}{N}}^i$  separately.

Approximate dual problem by:

$$\max_{u, z} z$$

$$\text{s.t. } z = \sum_{i \in I} z_i - u^T \gamma$$

$$z_i \leq \theta^i(\bar{u}^k) + x^{ik^T} (u - u^k) \quad \forall i,$$

$$\forall \text{ iterations } k.$$

This is a far larger problem than (MP),

but the extra constraints result in a better estimate for the optimal Lagrange multipliers.

Notes:

1. Can extend to convex objective functions.

Eg: Capacity of arcs is variable, with a convex cost function, eg, as flow increases, arc becomes more clogged, so pay more,

eg,  $f_a(y_a) = \frac{y_a^2}{(y_a - y_a)}$  where  $y_a$  = current flow,  $y_a$  = absolute capacity.

2. Computational: On ~~lower~~ PC, solve up to 500 nodes, 1000 arcs, 5000 commodities, in up to 4 hours.