

# Min-Cuts Algorithms

Hu SiXing, Hakki Can Karaimer, Pan An,  
Philipp Keck, Taehoon Kim

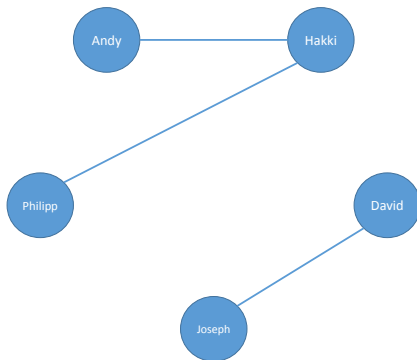
National University of Singapore

March 15, 2016

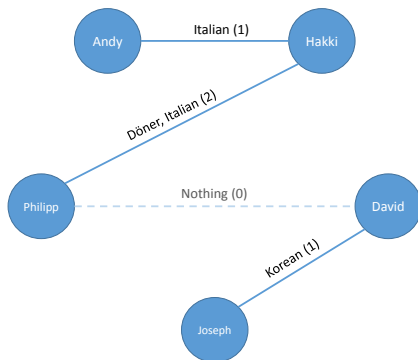
# Motivation

# Motivation

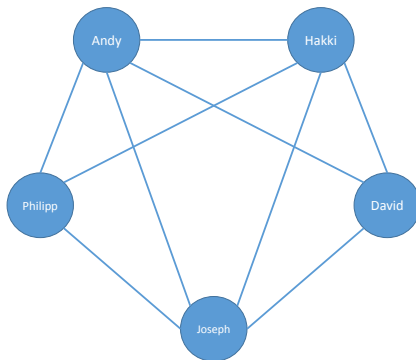
# Food Tour Example



# Food Tour Example



# Food Tour Example



# Definition

## Cut

- Undirected graph  $G = (V, E)$ ,  $n = \|V\|$ ,  $m = \|E\|$
- Cut problems can be described as partitioning  $V$  into  $S$  and  $\bar{S}$ , where  $S \subset V$ :

## Description

- Alternatively, finding a subset of  $E$  that, if removed, splits the graph into two connected components.
- Weight: number of edges between  $S$  and  $\bar{S}$ :

$$w(S, \bar{S}) = \|\{u, v\} \in E \mid u \in S \wedge v \in \bar{S}\|$$

# Definition

## Min-Cut

A cut with minimum weight and  $S, \bar{S} \neq \emptyset$

## Other Cut Problems

- Directed Cut: Given  $s, t \in V$ , ensure  $s \in S \wedge t \in \bar{S}$ ;
- K-Cut: Cuts the graph into  $k$  connected components;
- Sparsest Cut: The sparsest cut problem is to bipartition the vertices so as to minimize the ratio of the number of edges across the cut divided by the number of vertices in the smaller half of the partition.



# Generalization

## Difficulty of Cuts

Minimum cut can be considered as a subset of  $k$  – cut where  $k$  is a fixed number 2.

- $k$  – cut is NP – Complete problem if  $k$  is part of the input.
- Minimum cut is polynomial time calculable.

Add anything here?

# Karger's Algorithm

- Contraction method is used.
- Randomized selection of Edges.
- Running multiple times of the algorithm will provide more accurate result.

# Karger's Algorithm

- Basically one run of Karger's Algo takes  $O(n^2)$  time;
- One time running the algorithm will make errors at the probability of  $O(\frac{1}{n^2})$ ;
- Higher accuracy can be achieved by running the algorithm multiple times;
- It achieves error probability of  $\frac{1}{\text{poly}(n)}$  with  $O(n^4 \log n)$  time.

# Karger's Algorithm

- Improved Karger's algorithm was developed by Karger and Stein;
- Achieving  $O(n^2 \log^3 n)$  running time;
- $O(\frac{1}{n})$  error probability.

Derivation will be given in the later part.

# Karger's Algorithm

## Comparison

- A trivial algorithm checks all  $2^{|V|}$  possible subsets of  $V$  and computes the weight of the resulting cuts, which takes  $2^{|V|}$  time in total.
- Another algorithm (the one that's based on many max-flow-computation) takes  $2^{|V|}$  time.

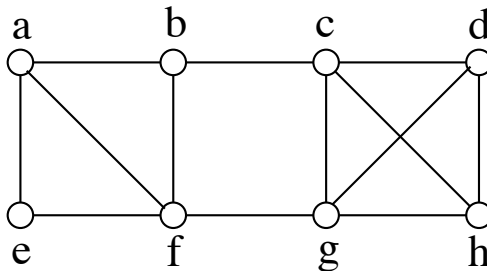
Well that's a lot of TODOS

# Algorithm

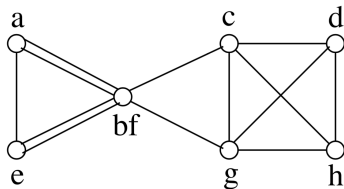
## Karger's Algorithm:

```
repeat
|   choose an edge  $(v, w)$  uniformly at random from  $G$ ;
|   let  $G \leftarrow \frac{G}{(v, w)}$ 
until  $G$  has 2 vertices;
```

# Algorithm

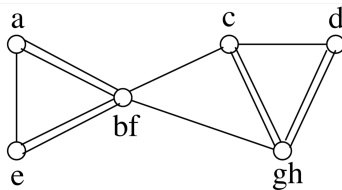


# Algorithm

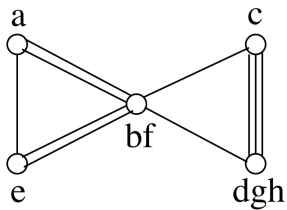




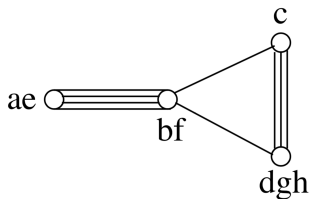
# Algorithm



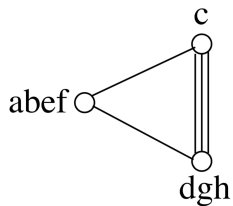
# Algorithm



# Algorithm



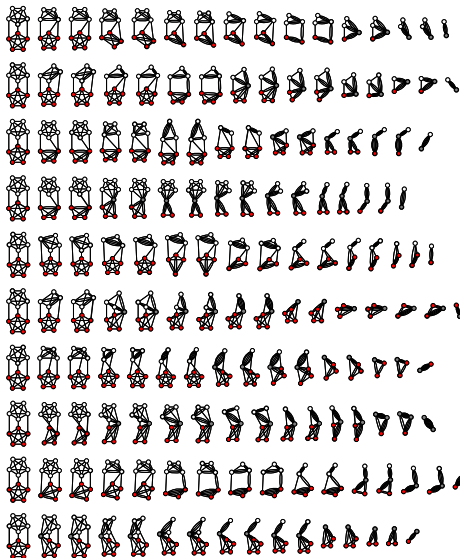
# Algorithm



# Algorithm

abef ○══○ cdgh

# Results



# Fact 1 – Sum of Degrees

$$\sum_{u \in V} \text{degree}(u) = 2|E|$$

Every edge contributes exactly once to the degree of exactly two nodes.

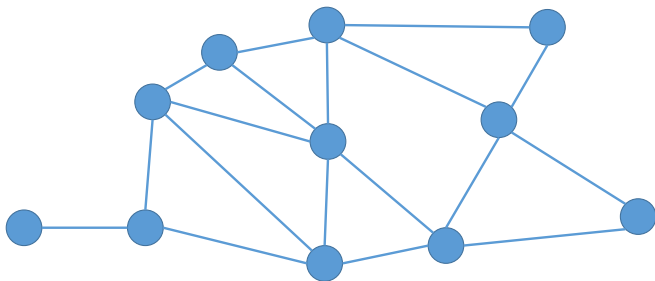
## Fact 2 – Average Degree

$$\begin{aligned} E(\text{degree}(X)) &= \sum_{u \in V} \Pr(X = u) \cdot \text{degree}(u) \\ &= \frac{1}{n} \sum_u \text{degree}(u) = \frac{2|E|}{n} \end{aligned} \tag{1}$$



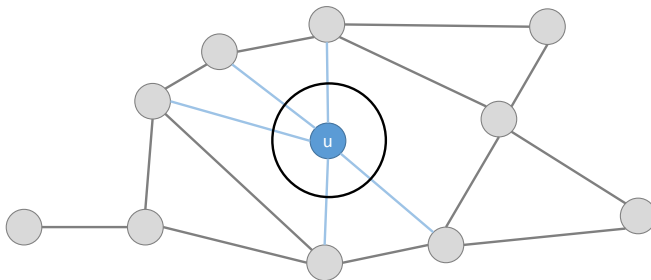
## Fact 3 – Min-cut Size

The size of a min-cut is at most  $\frac{2|E|}{n}$ .



# Fact 3 – Min-cut Size

The size of a min-cut is at most  $\frac{2|E|}{n}$ .



## Fact 3 – Min-cut Size

The size of a min-cut is at most  $\frac{2|E|}{n}$ .

### Proof

- For every node  $u$ , we have a cut of size  $\text{degree}(u)$ .
- Not all nodes can have degree above average, i.e.

$$\exists u \in V: \text{degree}(u) \leq \frac{2|E|}{n}$$

## Fact 4 – Pr[edge across min-cut]

- Fix a certain min-cut in a graph.
- At most  $2|E|/n$  of all edges are part of this min-cut.
- Choose a random edge out of all  $|E|$  edges.
- $\Pr(\text{edge crosses the cut}) = \frac{2|E|/n}{|E|} = \frac{2}{n}$

# Concentrating = Not Cutting

- TODO show the running example and explain how an edge that has been concentrated will never be cut.
- And the edges that remain at the end, because they have never been concentrated.

# Success Probability

- Fix a certain min-cut
- We can never contract an edge from that cut
- $\Pr[\text{first edge is not in min cut}] = 1 - \frac{2}{n}$
- Now  $n - 1$  edges remaining, so  
 $\Pr[\text{second edge is not in min cut}] = (1 - \frac{2}{n-1})$

# Success Probability

## First Cut is not Fixed Cut

$$\begin{aligned}
 & \Pr[\text{first cut is not the fixed cut}] \\
 &= (\Pr[\text{no contracted edge is in minimum cut}]) \\
 &\leq \left( \left(1 - \frac{2}{n}\right) \left(1 - \frac{2}{n-1}\right) \left(1 - \frac{2}{n-2}\right) \dots \left(1 - \frac{2}{4}\right) \left(1 - \frac{2}{3}\right) \right) \\
 &= \left( \frac{n-2}{n} \right)
 \end{aligned}$$

# Success Probability

## Series Converge to e

- $e = \lim(1 + \frac{1}{n})^n$
- $\lim(1 + \frac{1}{n})^n = \lim(1 + \frac{1}{\frac{n}{a}})^{\frac{n}{a} \cdot a}$
- Let  $x = \frac{n}{a}$

$$\lim(1 + \frac{1}{n})^n = \lim(1 + \frac{1}{x})^{x \cdot a} = (\lim(1 + \frac{1}{x})^x)^a = e^x$$



# Boosting Success Probability

Assume that we can get the best result by running the algorithm  $k$  times.

- Probability of at least one success:  $1 - (1 - \frac{1}{\binom{n}{2}})^k$
- To make it an  $\epsilon$ -series, we need  $k$  to contain  $\binom{n}{2}$ .

Here we can use  $\alpha = -1$  and  $k = \binom{n}{2} \cdot c \cdot \ln n$ . Running time is thus promised:

$$O(k) \cdot O(n^2) = O(n^4 \log n)$$

# Boosting Success Probability

$$\begin{aligned}1 - \left(1 - \frac{1}{\binom{n}{2}}\right)^k &= 1 - \left(1 + \frac{1}{\binom{n}{2}}\right)^{-k} \\&= 1 - \left(\left(1 + \frac{1}{\binom{n}{2}}\right)^{\binom{n}{2}}\right)^{-c \cdot \ln n} \\&= 1 - e^{-c \cdot \ln n} = 1 - (e^{\ln n})^{-c} \\&= 1 - \frac{1}{n^c}\end{aligned}$$

# Boosting Success Probability

The error probability is thus:

$$O\left(\frac{1}{n^c}\right)$$

# Extended Karger's Analysis

One run succeeds with  $\Omega(\frac{1}{\log n})$ . If we run the algorithm  $k = \log^2 n$  times:

$$\begin{aligned}\Pr[\text{at least one run succeeds}] &= 1 - \left(1 - \frac{1}{\log n}\right)^{\log^2 n} \\ &= 1 - \left(1 + \frac{1}{-\log n}\right)^{-\log n \cdot -\log n} \\ &= 1 - e^{-\log n} = 1 - \frac{1}{n} \\ &\Rightarrow \Pr[\text{error}] O\left(\frac{1}{n}\right)\end{aligned}$$

# A Parallel Implementation

## Compact Method

- Generating Permutations of Edges
- Binary searching the prefix that connects all nodes to two connected parts
-

# A Parallel Implementation

## Karger's Algorithm:

COMPACT( $G, L, k$ )

Data: A graph  $G$ , list of edges  $L$ , and parameter  $k$

if  $G$  has  $k$  vertices or  $L = \emptyset$  then

  | return  $G$

else

  | Let  $L_1$  and  $L_2$  be the first and second half of  $L$

  | if  $H$  has fewer connected components in graph  $H = (V, L_1)$  then

    | return COMPACT( $G, L, k$ )

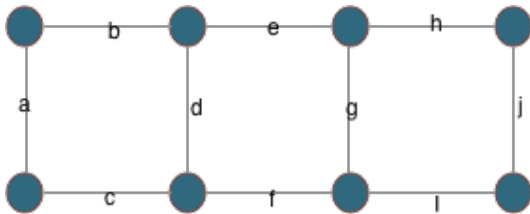
  | else

    | return COMPACT( $G \setminus L_1, L_2 \setminus L_1, k$ ).

  | end

end

# A Parallel Implementation

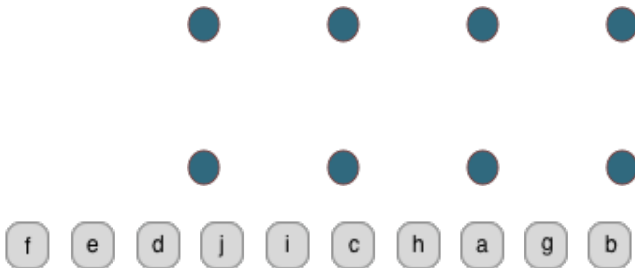


# A Parallel Implementation

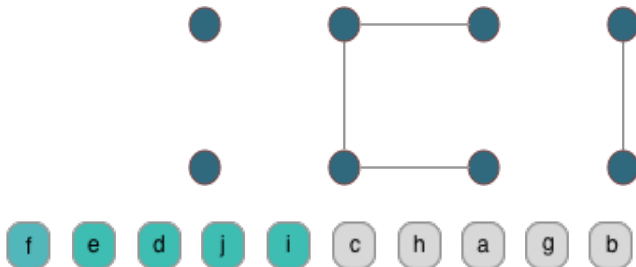




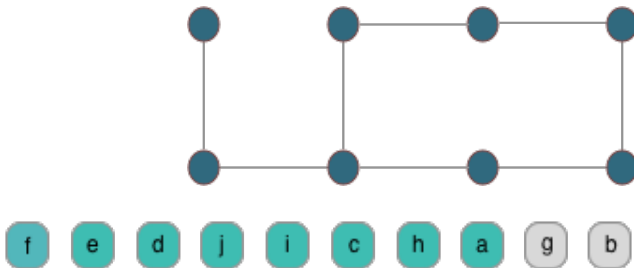
# A Parallel Implementation



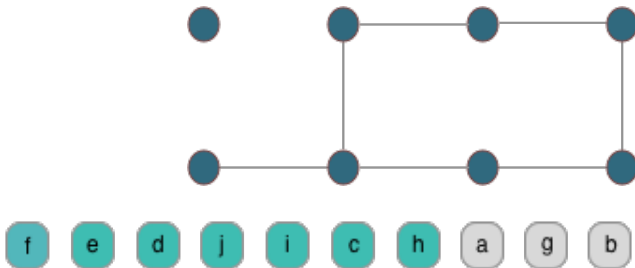
# A Parallel Implementation



# A Parallel Implementation



# A Parallel Implementation



# References