

Notes 5.1

• Interdicting max-flow works in much the same way as interdicting shortest paths.

• Let's see what doesn't work:

$$\min_{x \in \bar{X}} \quad \max_{v, y} \quad v$$

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

$$\sum_{(i,t)} y_{it} - \sum_{(t,i)} y_{ti} = v$$

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

$$y_{ij} \leq u_{ij} (1 - x_{ij})$$

$$y_{ij} \geq 0$$



this seems like a tempting thing

to do. Now, if x_{ij} is 1, then we reduce the capacity of arc (i,j) to 0.

• But it doesn't work.

- we can't stick this into a solver because of the min max objective

- even if we do our dual trick, we would have $(1 - x_{ij})u_{ij} * \pi_{ij}$ in the dual objective. So, not a MILP.

- Lets try to formulate it another way:

$$\min_{x \in X} \max_{v, y} v - \sum_{(ij)} d_{ij} y_{ij} x_{ij}$$

$$\text{s.t.} \quad \sum y_{is} - \sum y_{si} = -v$$

$$\sum y_{it} - \sum y_{ti} = v$$

$$\sum y_{ia} - \sum y_{ai} = 0$$

$$y_{ij} \leq u_{ij}$$

$$0 \leq y_{ij}$$

- In this formulation, if $x_{ij} = 1$ then we are penalizing each unit of flow on arc (ij) by the constant d_{ij} .

- If d_{ij} is 1, then any flow that goes through a penalized arc does not contribute to the max-flow objective (since it gets subtracted from v)

- So, attacking an arc does the right thing, it stops the max-flow LP from using that arc to get a better objective

- Let's see what happens when we do our dual trick.

$$\min_{x \in \bar{X}, \beta, \pi} \sum u_{ij} \pi_{ij}$$

$$\beta_s - \beta_t = 1$$

$$\beta_j - \beta_i + \pi_{ij} \geq -d_{ij} x_{ij} \quad \forall (i,j) \in E$$

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

β_i unrestricted

- This is a nice MILP that we can stick into a solver.

- Overall lesson: For our dual trick to work, the attack variables (x_{ij}) have to show up in the primal objective function only. It's ok if they multiply the primal variables (y_{ij}), since the dual trick takes care of that.

If the attack variables show up in the primal objective function only, then the dual trick produces a MILP, that we can solve using standard solvers.