Day 26 Notes

Zach Neveu

June 19, 2019

1 Agenda

- Solving min-cost flow
- Cycle cancelling algorithm

2 Min-Cost Flow

- Max-Flow can determine if given flow is viable
- Goal once this is found is to change it to minimize cost.

3 Cycle-Cancelling Algorithm

- Use FF to find a max flow
- If it's too small, no flow exists
- While a net-negative cycle exists in the residual flow network, augment the flow around the cycle.
- When no negative-cycles exist, the flow has optimal cost.

4 Application of Network Flow

Baseball Elimination

- 4 baseball teams trying to finish the season in first place.
- Win-Records:
  - NY: 92 wins
  - Boston: 90 wins
  - Baltimore: 91 wins
  - Toronto: 91 wins
- Question: Can Boston get the most wins allowing for ties?

- Games remaining
  - NY-Ba
  - NY-To
  - Ba-To
  - Ba-Bo
  - To-Bo
- Manual Answer: Boston can't win: NY has to lose both of their games. This means that To and Ba each have one win, then they play each other so one of them will end up with 93 wins.
- Looking at fig. 1 this makes sense, as there are only 2 units going to the sink and 3 coming from the source. This isn't feasible.

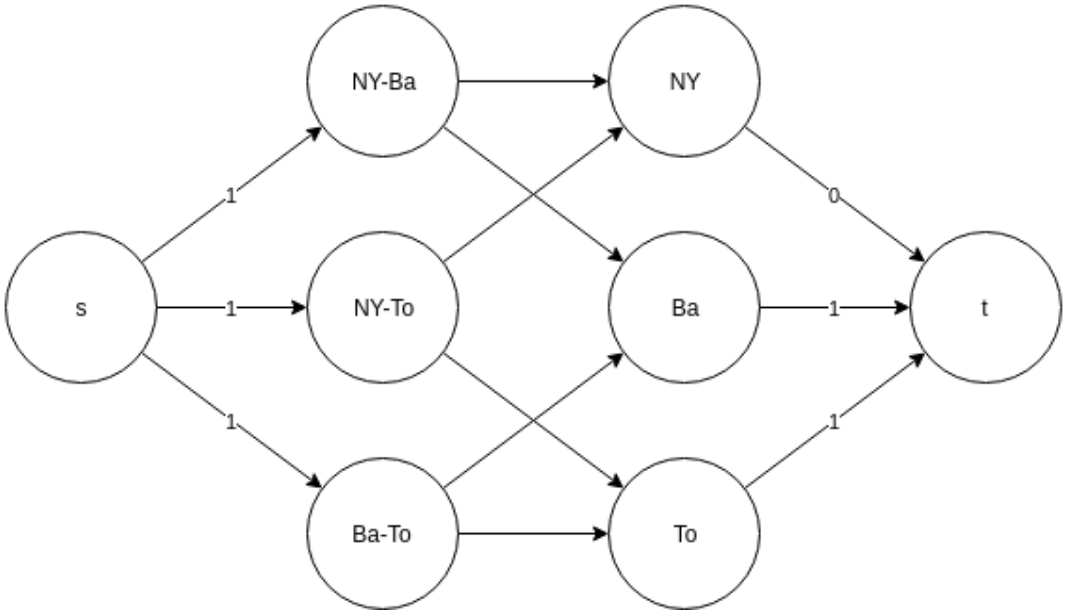

Figure 1: Baseball Games Flow Network

- Second (more complicated) example
- NY: 90, Ba: 88, To: 87, Bo: 79
- Remaining Games
  - Ny-Bo: x4
  - Ba-Bo: x4
  - To-Bo: x4
  - Ba-NY: x1
  - Ba-To: x1
  - NY-To: x6

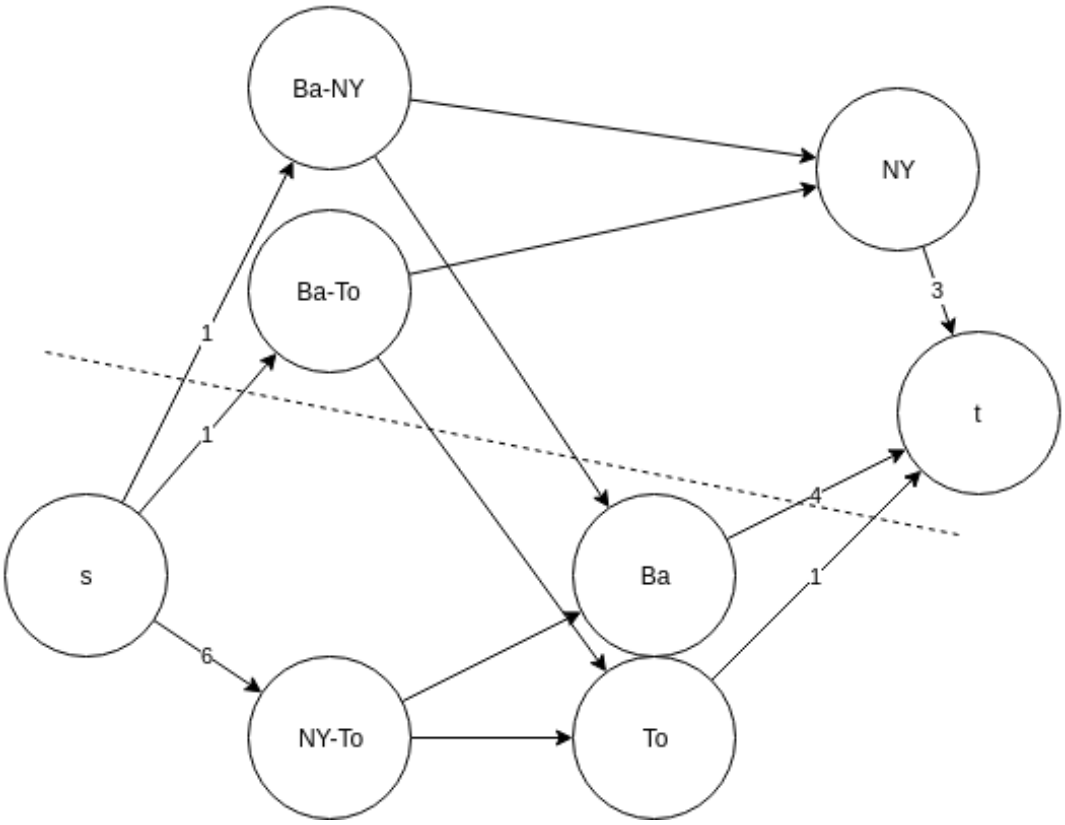

Figure 2: Harder Magic Number Problem

- Looking at fig. 2, the maximum cut is 7. Boston needs 8 games to win, so they cannot do it.

5 Hopping Airplane

- Goes to a bunch of sequential destinations, people get on and off at each destination.
- Every person starts at one city and ends at another.
- This makes $b[i, j]$ people who want to go from city i to city j (b)
- Each person pays a fair $f[i, j]$ to go from i to j (cost)
- Each plain has a capacity p
- Goal: maximize revenue
- Each city gets a node.

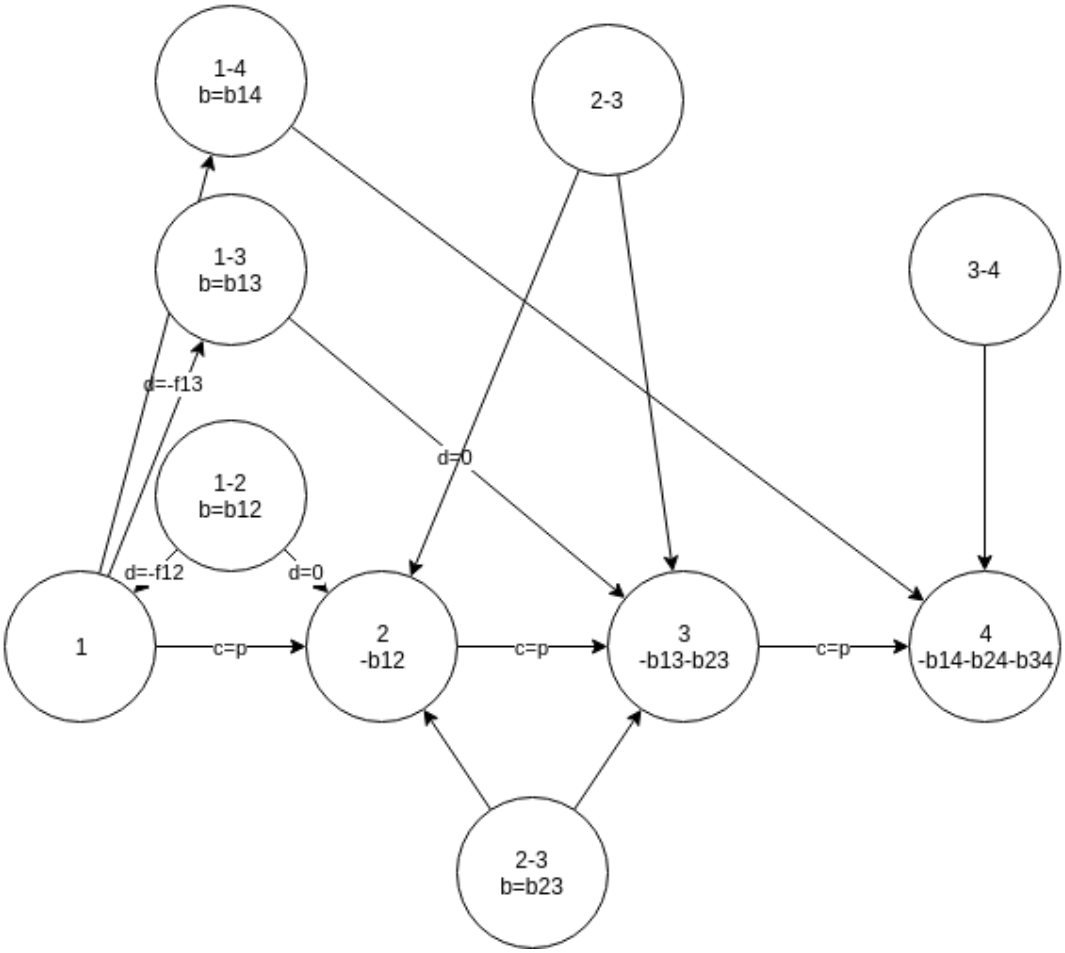

Figure 3: Hopping Airplane Example