COT 6936: Topics in Algorithms

Giri Narasimhan

ECS 254A / EC 2443; Phone: x3748

giri@cs.fiu.edu

http://www.cs.fiu.edu/~giri/teach/COT6936_S12.html

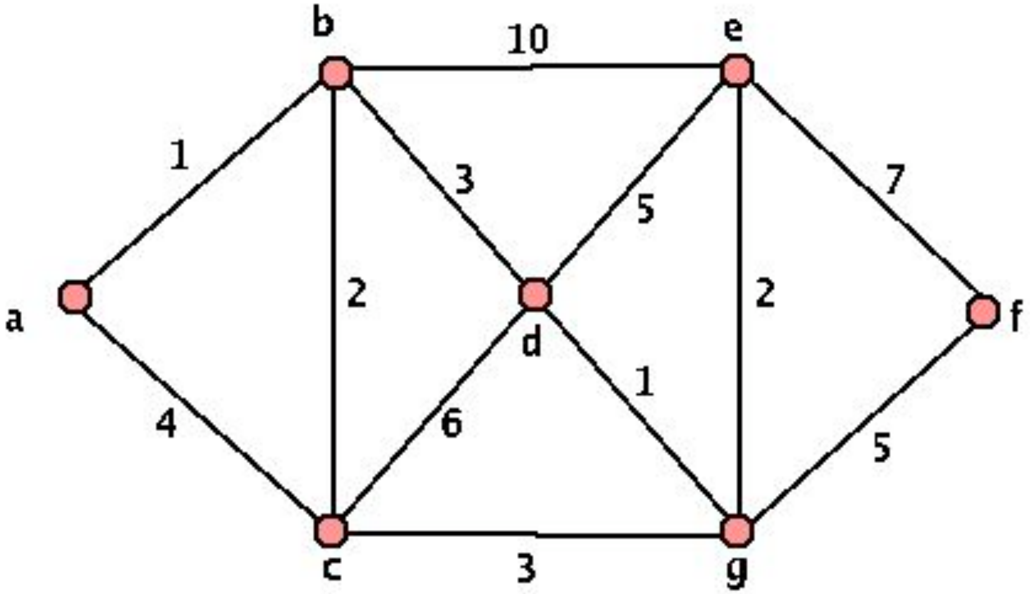
https://moodle.cis.fiu.edu/v2.1/course/view.php?id=**174**

COT 6936: Topics in Algorithms

Randomized Algorithms

Cut-Sets & Min-Cuts

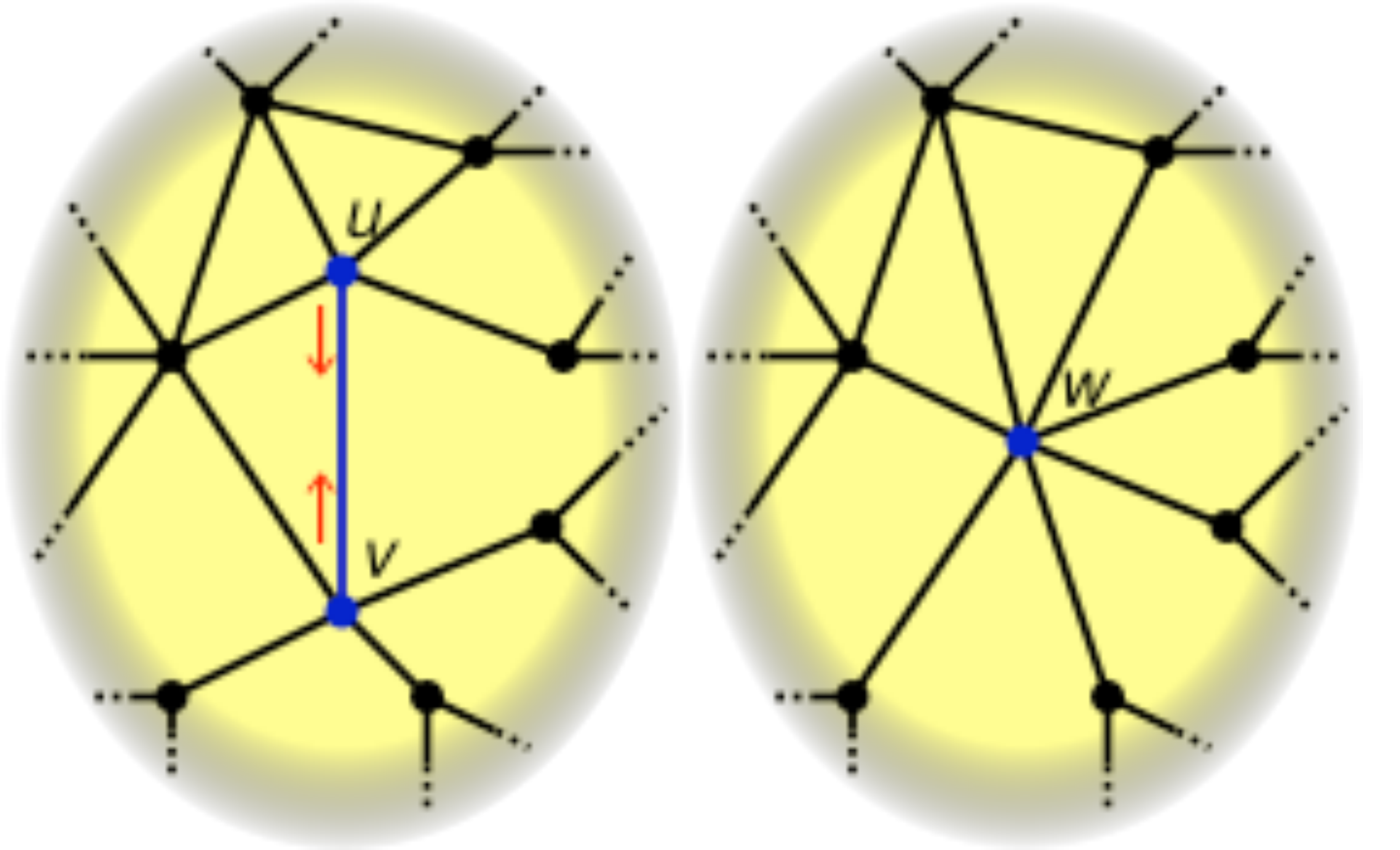
- Example 1: ({a,b,c,d}, {e,f,g})
  - Weight = 19
- Example 2: ({a,b,g}, {c,d,e,f})
  - Weight = 30
- Example 3: ({a}, {b,c,d,e,f,g})
  - Weight = 5



How is this different from Min-Cuts we considered in the context of Networks Flows?

- Undirected graphs
- No source or sink vertex
- "Robustness" parameter
- Global Min-Cut can be computed in poly time

Edge Contraction [Karger, 1992]

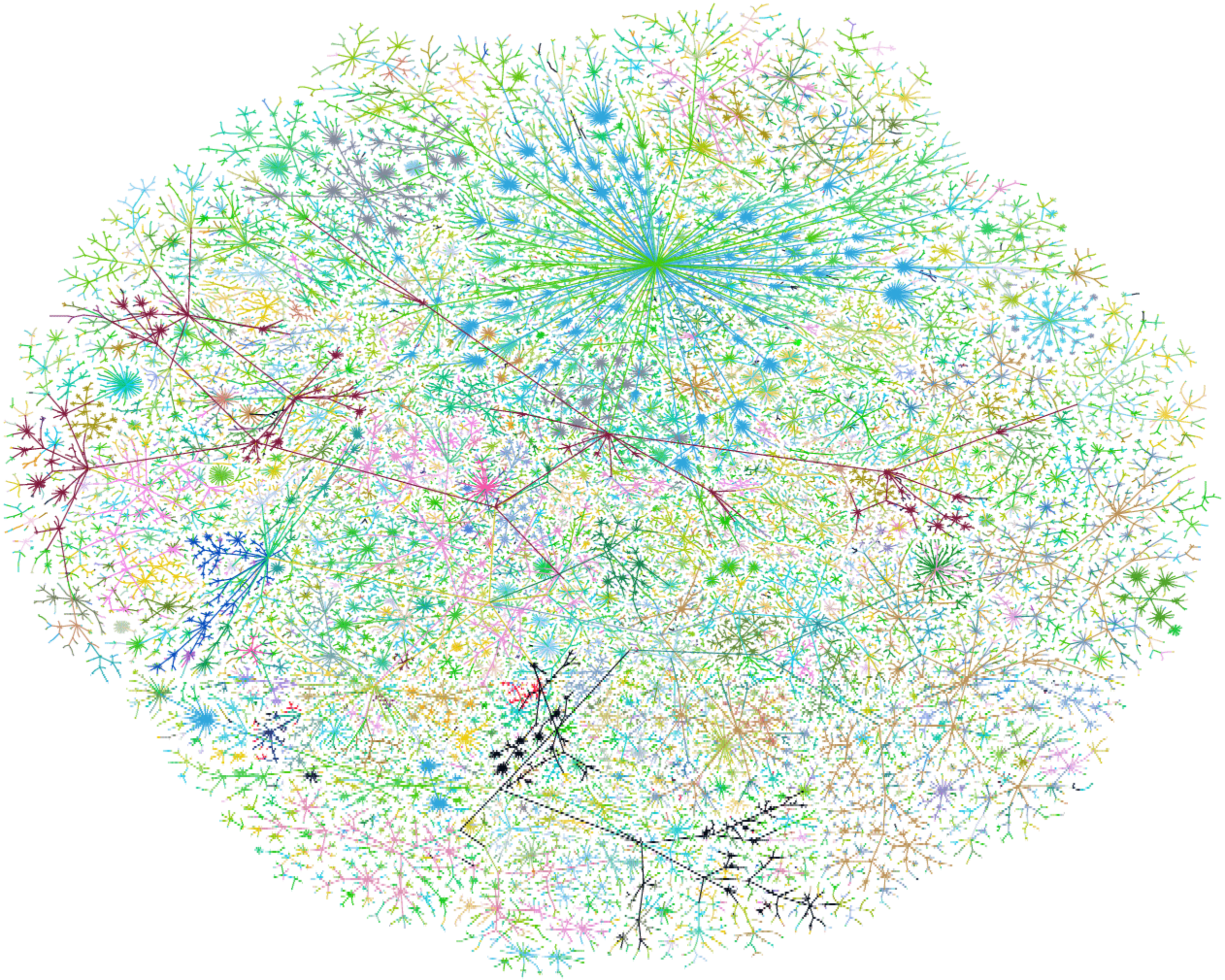


http://en.wikipedia.org/wiki/Edge_contraction

Edge Contractions and Min-Cuts

- Lemma: If you are not contracting an edge from the cut-set, edge contractions do not affect the size of min-cuts.
- Observation: Most edges are not part of the min-cut.
- Idea: Use randomization [Karger, 1992]

Min-Cuts in the Internet Graph



June 1999 Internet graph, Bill Cheswick
http://research.lumeta.com/ches/map/gallery/index.html

Randomized Algorithms: Min-Cut

- Algorithm:
  - Pick a random edge and contract it until only 2 vertices are remaining.
  - Report edges connecting the 2 remaining vertices as the min cut
- Analysis
  - Assume that the Min-cut is of size k
  - Prob {edge is not in Min-cut} $\geq 1 - 2/n$ (why?)
  - Prob {Min-cut is output} $\geq 2/n(n - 1)$ (why?)

Analysis: Min-Cut Algorithm (Cont'd)

- Observation:
  - If Min-Cut is of size k, then minimum degree of every vertex is k. (Why?)
- Number of edges in graph $\geq kn/2$
- Probability that <u>an</u> edge from Min-Cut is picked in iteration 1 is $\leq 2/n$
- Probability that <u>no</u> edge from Min-Cut is picked in iteration 1 is $\geq 1 - 2/n$
- Iteration i?

Analysis: Min-Cut Algorithm (Cont'd)

- E_i = Event that no edge from Min-Cut is picked in iteration i
- F_i = Event that no edge from Min-Cut is picked in iteration 1 through i

$$Pr(E_i|F_{i-1}) \geq 1 - \frac{k}{k(n-i+1)/2} = 1 - \frac{2}{n-i+1}.$$

- Need F_{n-2}!

Analysis: Min-Cut Algorithm (Cont'd)

$$\begin{aligned}
Pr(F_{n-2}) &= Pr(E_{n-2} \cap F_{n-3}) = Pr(E_{n-2}|F_{n-3})Pr(F_{n-3}) \\
&= Pr(E_{n-2}|F_{n-3}) \cdot Pr(E_{n-3}|F_{n-4}) \ldots Pr(E_2|F_1)Pr(F_1) \\
&\geq \Pi_{i=1}^{n-2}\left(1 - \frac{2}{n-i+1}\right) = \Pi_{i=1}^{n-2}\frac{n-i-1}{n-I+1} \\
&= \left(\frac{n-2}{n}\right)\left(\frac{n-3}{n-1}\right)\cdots\frac{4}{6}\frac{3}{5}\frac{2}{4}\frac{1}{3} \\
&= \frac{2}{n(n-1)}.
\end{aligned}$$

Analysis: Min-Cut Algorithm (Cont'd)

- Probability of contracting only edges not from Min-Cut, i.e., ending up with exactly the Min-Cut ≥ 2/n(n-1)
  - Rather low!
- Repeat the algorithm many times.
  - How many times?
  - Goal: repeat until prob of error is very small

$$\left(1 - \frac{2}{n(n-1)}\right)^{n(n-1)\ln n} \leq e^{-2\ln n} = \frac{1}{n^2}$$

Monte Carlo vs Las Vegas

- Monte Carlo algorithms:
  - Bounded run time in worst case
  - sometimes incorrect, but with bounded probability
    - One-sided versus two-sided errors
- Las Vegas algorithms:
  - always correct,
  - variable run times, but bounded expected time

Balls and Bins

- Balls and Bins Model
  - Throw m balls into n bins
  - Location of each ball chosen independently and uniformly at random

Balls and Bins

- Interesting questions to ask
  - How many balls in a bin on the average? m/n
  - How many bins are empty? $e^{m/n}$
  - How many balls in the fullest bin? $\Theta(\ln n/\ln \ln n)$
  - If m=n, how many bins are expected to have > 1 ball in it?
- Applications
  - Chain Hashing
  - Bucket Sort
  - Hash Table for passwords (reject if entry occupied)
  - Bloom Filters
  - Birthday Paradox

Birthday Paradox

Probability that m balls are put in distinct bins is

$$\left(1-\frac{1}{n}\right)\left(1-\frac{2}{n}\right)\cdots\left(1-\frac{m-1}{n}\right)=\prod_{j=1}^{m-1}\left(1-\frac{j}{n}\right).$$

Birthday Paradox

- To achieve probability ≥ ½, we need:
  - $m^2/2n \geq \ln 2$
  - $m \geq \text{sqrt}\{2n \ln 2\}$
- In a room with at least 23 people, the probability that at least two people have the same birthday is more than ½.

Average Size of a Chain in Hash Table

- Let N = # of possible hash values
- Let k = # items stored in the hash table
- Probability that exactly i out of k items hash to the same value is

$$p_i = \binom{k}{i} (N-1)^{k-i} N^{-k}.$$

Average Search Time

Unsuccessful Search:

$$
\begin{aligned}
A &= \sum_i (i+1)p_i = \sum_i \binom{k}{i}(i+1)(N-1)^{k-i}N^{-k} \\
&= \sum_i \binom{k}{i} i(N-1)^{k-i}N^{-k} + \sum_i \binom{k}{i}(N-1)^{k-i}N^{-k} \\
&= \sum_i k\binom{k-1}{i-1}(N-1)^{k-i}N^{-k} + 1 \\
&= kN^{-k}\sum_i \binom{k-1}{i}(N-1)^{k-i-1} + 1 \\
&= kN^{-k}N^{k-1} + 1 = 1 + k/N
\end{aligned}
$$

Successful Search:

$$
A' = \sum_{i,j} j q_{ij} = 1 + \frac{k-1}{2N}
$$

Maximum Load

- Prob that a bin has at least j balls is

$$\binom{n}{j}\left(\frac{1}{n}\right)^j \leq \frac{1}{j!} \leq \left(\frac{e}{j}\right)^j$$

Maximum Load: most balls in any bin

- Prob that one of n bins has at least j = (3 ln n / ln ln n) balls is

$$
\begin{aligned}
n\left(\frac{e}{j}\right)^j &\le n\left(\frac{e\ln\ln n}{3\ln n}\right)^{3\ln n/\ln\ln n} \\
&\le n\left(\frac{\ln\ln n}{3\ln n}\right)^{3\ln n/\ln\ln n} \\
&= e^{\ln n}\left(e^{\ln\ln\ln n-\ln\ln n}\right)^{3\ln n/\ln\ln n} \\
&= e^{-2\ln n+3(\ln n)(\ln\ln\ln n)/\ln\ln n} \\
&\le \frac{1}{n}
\end{aligned}
$$

Power of Two Choices

- Hashing with **two** hash functions
  - Dramatically reduces the expected size of the largest bin while doubling the average search cost.
- Dynamic Resource Allocation
  - Multiple identical resources to choose from
    - Find load of each one and pick least loaded
    - Pick random resource
    - Sample 2 random resources and pick less loaded one

Contention Resolution

- N processes $P_1, \ldots, P_N$ each competing for access to a single resource (shared database, shared communication channel, …)
- Time is divided into rounds
- If more than one process attempts to access resource, then all processes are locked out
- No communication between processes
- Need fair algorithm for large N
- Use randomization to break symmetry

Breaking Symmetry

- If small N, then assign round t mod N to process t. Not scalable!
- If large N, then each process attempts to access the resource in round t with prob p.
- Results:
  - To maximize prob of success, set $p = 1/n$
  - Prob of failure after en rounds is $\leq$ constant
  - Whp all N processes can access the resource in $t = 2en \ln n$ rounds

Breaking symmetry

- Many users want to share a resource
  - Want to pick a permutation quickly
  - Hash to 2^b bits and sort them
  - If $b = 3\log_2 n$ then two users will have distinct hash values with probability $1-1/n$

Randomized Algorithm for MAX 3-SAT

- Assume each clause has 3 distinct literals
- Randomly assign 0/1 to all variables
  - Each clause is satisfied with prob 7/8
  - Expected number of clauses = 7m/8
  - There exists a truth assignment that satisfies 7m/8 clauses

Bloom Filters

- Used to test set membership by using bit arrays to indicate which positions have been hashed to.

0	0	1	1	0	1	0	0	1

- Use k hash functions instead of 1.
- How large should k be for given error bound?

Applications

- Hashing with 2-way chaining
  - 2 hash function applied to each data item
  - Item inserted in shorter of two chains
- Dynamic Resource Allocation
  - Choosing a server among servers in a network
  - Choosing a disk to store an entity
  - Choosing a printer to serve a print job

Power of Two Choices

- Each ball comes with $d = 2$ possible bins, each chosen independently at random
- Ball is placed in the least full bin among the d choices
  - ties broken arbitrarily
- MAGICALLY, with high prob:
  - MAX LOAD = $\ln \ln n / \ln 2 + O(1)$
  - Down from $\Theta(\ln n / \ln \ln n)$ (when $d = 1$)
  - In general, when $d \geq 2$,
    - MAX LOAD = $\ln \ln n / \ln d + \Theta(1)$