Notes 4

- Recall that we can write the shortest path problem from s to t as an LP:
  - c_{ij} cost for edge (i,j) [data]
  - y_{ij} # of trucks on edge (i,j) [variable]

$$\min_{y} \sum_{(i,j)\in E} c_{ij}\, y_{ij}$$

$$\text{s.t.} \quad \sum_{(i,s)\in E} y_{is} - \sum_{(s,i)\in E} y_{si} = -1$$

$$\sum_{(i,t)\in E} y_{it} - \sum_{(t,i)\in E} y_{ti} = 1$$

$$\sum_{(i,a)\in E} y_{ia} - \sum_{(a,i)\in E} y_{ai} = 0 \quad \forall a \in V \quad \begin{matrix} a \neq s \\ a \neq t \end{matrix}$$

$$y_{ij} \in \{0,1\}$$

$$\Downarrow$$

But we mentioned we can substitute the binary constraints for $y_{ij} \geq 0$.

- Let's get some intuition about what makes this problem special and allows us to drop the binary constraints.

- Let's re-write the constraints in matrix notation

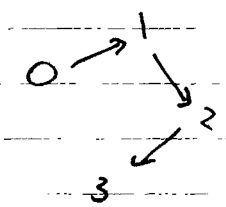
length m, one for every edge

$$A \, y = b$$

$n \times m$

one for every edge

one constraint for every row.

- what does A look like? e.g. 

$$\begin{array}{c|ccc|}
 & (0,1) & (1,2) & (2,3) \\
\hline
0 & -1 & 0 & 0 \\
1 & 1 & -1 & 0 \\
2 & 0 & 1 & -1 \\
3 & 0 & 0 & 1 \\
\hline
\end{array}$$

the source node of the edge has a -1 and the destination has a 1.

- A is called the "node-arc incidence matrix" and it's special property is that every column of A has exactly one $+1$ and one -1.

- How does this property allow us to drop the binary constraints?

- every column has exactly one +1, exactly one −1

$$\Downarrow$$

- the determinant of every basis matrix of A is either +1 or −1

$$+$$

- every extreme point of the feasible region, $\hat{y}$ is the solution to

$$B \hat{y} = b$$

some basis matrix.

- Cramer's rule from Linear Algebra tells us that

$$\hat{y}_j = \frac{\det(B \text{ with } j\text{th column replaced by } b)}{\det(B)}$$

jth coordinate of $\hat{y}$

this is either +1 or −1

this is either +1, −1 or 0, since b has one +1, one −1.

- since $\hat{y}_j \geq 0$, we know they are either 0 or 1.

- What happens if b has other integer values?
- what happens if we solve the LP with an interior point method?

- So, that is why we can drop the $y_{ij} \in \{0,1\}$ in favor of $y_{ij} \geq 0$ and know that the solutions we get out are either 0 or 1.

- the special property is called "unimodularity" and A is called a "totally unimodular" constraint matrix [Means determinant of every basis is either +1 or −1]

- So, shortest path as an LP:

$$\min_{Y} \sum c_{ij} y_{ij}$$

s.t. NET FLOW Constraints.

$$y_{ij} \geq 0 \quad \forall (i,j) \in E$$

- But should we solve shortest path as an LP or using Dijkstra's?

  - LP run time $O(big \cdot small^2) \Rightarrow O(m \cdot n^2)$
    ... in practice not that bad.

  - Dijkstra: $O(n \log n + m \log n)$

e.g. 1000 node graph with 2000 edges?

4.6

Interdicting Shortest Path

- LP formulation of shortest path is still very useful.

- Suppose a bad guy is trying to get from s to t, and we want to make his life as difficult as possible...

we want to maximize his shortest path by lengthening some edges.

- mathematically,

$$\max_{x \in \overline{X}} \quad \min_{y} \sum_{(i,j)\in E} (c_{ij} + d_{ij} x_{ij}) \, y_{ij}$$

s.t.

$$\sum_{(i,s)\in E} y_{is} - \sum_{(s,i)\in E} y_{si} = -1$$

$$\sum_{(i,t)\in E} y_{it} - \sum_{(t,i)\in E} y_{ti} = 1$$

$$\sum_{(i,n)\in E} y_{in} - \sum_{(n,i)\in E} y_{ni} = 0 \quad \forall n \in V,\ n \neq s,\ n \neq t$$

$$y_{ij} \geq 0 \quad \forall (i,j) \in E.$$

- think of $\overline{X}$ as: we can attack only a few places.

$$\sum_{(i,j)\in E} x_{ij} \leq 5$$

$$x_{ij} \in \{0,1\}$$ do we attack edge (i,j)?

4.7

This says that:

1. First we get to select a few edges to lengthen.

2. Second, the bad guy gets to do a shortest path from s to t, knowing what edges we've lengthened.

x tells us the best attack plan on the network.

- So, how do we find x? Can we stick it into a solver?

  ... no, the max min is a problem.

- But, we can turn it into a max max by taking the dual of the inner LP.

Dual:

$$\max_{x \in \underline{X}} \; \max_{\pi} \; \pi_t - \pi_s$$

$$\pi_j - \pi_i \leq c_{ij} + d_{ij} x_{ij} \quad (i,j) \in E$$

where there is one π_n variable for each node $n \in V$.

- The π_n variables always appear in pairs like: $\pi_i - \pi_j$.

so, we can add or subtract a value to all of them, without changing the objective function value or the constraint feasibility.

in other words, if $\pi = (\pi_0, \pi_1, \dots, \pi_n)$ is feasible and has obj value $obj(\pi)$, then

$\pi'(\pi_0 - c, \pi_1 - c, \dots, \pi_n - c)$ is also feasible and has the same obj function value.

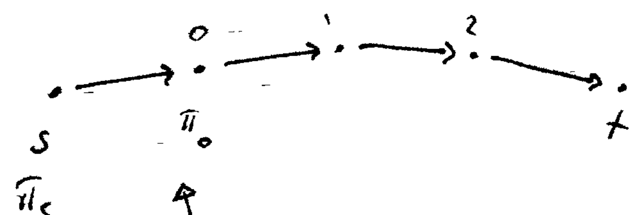
- so, we can add the constraint. $\pi_s = 0$.

$$\min_{x \in \underline{X}} \max_{\pi} \ \pi_t - \pi_s$$

$$\pi_j - \pi_i \leq c_{ij} + d_{ij} x_{ij} \qquad (i,j) \in E$$

$$\pi_s = 0$$

- interpretation of the π_i's?



← want to maximize this

how big can this be?

... gives another way of writing shortest path as LP.

- now, we have a regular MIP, that we can stick into a solver.

- that works just fine when we have a small number of x_{ij}'s... when there are only a few edges we can ^choose from^ attack.

- But, the LP relaxation of the MIP is weak, so it doesn't work great when we have many x_{ij}'s.

remember branch and bound?

Left Right.

fixed a variable (e.g x_{12}) to zero

fixed the variable to one

- get upper bound from LP relaxation
- get lower bound from finding a specific setting of all x_{ij}'s

- upper bound
- lower bound.

- We're hoping the lower bound of left is bigger than the upper bound of right, it lets us eliminate right.
- Doesn't work if the upper bound is huge, from a weak relaxation.

4.10

- What works better when we have lots of x_{ij}'s is doing Bender's decomposition.

- You've learned about Bender's before, so, instead of going over it again, let's see what it means in this case

- here is the "full master" (can get here from Bender's arguments)

$$\max_{\theta, x} \quad \theta$$

$$\theta \le \sum_{(i,j)\in P} (c_{ij} + d_{ij} x_{ij}) \quad \forall \text{ paths } P \text{ from } s \text{ to } t.$$

if all x_{ij} are 0, we don't attack anything, and θ is just equal to the shortest path distance. (this gives a third, inefficient way of writing shortest path)

- the full master is too big to actually write down and solve. So, we'll generate rows as we go along.

4.11

Solution plan:

- start the partial master just with the constraint corresponding to the uninterdicted shortest path from s to t.

$\bar{\theta}$ upper bound

$\underline{\theta}$ lower bound

while $\bar{\theta}$ is far from $\underline{\theta}$

- get a $\hat{\theta}$, $\hat{x}$ from the partial master.
  - $\hat{\theta}$ is an upper bound on our optimal solution value (because it comes from a relaxed problem with fewer constraints) set $\bar{\theta}$ to $\hat{\theta}$.
  - $\hat{x}$ gives us the best interdiction plan, ~~given~~ considering only the paths added so far to the partial master.

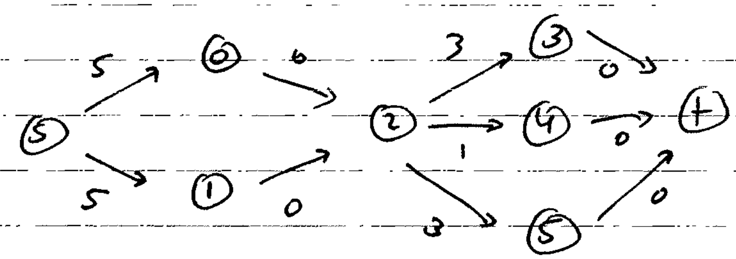
- under $\hat{x}$'s interdiction plan, solve for the bad guy's "best response" shortest path.

  ... in other words, solve shortest path with edge lengths $c_{ij} + d_{ij}\hat{x}_{ij}$
  - this gives the smallest rhs of the full master... the "most violated" constraint.
  - let that shortest path length be L. If $\bar{\theta} = \hat{\theta} \le L$, the "most violated" constraint is not violated, so we've reached optimality
  - otherwise set $\underline{\theta} = \max(\underline{\theta}, L)$ since $L, \hat{x}$ form an particular interdiction

idea: we are trying out interdiction plans $\hat{x}$

- every time we try an $\hat{x}$, we compute a best response from the bad guy.
- we remember all past best responses to compute the next $\hat{x}$

e.g.



- start partial master using shortest path (s) (1) (2) (4) (t), distance 6, so $\underline{\theta} = 6$. $\bar{\theta} = \infty$ (think we could potentially disconnect)

$$\max_{\theta, x} \quad \theta$$

$$\theta \le (5 + \infty x_{s1}) + (0 + \infty x_{12}) + (1 + \infty x_{24}) + (0 + \infty x_{4t})$$

$$\sum x_{ij} \le 2$$

$$x_{ij} \in \{0,1\}$$

- we get solution $\hat{\theta} = \infty$ $\hat{x}$: only $x_{24} = 1$. $\bar{\theta}$ still ∞.
- new shortest path under $\hat{x}$: (s) (1) (2) (5) (t), length 8.

$\underline{\theta} = \max(\underline{\theta}, L)$, so $\underline{\theta} = 8$.

add constraint $\theta \le 5 + \infty x_{s1} + \infty x_{12} + 3 + \infty x_{35} + \infty x_{4t}$

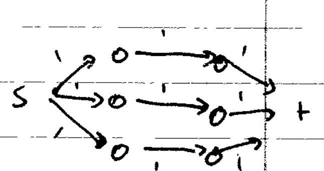
- resolve partial master to get $\hat{\theta} = \infty$, $\hat{x}$: only $x_{12} = 1$.

⋮

- Bender's helps us in two ways.
  1) We got back to using Dijkstra's to solve shortest path.

  2) We avoid "smearing" of the budget, so branch and bound works better.
  In other words, the partial master IP has a better LP relaxation than the IP for interdiction, so its faster to solve.

e.g.



suppose we can interdict these with $d_{ij} = 30$.

- In the example, we wanted to "blow up" edges, not just lengthen them
  we set the d_{ij} to ∞.

- turns out, you don't have to use ∞.

- let $c_{max} = \max_{(i,j) \in E} c_{ij}$, the max length of any edge.

  - every path in G has length $< n \cdot c_{max}$

- we can use $n\, c_{max}$ instead of ∞ because the bad guy would rather walk around the whole graph before using an edge of length $n\, c_{max}$.