COMPSCI 311: Introduction to Algorithms

Lecture 18: Network Flow Applications

Marius Minea

University of Massachusetts Amherst

slides credit: Dan Sheldon (adapted)

Review: Network Flow

- Residual graph G_f.
  Capacities: $c(e) - f(e)$ forward, $f(e) backward.
- Ford-Fulkerson

  Initialize flow f to all zeros
  ▷ Augment flow as long as it is possible
  while there exists an s-t path P in G_f **do**
  f = Augment(f, P)
  update residual graph G_f
  end while
- Analysis
  - Always maintain a flow: use facts of residual graph and augment operation, verify that definition of flow still holds
  - Termination and running time: flow increases at least one in each iteration, and cannot exceed total capacity leaving s
  - Correctness: Max-Flow Min-Cut Theorem

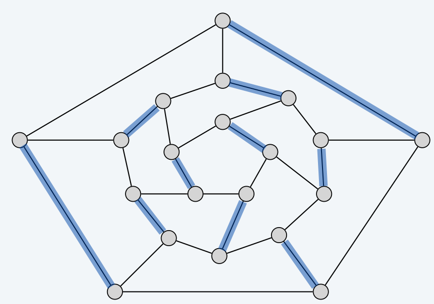
Max-Flow Min-Cut Theorem

- $v(f) \leq c(A, B)$ for any flow f and any s-t cut $c(A, B)$
- On termination, Ford-Fulkerson produces a max-flow f and min-cut (A, B)
- A = set of nodes reachable from s in residual graph to find the min-cut
- Complexity
  - $O(mnC_{\max})$ for basic version
  - $O(m^2 \log C_{\max})$ for capacity scaling
  - Capacity-independent for choosing shortest augmenting paths
    $O(m^2 n)$ Edmonds-Karp, $O(n^2 m)$ Dinitz

Matching

Def. Given an undirected graph $G = (V, E)$, subset of edges $M \subseteq E$ is a matching if each node appears in at most one edge in M.

Max matching. Given a graph G, find a max-cardinality matching.



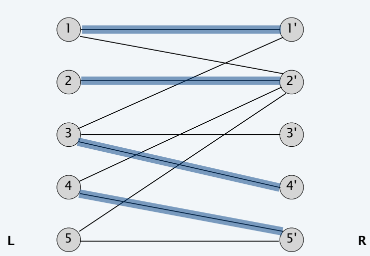
6

slide credit: Kevin Wayne / Pearson

Bipartite matching

Def. A graph G is bipartite if the nodes can be partitioned into two subsets L and R such that every edge connects a node in L with a node in R.

Bipartite matching. Given a bipartite graph $G = (L \cup R, E)$, find a max-cardinality matching.



matching: 1–1', 2–2', 3–4', 4–5'

7

slide credit: Kevin Wayne / Pearson

Formulating Matching as Network Flow problem

- **Goal**: given matching instance $G = (L \cup R, E)$:
  - create a flow network G',
  - find a maximum flow f in G'
  - use f to construct a maximum matching M in G.
- Standard approach for reducing a problem to network flow
- Intuition?

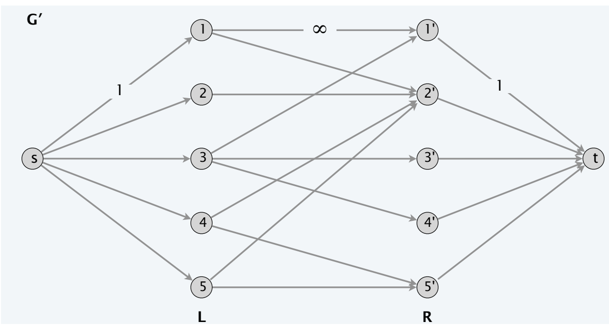
Connect left set L to source, right set R to sink.

Larger matchings should have larger flows.

How to select capacities?

Maximal Matching as Network Flow

- Add a source s and sink t
- For each edge $(u, v) \in E$, add $u \to v$ (directed), capacity 1
- Add an edge with capacity 1 from s to each node $u \in L$
- Add an edge with capacity 1 from each node $v \in R$ to t.



- Does it matter if we have unit or infinite capacities from L to

Clicker Question 1

Suppose I want to work with ∞ values (for shortest path, minimum cost, maximum flow, etc.) What properties would I need to add and compare values?

A: $x \neq \infty \to x < \infty$

B: $\infty + \infty = \infty$

C: Both A and B

D: A and something stronger than B

Maximal Matching: Analysis

- Run F-F to get an integral max-flow f
- Set M to the set of edges from L to R with flow $f(e) = 1$
- Claim: The set M is a maximum matching.

Let's prove that:

1. Integer flow f in G' $\implies$ matching M in G with $|M| = v(f)$
2. Matching M in G $\implies$ flow f in G' with $v(f) = |M|$

Therefore, max-flow f in G' $\iff$ maximum matching M in G

Proof of 1: given f, construct M

- M = edges from L to R carrying one unit of flow
- Capacity constraints $\Rightarrow$ at most 1 unit of flow leaving $u \in L$
- Edge flows are 0 or 1 $\Rightarrow$ M has at most one edge incident to u.
- Similar argument for $v \in R$

Maximal Matching: Analysis

Proof of 2: given M, construct f

- Set $f(e) = 1$ if $e \in M$
- Send one unit of flow from s to $u \in L$ if u is matched
- Send one unit of flow from $v \in R$ to t if t is matched
- All other edge flows values are zero
- Verify that capacity and flow conservation constraints are satisfied, and that $v(f) = |M|$.

Network flow II: quiz 1

What is running time of Ford–Fulkerson algorithms to find a maximum matching in a bipartite graph with $|L| = |R| = n$?

A. $O(m + n)$

B. $O(mn)$

C. $O(mn^2)$

D. $O(m^2 n)$

Perfect matchings in bipartite graphs

Def. Given a graph $G = (V, E)$, a subset of edges $M \subseteq E$ is a perfect matching if each node appears in exactly one edge in M.

Q. When does a bipartite graph have a perfect matching?

Structure of bipartite graphs with perfect matchings.

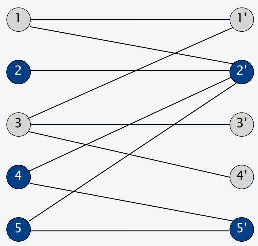
- Clearly, we must have $|L| = |R|$.
- Which other conditions are necessary?
- Which other conditions are sufficient?

Perfect matchings in bipartite graphs

Notation. Let S be a subset of nodes, and let $N(S)$ be the set of nodes adjacent to nodes in S.

Observation. If a bipartite graph $G = (L \cup R, E)$ has a perfect matching, then $|N(S)| \geq |S|$ for all subsets $S \subseteq L$.
Pf. Each node in S has to be matched to a different node in $N(S)$. ▪

S = { 2, 4, 5 }
N(S) = { 2', 5' }



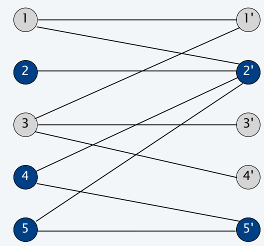
no perfect matching

Hall's marriage theorem

Theorem. [Frobenius 1917, Hall 1935] Let $G = (L \cup R, E)$ be a bipartite graph with $|L| = |R|$. Then, graph G has a perfect matching iff $|N(S)| \geq |S|$ for all subsets $S \subseteq L$.

Pf. ⇒ This was the previous observation.

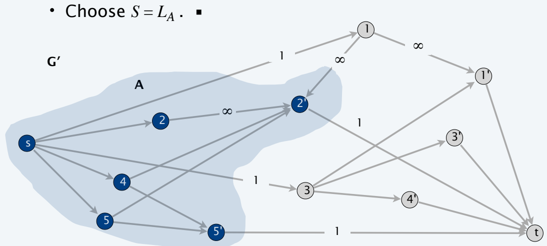
S = { 2, 4, 5 }
N(S) = { 2', 5' }



no perfect matching

Hall's marriage theorem

Pf. ⇐ Suppose G does not have a perfect matching.
- Formulate as a max-flow problem and let (A, B) be a min cut in G'.
- By max-flow min-cut theorem, $cap(A, B) < |L|$.
- Define $L_A = L \cap A$, $L_B = L \cap B$, $R_A = R \cap A$.
- $cap(A, B) = |L_B| + |R_A| \Rightarrow |R_A| < |L_A|$.
- Min cut can't use ∞ edges $\Rightarrow N(L_A) \subseteq R_A$.
- $|N(L_A)| \leq |R_A| < |L_A|$.
- Choose $S = L_A$. ▪



L_A = {2, 4, 5}
L_B = {1, 3}
R_A = {2', 5'}
$N(L_A)$ = {2', 5'}

Network flow II: quiz 2

Which of the following are properties of the graph G = (V, E)?

A. G has a perfect matching.

B. Hall's condition is satisfied: $|N(S)| \geq |S|$ for all subsets $S \subseteq V$.

C. Both A and B.

D. Neither A nor B.

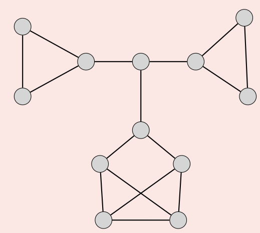


Image segmentation

Image segmentation.
- Divide image into coherent regions.
- Central problem in image processing.

Ex. Separate human and robot from background scene.

Grabcut image segmentation

Grabcut. [Rother–Kolmogorov–Blake 2004]

"GrabCut" — Interactive Foreground Extraction using Iterated Graph Cuts

Carsten Rother* Vladimir Kolmogorov† Andrew Blake‡
Microsoft Research Cambridge, UK

Figure 1: **Three examples of GrabCut.** The user drags a rectangle loosely around an object. The object is then extracted automatically.

Bokeh Effect: Blurring Background

- Using an expensive camera and appropriate lenses, you can get a “bokeh" effect on portrait photos: the background is blurred and the foreground is in focus.

- Can fake effect using cheap phone cameras and appropriate software

Formulating the Problem

Given set V of pixels, classify each as foreground or background. Assume you have:

- Likelihood that a pixel is in foreground (a_i) / background (b_i)
- Numeric penalty p_{ij} for assigning neighboring pixels i and j to different classes

Graph edges E: for each pixel, edge to neighbors (4? 8? other?)

Criteria:

- Accuracy if $a_i > b_i$, would prefer to label pixel i as foreground
- Smoothness: if many neighbors are labeled the same (foreground), would like to label pixel i as foreground (minimize penalties)

Image Segmentation as Network Flow

Maximize correct labeling scores, minimize penalties

Let A: set of pixels labeled foreground, B: pixels in background

Maximize: $\sum_{i \in A} a_i + \sum_{j \in B} b_j - \sum_{(i,j) \in E, i \in A, j \in B} p_{ij}$

Insight: (A, B) is a partition $\Rightarrow$ forms a **cut**

First sum is $\sum_{i \in V}(a_i + b_i) - \sum_{i \in A} b_i - \sum_{j \in B} a_j$
(constant minus “penalties” for mislabeling)

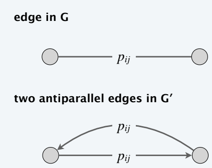
Must **minimize** $\sum_{i \in A} b_i + \sum_{j \in B} a_j + \sum_{(i,j) \in E, i \in A, j \in B} p_{ij}$

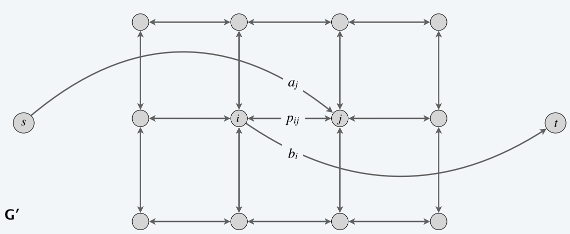
$\Rightarrow$ find **minimum cut**

Image segmentation

Formulate as min-cut problem $G' = (V', E')$.

- Include node for each pixel.
- Use two antiparallel edges instead of undirected edge.
- Add source s to correspond to foreground.
- Add sink t to correspond to background.





slide credit: Kevin Wayne / Pearson

Image segmentation

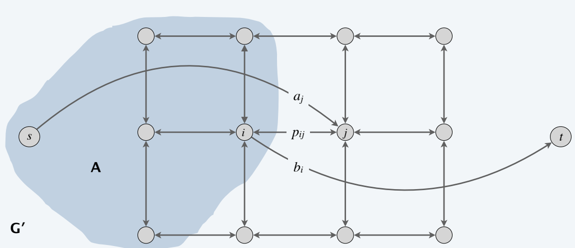
Consider min cut (A, B) in G'.

- A = foreground.

$$cap(A,B) = \sum_{j \in B} a_j + \sum_{i \in A} b_i + \sum_{\substack{(i,j) \in E \\ i \in A,\, j \in B}} p_{ij}$$

if i and j on different sides, p_{ij} counted exactly once

- Precisely the quantity we want to minimize.



slide credit: Kevin Wayne / Pearson

More Network Flows

- Extensions
  - Multiple sources and sinks
  - Circulations with supplies and demands
  - Flows with lower bounds
- Improved Algorithms: Preflow-push $O(n^3)$
- Applications
  - Network connectivity
  - Data mining: survey design
  - Airline scheduling
  - Baseball elimination
  - Multi-camera placement / scene reconstruction