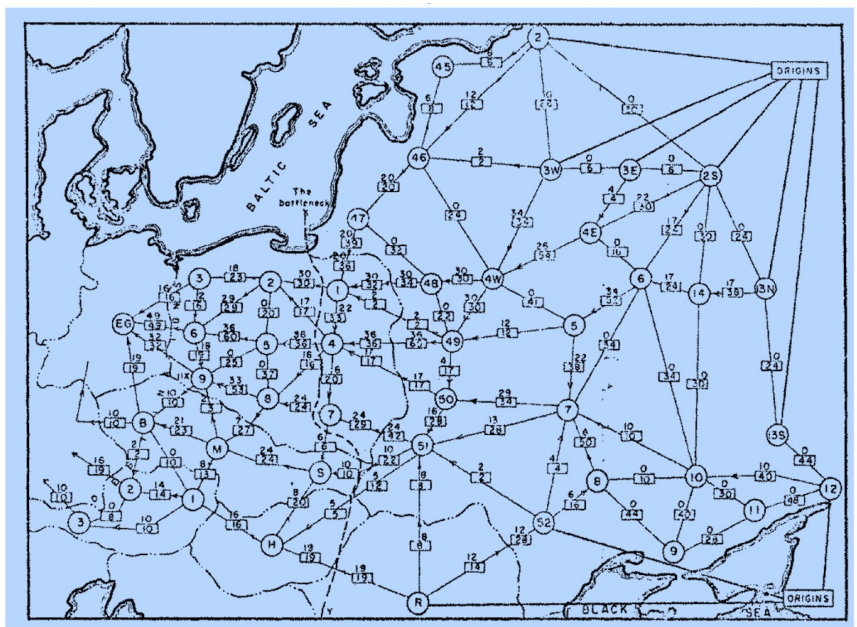
Probabilistic algorithms

AA-GEI FIB, UPC

Spring 2025-2026

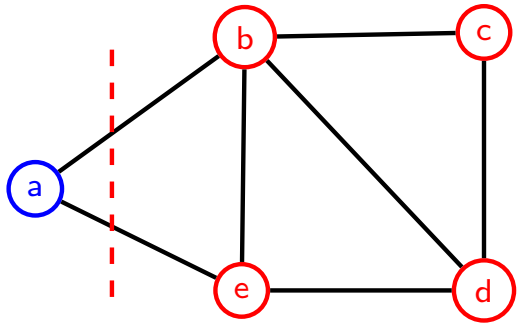
The Minimum Cut problem

In the mid 50's Harris and Ross studied the railway links between cities in the URSS and easter Europe and determined the easiest way to break the network by removing edges. The minimum cut of the graph.



The Minimum Cut problem

Given an undirected graph $G = (V, E)$ a cut is a partition of V in S and $\bar{S}$. The capacity of the cut is the number of edges with an end in S and the other in $\bar{S}$. The min cut is the cut with minimum capacity.



Complexity for deterministic algorithms

- Using Ford-Fulkerson: Max Flow-Min-Cut $O(n^2 m)$ or $O(nm)$ using J.Orlin's algorithms from 2013.
- Stoer-Wagner's algorithm (1994) $O(nm + n^2 \lg n)$ (non-flow, weighted graphs)

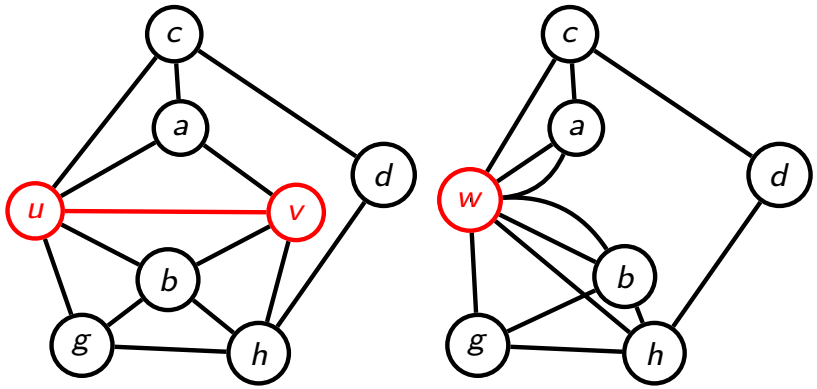
Monte-Carlo algorithm for the Min-Cut problem

D. Karger, 1993.

The algorithm contracts edges of $G = (V, E)$, this operation will produce a graph with multiple edges but without self-loops:

Contract $(e = (u, v))$

- replace u and v by a super-node w,
- preserve edges, update endpoints of u and v to w,
- eliminate self-loops but keep parallel edges.



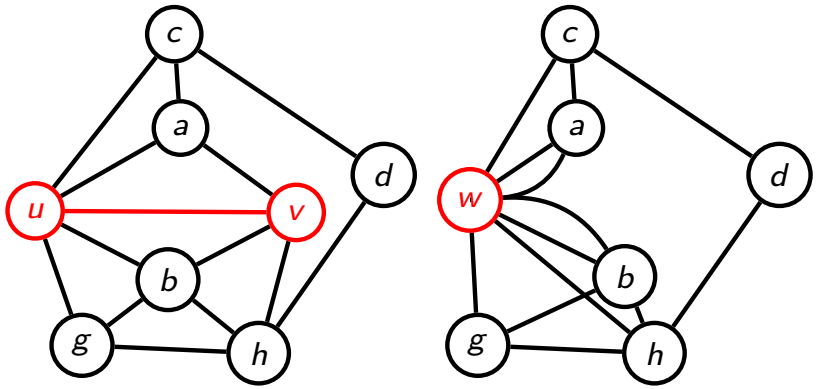
Monte-Carlo algorithm for the Min-Cut problem

D. Karger, 1993.

The algorithm contracts edges of $G = (V, E)$, this operation will produce a graph with multiple edges but without self-loops:

Contract $(e = (u, v))$

- replace u and v by a super-node w,
- preserve edges, update endpoints of u and v to w,
- eliminate self-loops but keep parallel edges.



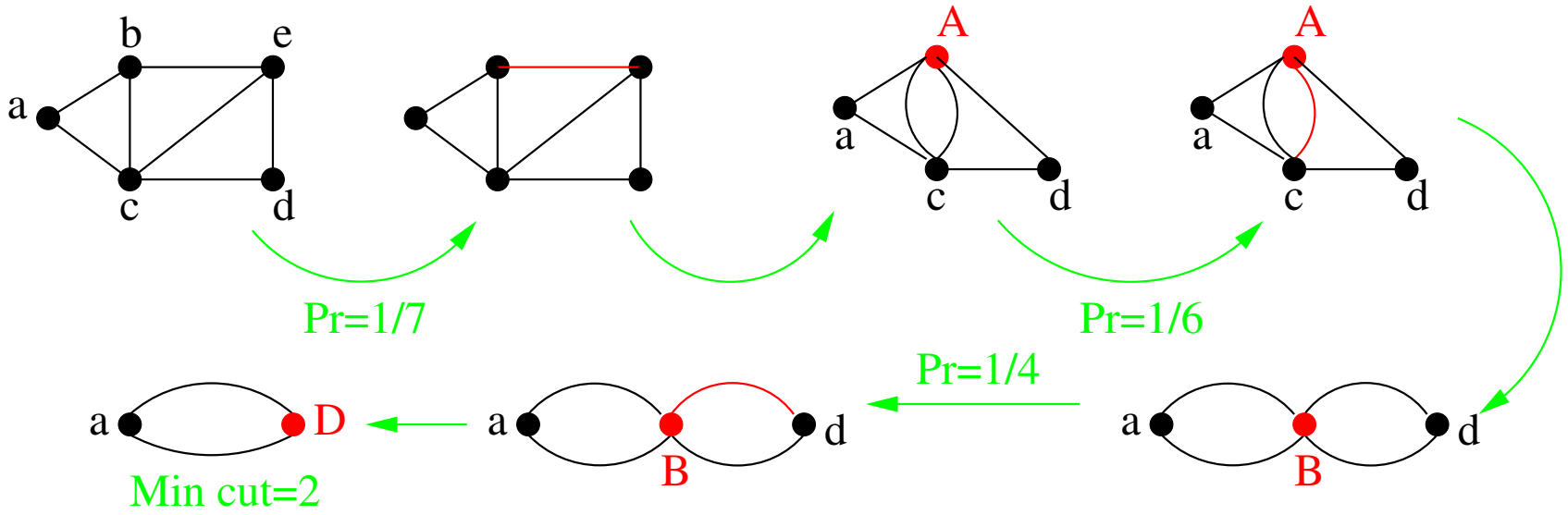
Given G, which DS would you use to implement **Contract**(e)?

Karger's algorithm

Karger $(G = (V, E))$
while $|V| > 2$ **do**
  Chose u.a.r. $e_i = (u, v) \in E$
  $G =$**Contract**(e_i)
end while
return the edges between the 2 remaining vertices

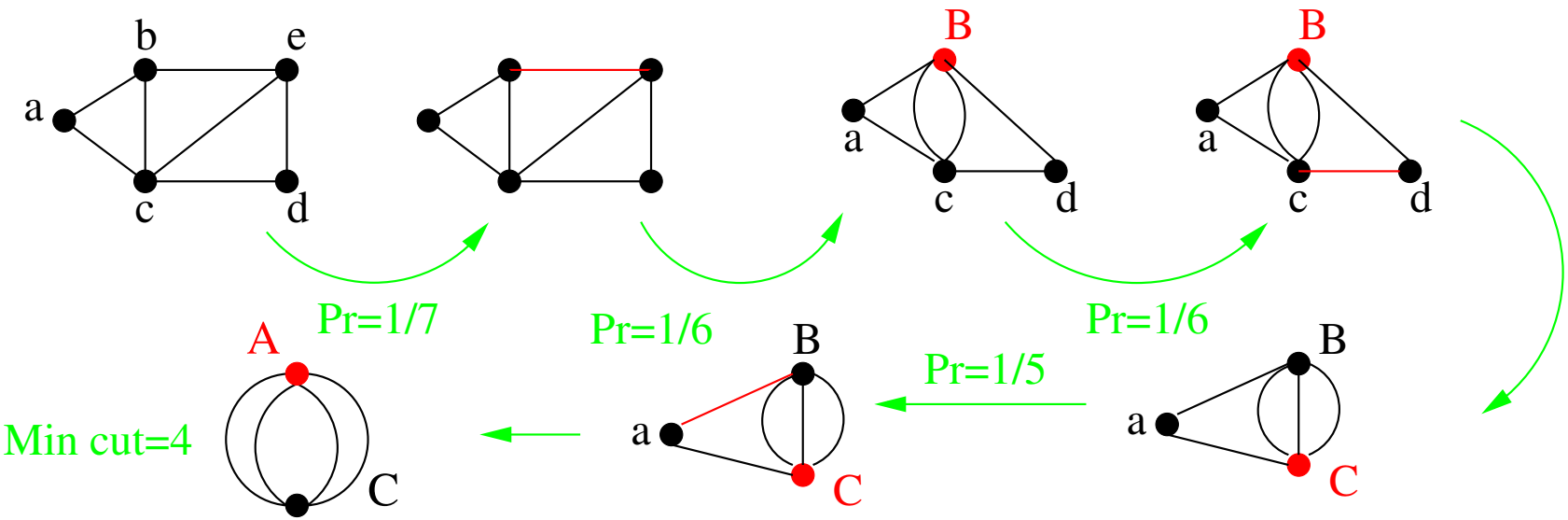
Karger's algorithm

Karger $(G = (V, E))$
while $|V| > 2$ **do**
  Chose u.a.r. $e_i = (u, v) \in E$
  $G =$**Contract**(e_i)
end while
return the edges between the 2 remaining vertices



Karger's algorithm

Karger $(G = (V, E))$
while $|V| > 2$ **do**
  Chose u.a.r. $e_i = (u, v) \in E$
  $G =$**Contract**(e_i)
end while
return the edges between the 2 remaining vertices



Analysis of the algorithm

The running time of the algorithm is $\Theta(n^2)$.

Assume G, with $|V| = n$ has a min-cut set $C \subseteq E$ of size k.

Notice:

- Any cut in a contracted graph is a cut in the initial graph,
- Karger's returns a cut,
- A contraction eliminates all the set of edges among the identified vertices.
- Karger's might provide a cut that is not of minimum size.

Theorem

Karger's algorithm returns a min-cut with probability $\geq 2/n^2$.

Proof of the Theorem

- Let C be a min-cut of G, assume that $|C| = k$
- Let G_i be the graph after i contractions, G_i has $n - i$ nodes.
- If no $e \in C$ has been contracted then C is still a min-cut of G_i.
- Let $\mathcal{E}_i$ be the event none of the edge(s) contracted at the i-iteration is in C and let $\mathcal{O}_i = \cap_{j=1}^{i} \mathcal{E}_i$ i.e. no edge in C is contracted in the first i iterations.

Proof of the Theorem

- We want to compute $\mathbf{Pr}[\mathcal{O}_{n-2}]$ probability of success.
- Notice $\mathbf{Pr}[\mathcal{E}_1] = \mathbf{Pr}[\mathcal{O}_1] \geq 1 - \frac{2k}{nk}$
  As $|C| = k$ all vertices in G must have degree $\geq k$, $|E(G)| \geq nk/2$.
  So 1st contracted edge chosen u.a.r. among the $\geq nk/2$ edges and $|C| = k$)
- The $\mathbf{Pr}[\mathcal{E}_2|\mathcal{O}_1] \geq 1 - \frac{k}{k(n-1)/2} \geq 1 - 2/(n-1)$
  If 1st. contraction did not eliminate an edge in C (i.e conditioning on $\mathcal{O}_1$), we are left with $|V(G_1)| = n - 1$ and $|E(G_1)| \geq k(n-1)/2$, again $\deg(v) \geq k$
- Working iteratively, $\mathbf{Pr}[\mathcal{E}_i|\mathcal{O}_{i-1}] \geq 1 - \frac{2}{(n-i+1)}$.

Proof of the Theorem

From $\mathbf{Pr}[A \cap B] = \mathbf{Pr}[A|B]\,\mathbf{Pr}[B]$:

$$
\begin{aligned}
\mathbf{Pr}[\mathcal{O}_{n-2}] &= \mathbf{Pr}[\mathcal{E}_{n-2} \cap \mathcal{O}_{n-3}]) \\
&= \mathbf{Pr}[\mathcal{E}_{n-2} \mid \mathcal{O}_{n-3}]\,\mathbf{Pr}[\mathcal{O}_{n-3}] \\
&= \mathbf{Pr}[\mathcal{E}_{n-2}|\mathcal{O}_{n-3}]\,\mathbf{Pr}[\mathcal{E}_{n-3}|\mathcal{O}_{n-4}] \ldots \mathbf{Pr}[\mathcal{E}_2|\mathcal{O}_1]\,\mathbf{Pr}[\mathcal{O}_1] \\
&\geq \prod_{i=1}^{n-2}\left(1 - \frac{2}{n-i+1}\right) = \prod_{i=1}^{n-2}\left(\frac{n-i-1}{n-i+1}\right) \\
&= (\frac{n-2}{n})(\frac{n-3}{n-1})(\frac{n-4}{n-2})\cdots(\frac{3}{5})(\frac{2}{4})(\frac{1}{3}) \\
&= \frac{2}{n(n-1)}
\end{aligned}
$$

□

Amplification

To increase the probability of success, run Karger's algorithm several times.

Theorem

If we run Karger's min-cut algorithm $n(n-1)\lg n$ times and output the smallest cut found in all the runs the probability of failure (it is not the global min-cut) is $\leq$

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

The proof is straightforward using the definition of e^{-1}

A randomized algorithm for MAX 3-SAT

A 3-SAT formula is a Boolean formula in CNF such that each clause has exactly 3 literals and each literal corresponds to a different variable.

$$(x_2 \vee \overline{x}_3 \vee \overline{x}_4) \wedge (x_2 \vee x_3 \vee \overline{x}_4) \wedge (\overline{x}_1 \vee x_2 \vee x_4) \wedge (\overline{x}_1 \vee \overline{x}_2 \vee x_3) \wedge (x_1 \vee \overline{x}_2 \vee \overline{x}_4)$$

MAXIMUM 3-SAT. Given a 3-SAT formula, find a truth assignment that satisfies as many clauses as possible.

The problem is NP-hard. We can try to design a randomized algorithm that produces a good assignment, even if it is not optimal.

A randomized algorithm for MAX 3-SAT

Algorithm

For each variable, flip a fair coin, and assign to the variable True (1) if it is heads, False (0), otherwise.

Note that a variable gets 1 with probability $\frac{1}{2}$, and this assignment is made independently of the other variables.

What is the expected number of satisfied clauses?

Assume that the 3-SAT formula has n variables and m clauses.

- Let $Z =$ number of clauses satisfied by the random assignment
- For $1 \leq j \leq m$, define the random variables
  $Z_j = 1$ if clause j is satisfied, 0 otherwise.
- By definition, $Z = \sum_{j=1}^{m} Z_j$.
- $Pr[Z_j = 1] = 1 - (1/2)^3 = 7/8$, so $E[Z_j] = 7/8$. Therefore ,

$$E[Z] = \sum_{j=1}^{m} E[Z_j] = \frac{7}{8}m$$

A randomized algorithm for MAX 3-SAT

How good is the solution computed by the random algorithm?

- For a 3-CNF formula let $opt(F)$ be the maximum number of clauses than can be satisfied by an assignment.
- As for any assignment x the number of satisfied clauses is always $\leq opt(F)$, we have that $E[Z] \leq opt(F)$.
- Of course $opt(F) \leq m$, that is $\frac{7}{8} opt(F) \leq \frac{7}{8} m = E[Z]$, then

$$\frac{opt(F)}{E[Z]} \leq \frac{8}{7}$$

- We get $\frac{7}{8} opt(F) \leq E[Z] \leq opt$, a randomized approximation algorithm.

The probabilistic method

Claim

For any instance of 3-SAT, there exists a truth assignment that satisfies at least a 7/8 fraction of all clauses.

Proof. Random variable must have one event on which the measured value is at least its expectation.

Probabilistic method [Paul Erdös].
Prove the existence of a non-obvious property by showing that a random construction produces it with positive probability