

# CSE 400: Fundamentals of Probability in Computing

## Lecture 10: Randomized Min-Cut Algorithm

### Group-6 Scribe Refactoring

February 5, 2026

## 1 Introduction

This lecture studies the **Minimum Cut (Min-Cut)** problem in graphs, including deterministic and randomized approaches.

Applications include:

- Network design and optimization
- Communication network reliability
- VLSI circuit partitioning

## 2 Min-Cut Problem

### 2.1 Definition of a Cut

Given a graph  $G = (V, E)$ :

A **cut-set** is a set of edges whose removal disconnects the graph into two or more components.

### 2.2 Minimum Cut

The **minimum cut** of  $G$  is a cut-set of minimum cardinality.

Formally:

$$\text{Min-Cut}(G) = \min_{S \subset V} |E(S, V \setminus S)|$$

where  $E(S, V \setminus S)$  is the set of edges crossing the partition.

## 3 Edge Contraction

The core operation in many min-cut algorithms is **edge contraction**.

When contracting edge  $(u, v)$ :

- Merge vertices  $u$  and  $v$
- Remove self-loops

- Retain parallel edges

After repeated contractions, the graph reduces in size.

## 4 Max-Flow Min-Cut Theorem

**Theorem:** In a flow network, the maximum flow from source  $S$  to sink  $T$  is equal to the capacity of the minimum cut.

### 4.1 Definitions

- Capacity of cut: Sum of capacities of edges from  $X$  to  $Y$
- Minimum cut capacity: Smallest such capacity
- Maximum flow: Largest feasible flow from  $S$  to  $T$

## 5 Deterministic Min-Cut Algorithm

### 5.1 Stoer-Wagner Algorithm

Let  $s, t$  be vertices in  $G$ .

A minimum cut of  $G$  is the smaller of:

- Minimum  $s-t$  cut of  $G$
- Minimum cut of  $G/\{s, t\}$

### 5.2 Pseudocode

#### Algorithm 1: MinimumCutPhase( $G, a$ )

```

A ← {a}
while A ≠ V do
    add most tightly connected vertex to A
return cut weight

```

#### Algorithm 2: MinimumCut( $G$ )

```

while |V| > 1 do
    choose a ∈ V
    MinimumCutPhase( $G, a$ )
    update minimum cut if needed
    merge last two added vertices
return minimum cut

```

### 5.3 Time Complexity

Stoer-Wagner runs in:

$$O(VE + V^2 \log V)$$

## 6 Randomized Min-Cut Algorithm

### 6.1 Why Randomization?

- Avoid worst-case inputs
- Faster in practice
- Easier parallelization
- Approximation guarantees

## 7 Karger's Randomized Algorithm

### 7.1 Algorithm Idea

1. While  $|V| > 2$ :
  - Choose random edge
  - Contract it
2. Return remaining cut

### 7.2 Time Complexity

$$O(V^2)$$

## 8 Probability of Success

### Theorem:

Karger's algorithm outputs a minimum cut with probability at least

$$\frac{2}{n(n-1)}$$

### 8.1 Proof Sketch

Let the minimum cut size be  $k$ .

At each contraction step:

$$P(\text{avoid min-cut edge}) \geq 1 - \frac{k}{\text{total edges}}$$

After  $n - 2$  contractions:

$$P(\text{success}) \geq \prod_{i=0}^{n-3} \left(1 - \frac{2}{n-i}\right) = \frac{2}{n(n-1)}$$

Thus repeating algorithm multiple times increases success probability.

## 9 Comparison

Property	Deterministic	Randomized
Guarantee	Exact	High Probability
Time Complexity	$O(VE + V^2 \log V)$	$O(V^2)$
Worst Case	Predictable	Probabilistic
Parallelization	Limited	Easy

## 10 Python Simulation

A simulation notebook was provided for experimentation with Karger's algorithm.

Students are encouraged to:

- Run multiple trials
- Observe probability convergence
- Compare deterministic vs randomized performance

## 11 Exam-Oriented Summary

- Understand definition of cut and minimum cut.
- Know edge contraction clearly.
- Be able to state Max-Flow Min-Cut theorem.
- Write Stoer-Wagner pseudocode.
- Write Karger's algorithm.
- Derive probability  $\frac{2}{n(n-1)}$ .
- Compare deterministic vs randomized approaches.
- Know time complexities.

## 12 Key Takeaways

- Min-cut measures graph connectivity.
- Deterministic algorithms guarantee exact solutions.
- Randomized algorithms trade certainty for efficiency.
- Repetition improves randomized success probability.