Karger's Contraction Algorithm

Introduction to Big Data Algorithms

Sebastian Forster

University of Salzburg

Global Minimum Cut

Definition

A cut of a graph $G = (V, E)$ with edge capacities is a non-trivial partition (S, T) of the nodes (i.e., $S \cup T = V$, $S \cap T = \emptyset$, and $\emptyset \subset S \subset V$). The capacity of a cut (S, T) is $c(S, T) := \sum_{(u,v) \in E : u \in S, v \in T} c(u, v)$.

Global Minimum Cut

Definition

A cut of a graph $G = (V, E)$ with edge capacities is a non-trivial partition (S, T) of the nodes (i.e., $S \cup T = V$, $S \cap T = \emptyset$, and $\emptyset \subset S \subset V$). The capacity of a cut (S, T) is $c(S, T) := \sum_{(u,v) \in E : u \in S, v \in T} c(u, v)$.

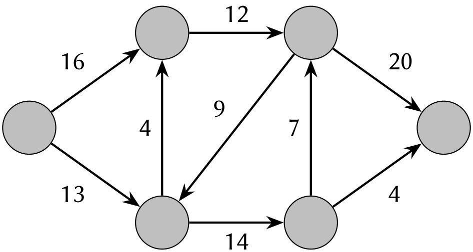
Remark: In the literature, a cut is often defined as a subset of nodes (and $T := V \setminus S$ is implied)

Global Minimum Cut

Definition

A cut of a graph $G = (V, E)$ with edge capacities is a non-trivial partition (S, T) of the nodes (i.e., $S \cup T = V$, $S \cap T = \emptyset$, and $\emptyset \subset S \subset V$). The capacity of a cut (S, T) is $c(S, T) := \sum_{(u,v) \in E : u \in S, v \in T} c(u, v)$.

Remark: In the literature, a cut is often defined as a subset of nodes (and $T := V \setminus S$ is implied)

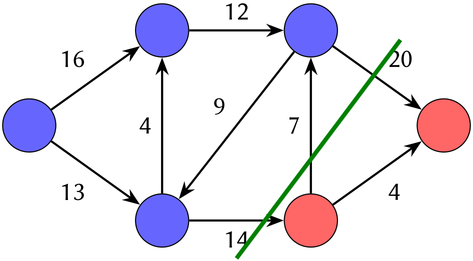


Global Minimum Cut

Definition

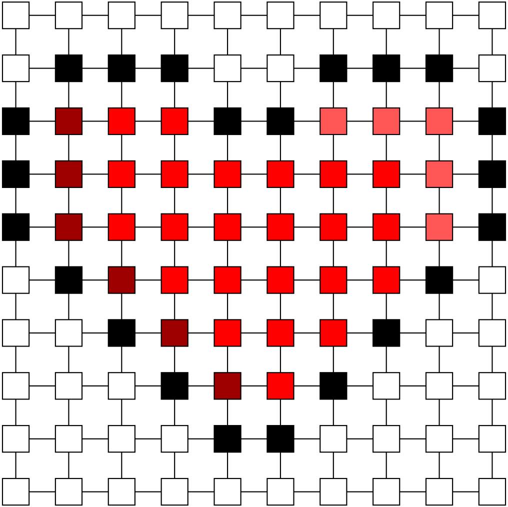
A cut of a graph $G = (V, E)$ with edge capacities is a non-trivial partition (S, T) of the nodes (i.e., $S \cup T = V$, $S \cap T = \emptyset$, and $\emptyset \subset S \subset V$). The capacity of a cut (S, T) is $c(S, T) := \sum_{(u,v) \in E : u \in S, v \in T} c(u, v)$.

Remark: In the literature, a cut is often defined as a subset of nodes (and $T := V \setminus S$ is implied)



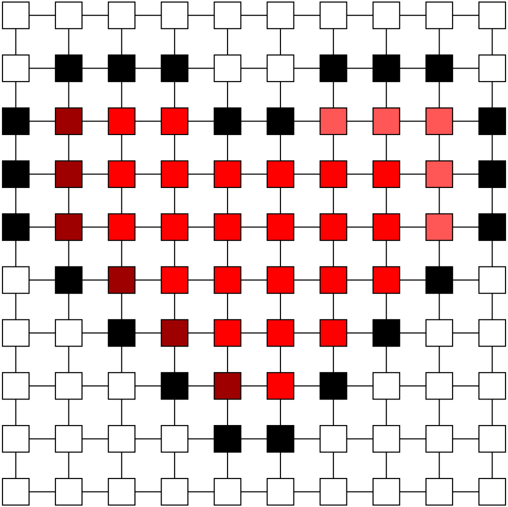
Example: Image Segmentation

Image as **grid graph**



Example: Image Segmentation

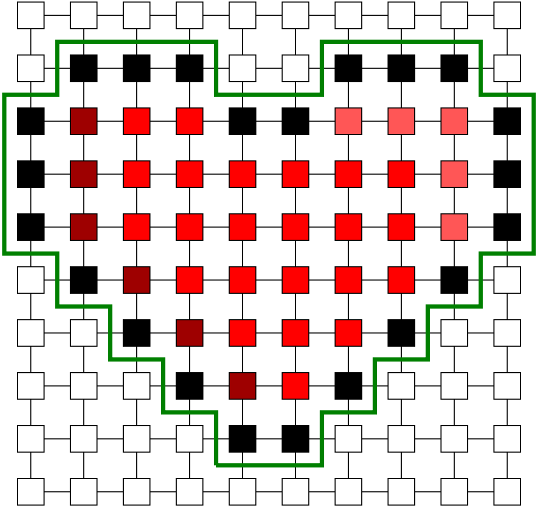
Image as **grid graph**



Edge weights: measure similarity of neighboring pixels (the more similarity, the higher the costs of separating the neighbors)

Example: Image Segmentation

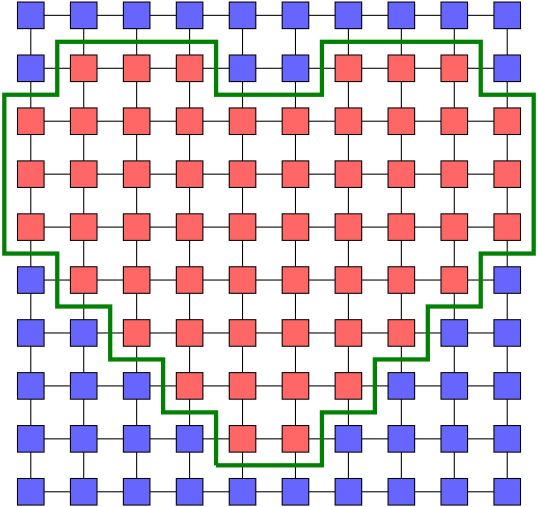
Image as **grid graph**



Edge weights: measure similarity of neighboring pixels (the more similarity, the higher the costs of separating the neighbors)

Example: Image Segmentation

Image as **grid graph**



Edge weights: measure similarity of neighboring pixels (the more similarity, the higher the costs of separating the neighbors)

Edge Connectivity

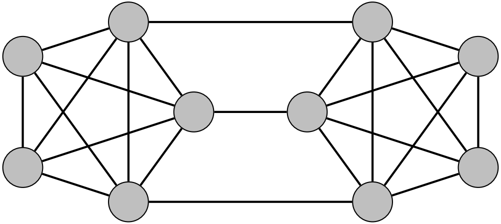
Definition

An (unweighted) graph G is k-**edge-connected**, if G is connected after removing less than k edges. The **edge connectivity** λ of G is the maximum k for which G is k-edge-connected.

Edge Connectivity

Definition

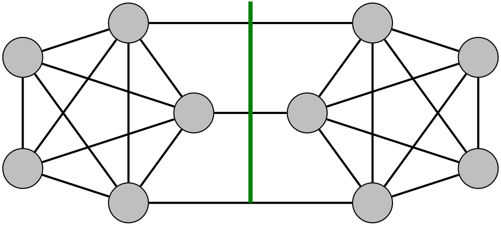
An (unweighted) graph G is k**-edge-connected**, if G is connected after removing less than k edges. The **edge connectivity** λ of G is the maximum k for which G is k-edge-connected.



Edge Connectivity

Definition

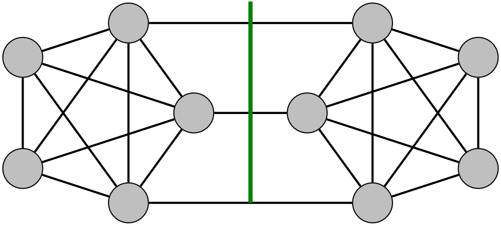
An (unweighted) graph G is k**-edge-connected**, if G is connected after removing less than k edges. The **edge connectivity** λ of G is the maximum k for which G is k-edge-connected.



Edge Connectivity

Definition

An (unweighted) graph G is k-**edge-connected**, if G is connected after removing less than k edges. The **edge connectivity** λ of G is the maximum k for which G is k-edge-connected.



Observation

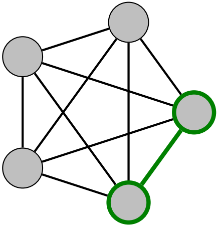
In an unweighted graph, the edge connectivity equals the capacity of the minimum cut ($\lambda = \Gamma^*$).

Edge Contraction

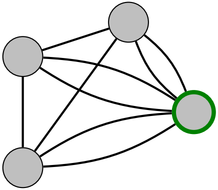
Basic operation: contracting an edge $e = (u, v)$ in $G = (V, E)$

Edge Contraction

Basic operation: contracting an edge $e = (u, v)$ in $G = (V, E)$



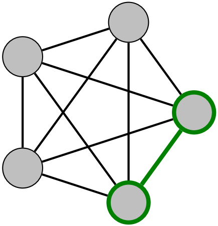
$G = (V, E)$



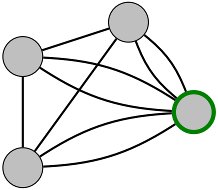
$G' = (V', E')$

Edge Contraction

Basic operation: contracting an edge $e = (u, v)$ in $G = (V, E)$



$G = (V, E)$



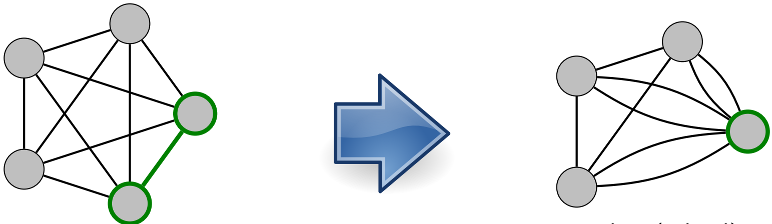
$G' = (V', E')$

Formal definition:

- $V' = V \setminus \{u, v\} \cup \{w\}$

Edge Contraction

Basic operation: contracting an edge $e = (u, v)$ in $G = (V, E)$



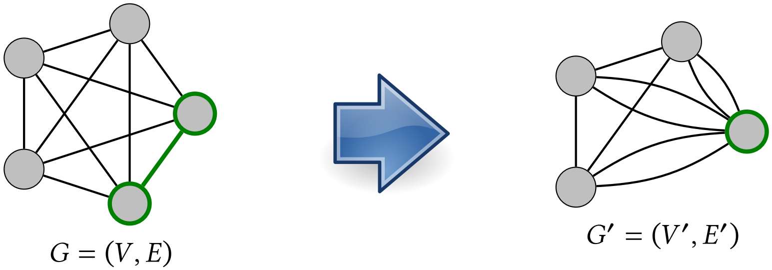
$G = (V, E)$ $\quad$ $G' = (V', E')$

Formal definition:

- $V' = V \setminus \{u, v\} \cup \{w\}$
- $E' = E \setminus \{e : E \mid u \in e \vee v \in e\} \cup \{(w', w) : (u, w') \in E \vee (v, w') \in E\}$

Edge Contraction

Basic operation: contracting an edge $e = (u, v)$ in $G = (V, E)$

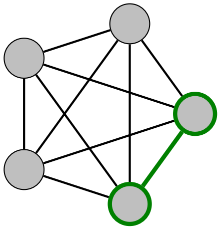


Formal definition:

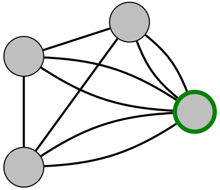
- $V' = V \setminus \{u, v\} \cup \{w\}$
- $E' = E \setminus \{e : E \mid u \in e \vee v \in e\} \cup \{(w', w) : (u, w') \in E \vee (v, w') \in E\}$
- Attention: leads to multigraphs with parallel edges

Edge Contraction

Basic operation: contracting an edge $e = (u, v)$ in $G = (V, E)$



$G = (V, E)$



$G' = (V', E')$

Formal definition:

- $V' = V \setminus \{u, v\} \cup \{w\}$
- $E' = E \setminus \{e : E \mid u \in e \vee v \in e\} \cup \{(w', w) : (u, w') \in E \vee (v, w') \in E\}$
- Attention: leads to multigraphs with parallel edges

Running time: $O(n)$ per contraction

Karger's Contraction Algorithm

Algorithm:

while $|V| > 2$ **do**
    Choose an edge e uniformly at random from E
    $(V, E) \leftarrow contract(V, E, e)$
return $|E|$

Karger's Contraction Algorithm

Algorithm:

while $|V| > 2$ **do**
    Choose an edge e uniformly at random from E
    $(V, E) \leftarrow contract(V, E, e)$
return $|E|$

Running time:

- In every iteration one node is eliminated
- Thus: $n - 2$ iterations

Karger's Contraction Algorithm

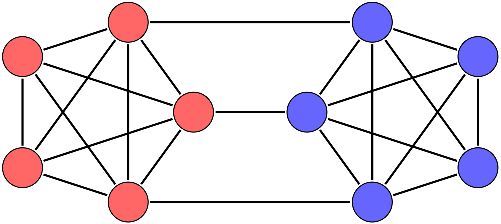
Algorithm:

```
while |V| > 2 do
    Choose an edge e uniformly at random from E
    (V, E) ← contract(V, E, e)
return |E|
```

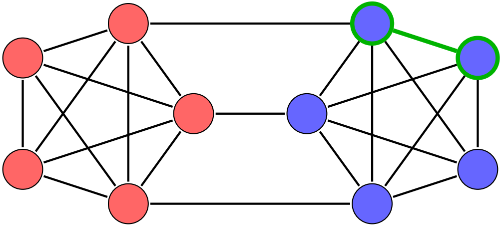
Running time:

- In every iteration one node is eliminated
- Thus: $n - 2$ iterations
- Running time $O(n)$ per iteration
- Thus: total running time $O(n^2)$

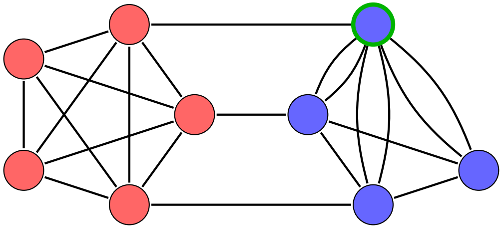
Example



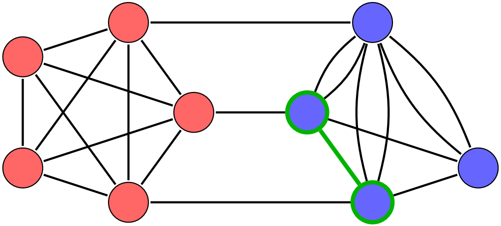
Example



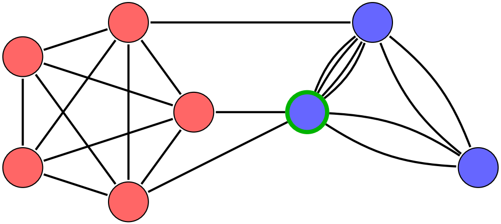
Example



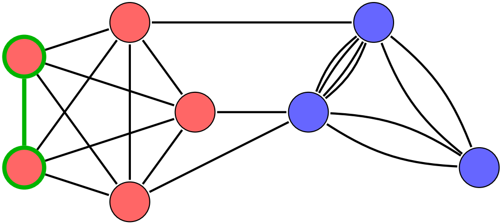
Example



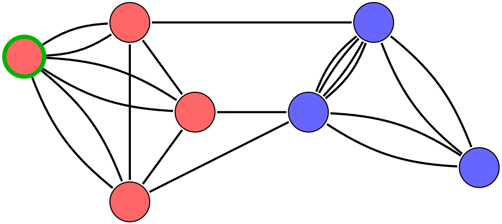
Example



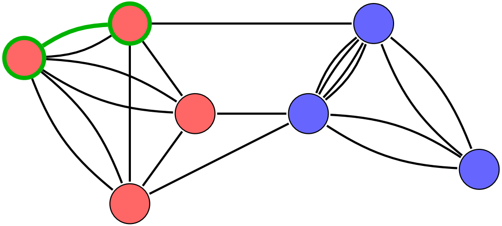
Example



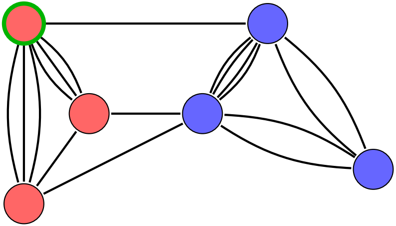
Example



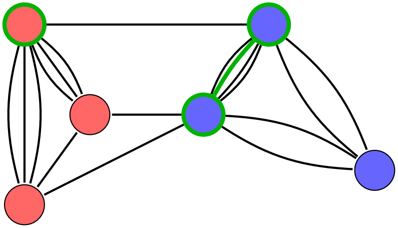
Example



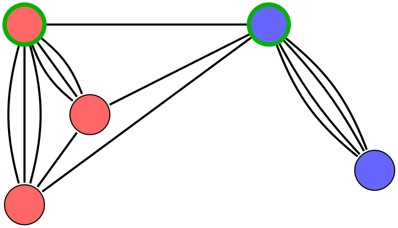
Example



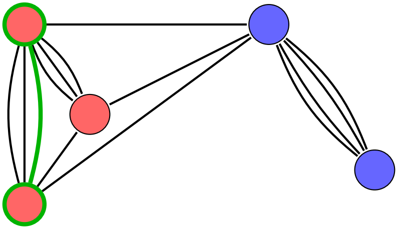
Example



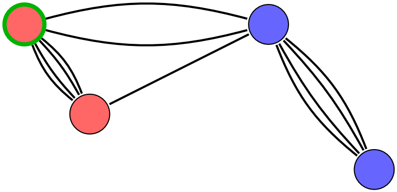
Example



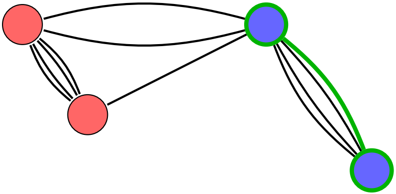
Example



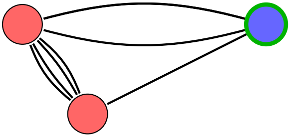
Example



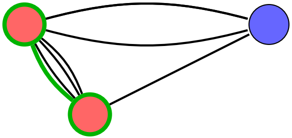
Example



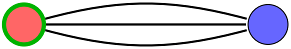
Example



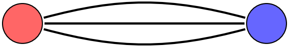
Example



Example



Example



Analysis of Success Probability

Intuition:

- Algorithm only errs if edge crossing the minimum cut is chosen

Analysis of Success Probability

Intuition:

- Algorithm only errs if edge crossing the minimum cut is chosen
- By minimality: low probability of choosing such an edge

Analysis of Success Probability

Intuition:

- Algorithm only errs if edge crossing the minimum cut is chosen
- By minimality: low probability of choosing such an edge

Analysis of first contraction:

Analysis of Success Probability

Intuition:

- Algorithm only errs if edge crossing the minimum cut is chosen
- By minimality: low probability of choosing such an edge

Analysis of first contraction:

- Let λ be the edge connectivity

Analysis of Success Probability

Intuition:

- Algorithm only errs if edge crossing the minimum cut is chosen
- By minimality: low probability of choosing such an edge

Analysis of first contraction:

- Let λ be the edge connectivity
- Every node has degree $\geq \lambda$

Analysis of Success Probability

Intuition:

- Algorithm only errs if edge crossing the minimum cut is chosen
- By minimality: low probability of choosing such an edge

Analysis of first contraction:

- Let λ be the edge connectivity
- Every node has degree $\geq \lambda$
- $m = \frac{1}{2} \sum_{v \in V} \deg(v) \geq \frac{n}{2} \cdot \lambda$

Analysis of Success Probability

Intuition:

- Algorithm only errs if edge crossing the minimum cut is chosen
- By minimality: low probability of choosing such an edge

Analysis of first contraction:

- Let λ be the edge connectivity
- Every node has degree $\geq \lambda$
- $m = \frac{1}{2} \sum_{v \in V} \deg(v) \geq \frac{n}{2} \cdot \lambda$
- Every edge is chosen with probability $\frac{1}{m}$

Analysis of Success Probability

Intuition:

- Algorithm only errs if edge crossing the minimum cut is chosen
- By minimality: low probability of choosing such an edge

Analysis of first contraction:

- Let λ be the edge connectivity
- Every node has degree $\geq \lambda$
- $m = \frac{1}{2} \sum_{v \in V} \deg(v) \geq \frac{n}{2} \cdot \lambda$
- Every edge is chosen with probability $\frac{1}{m}$
- Probability of choosing one of the λ edges of the minimum cut (S^*, T^*):

Analysis of Success Probability

Intuition:

- Algorithm only errs if edge crossing the minimum cut is chosen
- By minimality: low probability of choosing such an edge

Analysis of first contraction:

- Let λ be the edge connectivity
- Every node has degree $\geq \lambda$
- $m = \frac{1}{2} \sum_{v \in V} \deg(v) \geq \frac{n}{2} \cdot \lambda$
- Every edge is chosen with probability $\frac{1}{m}$
- Probability of choosing one of the λ edges of the minimum cut (S^*, T^*):

$$\frac{\lambda}{m} \leq \frac{2}{n}$$

Analysis of Success Probability

Intuition:

- Algorithm only errs if edge crossing the minimum cut is chosen
- By minimality: low probability of choosing such an edge

Analysis of first contraction:

- Let λ be the edge connectivity
- Every node has degree $\geq \lambda$
- $m = \frac{1}{2} \sum_{v \in V} \deg(v) \geq \frac{n}{2} \cdot \lambda$
- Every edge is chosen with probability $\frac{1}{m}$
- Probability of choosing one of the λ edges of the minimum cut (S^*, T^*):

$$\frac{\lambda}{m} \leq \frac{2}{n}$$

Arbitrary contraction: If minimum cut still has λ edges, the same argument works!

Analysis of Success Probability

Intuition:

- Algorithm only errs if edge crossing the minimum cut is chosen
- By minimality: low probability of choosing such an edge

Analysis of first contraction:

- Let λ be the edge connectivity
- Every node has degree $\geq \lambda$
- $m = \frac{1}{2} \sum_{v \in V} \deg(v) \geq \frac{n}{2} \cdot \lambda$
- Every edge is chosen with probability $\frac{1}{m}$
- Probability of choosing one of the λ edges of the minimum cut (S^*, T^*):

$$\frac{\lambda}{m} \leq \frac{2}{n}$$

Arbitrary contraction: If minimum cut still has λ edges, the same argument works!
(Probability $\leq \frac{2}{n'}$ for graph with n' nodes)

Analysis of Success Probability (cont'd)

Event E_i: in iteration i none of the λ edges of minimum cut (S^*, T^*) is chosen

Analysis of Success Probability (cont'd)

Event E_i: in iteration i none of the λ edges of minimum cut (S^*, T^*) is chosen

Probability of never choosing an edge from (S^*, T^*):

Analysis of Success Probability (cont'd)

Event E_i: in iteration i none of the λ edges of minimum cut (S^*, T^*) is chosen

Probability of never choosing an edge from (S^*, T^*):

$$\Pr[E_1 \wedge E_2 \wedge \dots \wedge E_{n-2}] = \Pr[E_1] \cdot \Pr[E_2 \mid E_1] \cdot \dots \cdot \Pr[E_{n-2} \mid E_1 \wedge \dots \wedge E_{n-3}]$$

Analysis of Success Probability (cont'd)

Event E_i: in iteration i none of the λ edges of minimum cut (S^*, T^*) is chosen

Probability of never choosing an edge from (S^*, T^*):

$$\begin{aligned}\Pr[E_1 \wedge E_2 \wedge \dots \wedge E_{n-2}] &= \Pr[E_1] \cdot \Pr[E_2 \mid E_1] \cdot \dots \cdot \Pr[E_{n-2} \mid E_1 \wedge \dots \wedge E_{n-3}] \\ &\geq \left(1 - \frac{2}{n}\right) \cdot \left(1 - \frac{2}{n-1}\right) \cdot \dots \cdot \left(1 - \frac{2}{3}\right)\end{aligned}$$

Analysis of Success Probability (cont'd)

Event E_i: in iteration i none of the λ edges of minimum cut (S^*, T^*) is chosen

Probability of never choosing an edge from (S^*, T^*):

$$\begin{aligned}
\Pr[E_1 \wedge E_2 \wedge \dots \wedge E_{n-2}] &= \Pr[E_1] \cdot \Pr[E_2 \mid E_1] \cdot \dots \cdot \Pr[E_{n-2} \mid E_1 \wedge \dots \wedge E_{n-3}] \\
&\geq \left(1 - \frac{2}{n}\right) \cdot \left(1 - \frac{2}{n-1}\right) \cdot \dots \cdot \left(1 - \frac{2}{3}\right) \\
&= \frac{n-2}{n} \cdot \frac{n-3}{n-1} \cdot \frac{n-4}{n-2} \cdot \frac{n-5}{n-3} \cdot \dots \cdot \frac{4}{6} \cdot \frac{3}{5} \cdot \frac{2}{4} \cdot \frac{1}{3}
\end{aligned}$$

Analysis of Success Probability (cont'd)

Event E_i: in iteration i none of the λ edges of minimum cut (S^*, T^*) is chosen

Probability of never choosing an edge from (S^*, T^*):

$$\begin{aligned}
\Pr[E_1 \wedge E_2 \wedge \dots \wedge E_{n-2}] &= \Pr[E_1] \cdot \Pr[E_2 \mid E_1] \cdot \dots \cdot \Pr[E_{n-2} \mid E_1 \wedge \dots \wedge E_{n-3}] \\
&\geq \left(1 - \frac{2}{n}\right) \cdot \left(1 - \frac{2}{n-1}\right) \cdot \dots \cdot \left(1 - \frac{2}{3}\right) \\
&= \frac{n-2}{n} \cdot \frac{n-3}{n-1} \cdot \frac{n-4}{n-2} \cdot \frac{n-5}{n-3} \cdot \dots \cdot \frac{4}{6} \cdot \frac{3}{5} \cdot \frac{2}{4} \cdot \frac{1}{3} \\
&= \frac{2}{n \cdot (n-1)}
\end{aligned}$$

Boosting of Success Probability

Repeat algorithm $r = \left\lceil \frac{n(n-1)}{2} \right\rceil$ times

(and return minimum of all return values)

Boosting of Success Probability

Repeat algorithm $r = \left\lceil \frac{n(n-1)}{2} \right\rceil$ times

(and return minimum of all return values)

Error probability with r repetitions:

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

Boosting of Success Probability

Repeat algorithm $r = \left\lceil \frac{n(n-1)}{2} \right\rceil$ times

(and return minimum of all return values)

Error probability with r repetitions:

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

Important Inequality

$(1-x)^{\frac{1}{x}} \leq \frac{1}{e}$ for $x \geq 1$

Boosting of Success Probability

Repeat algorithm $r = \left\lceil \frac{n(n-1)}{2} \right\rceil$ times

(and return minimum of all return values)

Error probability with r repetitions:

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

Important Inequality

$(1 - x)^{\frac{1}{x}} \leq \frac{1}{e}$ for $x \geq 1$

In general: Error probability with $r = \left\lceil \frac{n(n-1)}{2} \ln \frac{1}{p} \right\rceil$ repetitions:

Boosting of Success Probability

Repeat algorithm $r = \left\lceil \frac{n(n-1)}{2} \right\rceil$ times

(and return minimum of all return values)

Error probability with r repetitions:

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

Important Inequality

$(1-x)^{\frac{1}{x}} \leq \frac{1}{e}$ for $x \geq 1$

In general: Error probability with $r = \left\lceil \frac{n(n-1)}{2} \ln \frac{1}{p} \right\rceil$ repetitions:

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

Boosting of Success Probability

Repeat algorithm $r = \left\lceil \frac{n(n-1)}{2} \right\rceil$ times

(and return minimum of all return values)

Error probability with r repetitions:

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

Important Inequality

$(1-x)^{\frac{1}{x}} \leq \frac{1}{e}$ for $x \geq 1$

In general: Error probability with $r = \left\lceil \frac{n(n-1)}{2} \ln \frac{1}{p} \right\rceil$ repetitions:

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

Typical: Error probability of $\frac{1}{n^2}$ requires $O(n^2 \log n)$ repetitions

Summary

Theorem

The contraction algorithm computes the edge connectivity with error probability p in time $O(n^4 \log \frac{1}{p})$.

Summary

Theorem

The contraction algorithm computes the edge connectivity with error probability p in time $O(n^4 \log \frac{1}{p})$.

$O(n^2 \log \frac{1}{p})$ repetitions of contraction algorithm

Summary

Theorem

The contraction algorithm computes the edge connectivity with error probability p in time $O(n^4 \log \frac{1}{p})$.

$O(n^2 \log \frac{1}{p})$ repetitions of contraction algorithm

Remarks:

- More efficient approach using this idea: running time $O(n^2 \log^3 n)$ [Karger/Stein '96]

Summary

Theorem

The contraction algorithm computes the edge connectivity with error probability p in time $O(n^4 \log \frac{1}{p})$.

$O(n^2 \log \frac{1}{p})$ repetitions of contraction algorithm

Remarks:

- More efficient approach using this idea: running time $O(n^2 \log^3 n)$ [Karger/Stein '96]
- Contraction idea does not generalize to directed graphs

Questions