CSEP 521: Network Flow and Linear Programming
Richard Anderson, March 9, 2021

Announcements
- Remaining lectures on Optimization
- Combinatorial Optimization
- Linear Programming
- Course Evaluation
  - You will have received a link
- Readings
  - Skim textbook chapters on Matching/Network Flow (CLRS or KT)
  - Skim textbook chapters on Linear Programming
  - DasGupta, Papadimitriou, and Vazarani
- Last homework is due Thursday, March 11
  - Notify instructor if any homework is going to be turned in after March 14

Optimization
- Solve a problem by expressing it as minimizing or maximizing a real valued function
- Examples:
  - Page layout
  - Allocation of industrial materials for a five year plan
  - Placement of on-line ads
  - Pricing of airline seats

Optimization
- Local improvement algorithms
  - Iteratively improve solution until a local maximum (or minimum) is reached
  - Prove that the maximum is a global maximum
- Duality
  - Pairs of problems that bound solutions
  - Finding the maximum for one problem finds the minimum for another problem

Network Flow Definitions
- Flowgraph: Directed graph with distinguished vertices $s$ (source) and $t$ (sink)
- Capacities on the edges, $c(e) \geq 0$
- Problem, assign flows $f(e)$ to the edges such that:
  - $0 \leq f(e) \leq c(e)$
  - Flow is conserved at vertices other than $s$ and $t$
  - Flow conservation: flow going into a vertex equals the flow going out
  - The flow leaving the source is a large as possible
Residual Graph

- Flow graph $G$, Residual Graph $G_R$
- $G$: edge $e$ from $u$ to $v$ with capacity $c$ and flow $f$
- $G_R$: edge $e'$ from $u$ to $v$ with capacity $c - f$
- $G_R$: edge $e''$ from $v$ to $u$ with capacity $f$
- Find a path from $s$ to $t$ in $G_R$ with minimum edge capacity (in $G_R$) of $b > 0$
- Add flow $b$ to the path from $s$ to $t$ in $G$

Ford-Fulkerson Algorithm (1956)

```
while not done
    Construct residual graph $G_R$
    Find an $s$-$t$ path $P$ in $G_R$ with capacity $b > 0$
    Add $b$ units along $P$ in $G$
```

If the sum of the capacities of edges leaving $S$ is at most $C$, then the algorithm takes at most $C$ iterations

Flow Example
Cuts in a graph

• Cut: Partition of V into disjoint sets S, T with s in S and t in T.
• Cap(S,T): sum of the capacities of edges from S to T
• Flow(S,T): net flow out of S
  • Sum of flows out of S minus sum of flows into S

• Flow(S,T) <= Cap(S,T)

Cap(S,T) and Flow(S,T)

S={s, a, b, e, h},  T = {c, f, i, d, g, t}

Cap(S,T) = 95,  Flow(S,T) = 80
– 15 = 65

Minimum value cut

MaxFlow – MinCut Theorem

• There exists a flow which has the same value as the minimum cut
• This shows that both the flow is maximum and the cut is minimum since the flow is always less than or equal to the cut
• The proof is to run the Ford-Fulkerson algorithm until it completes and look at the residual graph
• This is a "duality theorem" as the MaxFlow and MinCut problems are duals

History

• Ford / Fulkerson studied network flow in the context of the Soviet Rail Network

Ford Fulkerson Runtime

• Cost per phase times number of phases
• Phases
  • Capacity leaving source: C
  • Add at least one unit per phase
• Cost per phase
  • Build residual graph: O(m)
  • Find s-t path in residual: O(m)
Better methods of finding augmenting paths

- Find the maximum capacity augmenting path
  - $O(m \log(C))$ time algorithm for network flow
- Find the shortest augmenting path
  - $O(m^2)$ time algorithm for network flow
- Find a blocking flow in the residual graph
  - $O(mn \log n)$ time algorithm for network flow

Network Flow Applications

Converting Matching to Network Flow

Resource Allocation: Assignment of reviewers

- A set of papers $P_1, \ldots, P_n$
- A set of reviewers $R_1, \ldots, R_m$
- Paper $P_i$ requires $A_i$ reviewers
- Reviewer $R_j$ can review $B_j$ papers
- For each reviewer $R_j$, there is a list of paper $L_{j1}, \ldots, L_{jk}$ that $R_j$ is qualified to review

Image Segmentation

- Separate foreground from background

MinCost Network Flow Problem

- Directed graph with source and sink
- Each edge has a capacity $c(e)$ and a cost $d(e)$
- A total flow value $K$ is specified
- Find a flow function $f(e)$ on the edges such that
  - $0 \leq f(e) \leq c(e)$
  - Sum of the flows leaving the source is $K$
  - Flow is conserved at the vertices
  - The cost of the flow, $\sum f(e)d(e)$, is minimized
Applications of min cost flow

- Transportation problems — taking cost into account
  - An oil company is charged for using pipeline
- Allocation problems
  - Account for costs and profits
    - Showing internet ads
      - Certain number of ads of different types are available to show
      - Ads are required to reach certain demographics
      - Different profits associated with different users

Mincost Flow

\[ K = 7 \]

Solving MinCost Flow

- Circulation — a flow problem with no source or sink
  - Add an edge from \( t \) to \( s \) with capacity \( K \)
- Find a flow of size \( K \) (\( K \) units leaving \( s \))
- Build residual graph \( G_R \) (except for \( (t,s) \))
  - \( G \): edge \( e \) from \( u \) to \( v \) with capacity \( c \), cost \( d \), and flow \( f \)
  - \( G_R \): edge \( e' \) from \( u \) to \( v \) with capacity \( c - f \), cost \( d \)
  - \( G_R \): edge \( e'' \) from \( v \) to \( u \) with capacity \( f \), cost \( -d \)

Mincost flow algorithm

- Adding flow along a cycle preserves the conservation of flow
- With negative cost cycles the cost keeps decreasing
- We could prove that when this stops, it has an optimal solution
- Basic algorithm needs serious engineering to make it practical