CSE 421
Algorithms

Richard Anderson
Lecture 24
Network Flow Applications

Today's topics

- Problem Reductions
  - Undirected Flow to Flow
  - Bipartite Matching
  - Disjoint Path Problem
- Circulations
- Lowerbound constraints on flows
- Survey design

Problem Reduction

- Reduce Problem A to Problem B
  - Convert an instance of Problem A to an instance Problem B
  - Use a solution of Problem B to get a solution to Problem A
- Practical
  - Use a program for Problem B to solve Problem A
- Theoretical
  - Show that Problem B is at least as hard as Problem A

Problem Reduction Examples

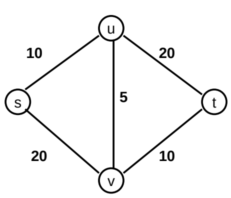
- Reduce the problem of finding the Maximum of a set of integers to finding the Minimum of a set of integers

Find the maximum of: 8, -3, 2, 12, 1, -6

Construct an equivalent minimization problem

Undirected Network Flow

- Undirected graph with edge capacities
- Flow may go either direction along the edges (subject to the capacity constraints)



Construct an equivalent flow problem

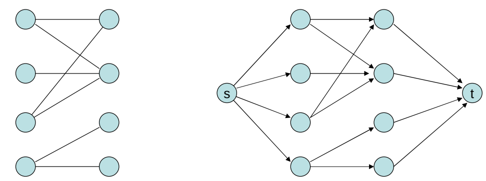
Bipartite Matching

- A graph G=(V,E) is bipartite if the vertices can be partitioned into disjoints sets X,Y
- A matching M is a subset of the edges that does not share any vertices
- Find a matching as large as possible

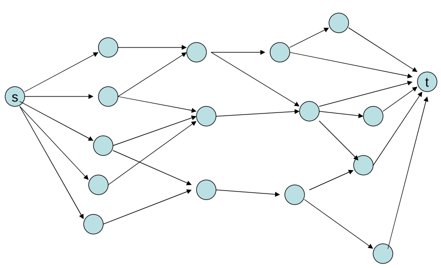
Application

- A collection of teachers
- A collection of courses
- And a graph showing which teachers can teach which courses

Converting Matching to Network Flow



Finding edge disjoint paths



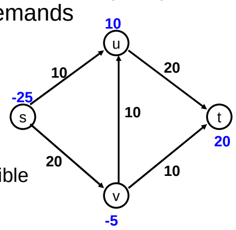
Construct a maximum cardinality set of edge disjoint paths

Theorem

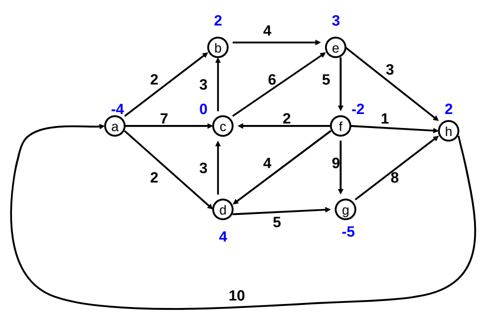
- The maximum number of edge disjoint paths equals the minimum number of edges whose removal separates s from t

Circulation Problem

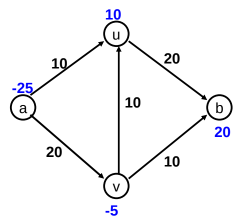
- Directed graph with capacities, c(e) on the edges, and demands d(v) on vertices
- Find a flow function that satisfies the capacity constraints and the vertex demands
  - $0 <= f(e) <= c(e)$
  - $f^{in}(v) – f^{out}(v) = d(v)$
- Circulation facts:
  - Feasibility problem
  - d(v) < 0: source; d(v) > 0: sink
  - Must have $\Sigma_v d(v)=0$ to be feasible



Find a circulation in the following graph



Reducing the circulation problem to Network flow

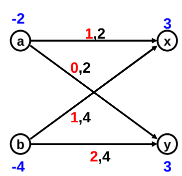


Formal reduction

- Add source node s, and sink node t
- For each node v, with d(v) < 0, add an edge from s to v with capacity -d(v)
- For each node v, with d(v) > 0, add an edge from v to t with capacity d(v)
- Find a maximum s-t flow. If this flow has size $\Sigma_v$cap(s,v) then the flow gives a circulation satisifying the demands

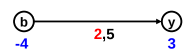
Circulations with lowerbounds on flows on edges

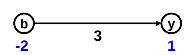
- Each edge has a lowerbound l(e).
  - The flow f must satisfy l(e) <= f(e) <= c(e)



Removing lowerbounds on edges

- Lowerbounds can be shifted to the demands





Formal reduction

- $L_{in}(v)$: sum of lowerbounds on incoming edges
- $L_{out}(v)$: sum of lowerbounds on outgoing edges
- Create new demands d' and capacities c' on vertices and edges
  - $d'(v) = d(v) + l_{out}(v) - l_{in}(v)$
  - $c'(e) = c(e) - l(e)$

Application

- Customized surveys
  - Ask customers about products
    - Only ask customers about products they use
    - Limited number of questions you can ask each customer
    - Need to ask a certain number of customers about each product
    - Information available about which products each customer has used

Details

- Customer $C_1, . . ., C_n$
- Products $P_1, . . ., P_m$
- S_i is the set of products used by C_i
- Customer C_i can be asked between c_i and c'_i questions
- Questions about product P_j must be asked on between p_j and p'_j surveys

Circulation construction

Today's topics

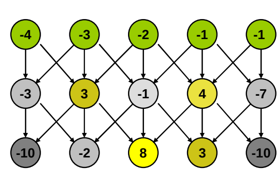
- Open Pit Mining Problem
- Task Selection Problem
- Reduction to Min Cut problem

S, T is a cut if S, T is a partition of the vertices with
s in S and t in T
The capacity of an S, T cut is the sum of the capacities of
all edges going from S to T

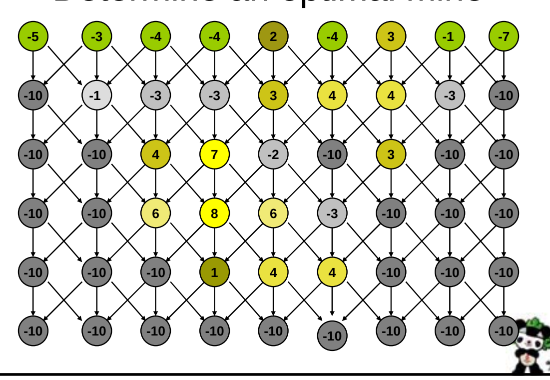
Open Pit Mining

- Each unit of earth has a profit (possibly negative)
- Getting to the ore below the surface requires removing the dirt above
- Test drilling gives reasonable estimates of costs
- Plan an optimal mining operation

Mine Graph

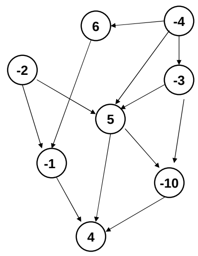


Determine an optimal mine



Generalization

- Precedence graph G=(V,E)
- Each v in V has a profit p(v)
- A set F if *feasible* if when w in F, and (v,w) in E, then v in F.
- Find a feasible set to maximize the profit

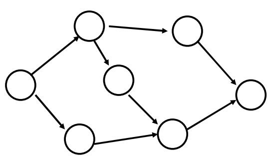


Min cut algorithm for profit maximization

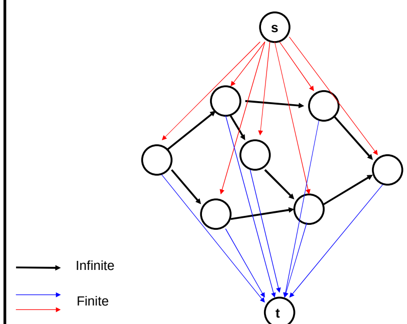
- Construct a flow graph where the minimum cut identifies a feasible set that maximizes profit

Precedence graph construction

- Precedence graph G=(V,E)
- Each edge in E has infinite capacity
- Add vertices s, t
- Each vertex in V is attached to s and t with finite capacity edges

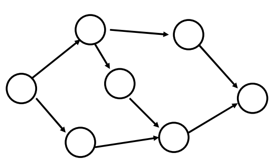


Show a finite value cut with at least two vertices on each side of the cut



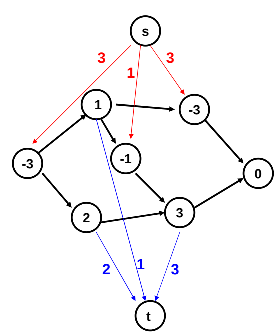
The sink side of a finite cut is a feasible set

- No edges permitted from S to T
- If a vertex is in T, all of its ancestors are in T

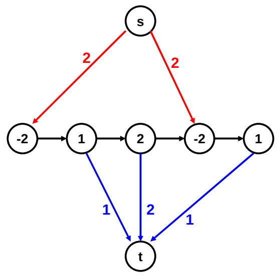


Setting the costs

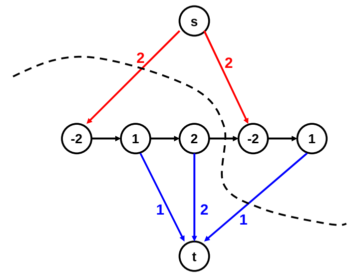
- If p(v) > 0,
  - cap(v,t) = p(v)
  - cap(s,v) = 0
- If p(v) < 0
  - cap(s,v) = -p(v)
  - cap(v,t) = 0
- If p(v) = 0
  - cap(s,v) = 0
  - cap(v,t) = 0



Enumerate all finite s,t cuts and show their capacities



Minimum cut gives optimal solution Why?

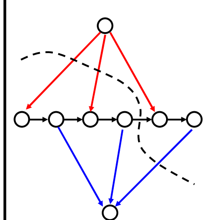


Computing the Profit

- Cost(W) = $\Sigma_{\{w \text{ in } W;\ p(w) < 0\}} -p(w)$
- Benefit(W) = $\Sigma_{\{w \text{ in } W;\ p(w) > 0\}}\ p(w)$
- Profit(W) = Benefit(W) – Cost(W)

- Maximum cost and benefit
  - C = Cost(V)
  - B = Benefit(V)

Express Cap(S,T) in terms of B, C, Cost(T), Benefit(T), and Profit(T)



Summary

- Construct flow graph
  - Infinite capacity for precedence edges
  - Capacities to source/sink based on cost/benefit
- Finite cut gives a feasible set of tasks
- Minimizing the cut corresponds to maximizing the profit
- Find minimum cut with a network flow algorithm

Today's topics

- More network flow reductions
  - Airplane scheduling
  - Image segmentation
  - Baseball elimination

Airplane Scheduling

- Given an airline schedule, and starting locations for the planes, is it possible to use a fixed set of planes to satisfy the schedule.
- Schedule
  - [segments] Departure, arrival pairs (cities and times)
- Approach
  - Construct a circulation problem where paths of flow give segments flown by each plane

Example

- Seattle->San Francisco, 9:00 – 11:00
- Seattle->Denver, 8:00 – 11:00
- San Francisco -> Los Angeles, 13:00 – 14:00
- Salt Lake City -> Los Angeles, 15:00-17:00
- San Diego -> Seattle, 17:30-> 20:00
- Los Angeles -> Seattle, 18:00->20:00
- Flight times:
  - Denver->Salt Lake City, 2 hours
  - Los Angeles->San Diego, 1 hour

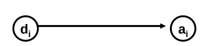
Can this schedule be full filled with two planes, starting from Seattle?

Compatible segments

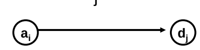
- Segments S_1 and S_2 are compatible if the same plane can be used on S_1 and S_2
  - End of S_1 equals start of S_2, and enough time for turn around between arrival and departure times
  - End of S_1 is different from S_2, but there is enough time to fly between cities

Graph representation

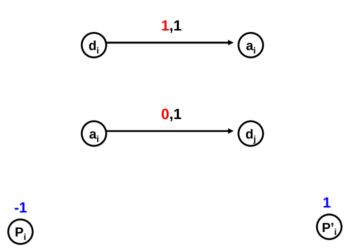
- Each segment, S_i, is represented as a pair of vertices (d_i, a_i, for departure and arrival), with an edge between them.



- Add an edge between a_i and d_j if S_i is compatible with S_j.



Setting up a flow problem



Result

- The planes can satisfy the schedule iff there is a feasible circulation

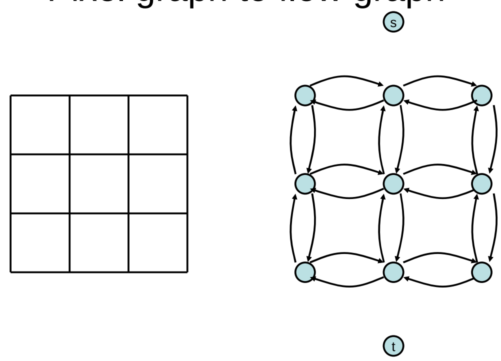
Image Segmentation

- Separate foreground from background

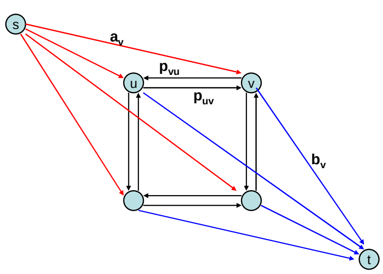
Image analysis

- a_i: value of assigning pixel i to the foreground
- b_i: value of assigning pixel i to the background
- p_{ij}: penalty for assigning i to the foreground, j to the background or vice versa
- A: foreground, B: background
- $Q(A,B) = \Sigma_{\{i \text{ in } A\}} a_i + \Sigma_{\{j \text{ in } B\}} b_j - \Sigma_{\{(i,j) \text{ in } E,\ i \text{ in } A,\ j \text{ in } B\}} p_{ij}$

Pixel graph to flow graph



Mincut Construction



Baseball elimination

- Can the Dinosaurs win the league?
- Remaining games:
  - AB, AC, AD, AD, AD, BC, BC, BC, BD, CD

	W	L
Ants	4	2
Bees	4	2
Cockroaches	3	3
Dinosaurs	1	5

A team wins the league if it has strictly more wins than any other team at the end of the season
A team ties for first place if no team has more wins, and there is some other team with the same number of wins

Baseball elimination

- Can the Fruit Flies win the league?
- Remaining games:
  - AC, AD, AD, AD, AF, BC, BC, BC, BC, BC, BD, BE, BE, BE, BE, BF, CE, CE, CE, CF, CF, DE, DF, EF, EF

	W	L
Ants	17	12
Bees	16	7
Cockroaches	16	7
Dinosaurs	14	13
Earthworms	14	10
Fruit Flies	12	15

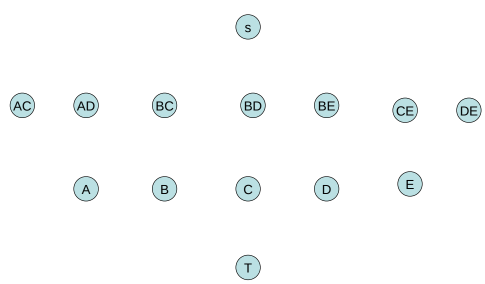
Assume Fruit Flies win remaining games

- Fruit Flies are tied for first place if no team wins more than 19 games
- Allowable wins
  - Ants (2)
  - Bees (3)
  - Cockroaches (3)
  - Dinosaurs (5)
  - Earthworms (5)
- 18 games to play
  - AC, AD, AD, AD, BC, BC, BC, BC, BC, BD, BE, BE, BE, BE, CE, CE, CE, DE

	W	L
Ants	17	13
Bees	16	8
Cockroaches	16	9
Dinosaurs	14	14
Earthworms	14	12
Fruit Flies	19	15

Remaining games

AC, AD, AD, AD, BC, BC, BC, BC, BC, BD, BE, BE, BE, BE, CE, CE, CE, DE



Network flow applications summary

- Bipartite Matching
- Disjoint Paths
- Airline Scheduling
- Survey Design
- Baseball Elimination
- Project Selection
- Image Segmentation