Min-Cut Algorithm Lecture

L10_S2.pdf
PDF

Create a lecture scribe using only the provided lecture slides and textbook context. The scribe should work as reliable reference material for exam preparation and should accurately reflect what was taught in the lecture so that it can be used for revision.

While writing the scribe, explain definitions, notation, assumptions, theorem statements or results, formulas, algorithms, proofs, derivations and worked examples step by step following the same logical reasoning used in the lecture. Include all important steps, mathematical expressions and conclusions needed to understand and reconstruct the concepts during exam revision.

Use only the provided lecture slides and textbook context. Do not add any new explanations, examples, assumptions, formulas or reasoning that are not present in the given materials.

Write in a formal academic style and organize the scribe clearly so that the flow of concepts, definitions, results, algorithms, proofs and examples is easy to follow during revision.

Lecture Scribe: Randomized Min-Cut Algorithm

Course: CSE400 – Fundamentals of Probability in Computing

Instructor: Dhaval Patel, PhD

Based strictly on Lecture Slides and Textbook Context

---

1. Min-Cut Problem

1.1 Motivation: Why Use Min-Cut?

The minimum cut (min-cut) algorithm is used to solve problems related to:

- Network connectivity
- Network reliability
- Optimization problems in networks

These applications arise because the min-cut identifies the smallest set of edges whose removal disconnects the network.

Applications

(1) Network Design
The min-cut helps improve communication efficiency and optimize network flow. It is used to find the minimum capacity cut, which identifies bottlenecks in the network.

(2) Communication Networks
The min-cut helps understand network vulnerability to failures. It enables the construction of robust and fault-tolerant networks.

(3) VLSI Design
In Very Large Scale Integration (VLSI), the min-cut algorithm helps partition circuits into smaller components. This reduces interconnectivity complexity.

---

2. Definition of Cut and Minimum Cut

2.1 Graph Notation

Let

$$G = (V, E)$$

Where:

- V = set of vertices
- E = set of edges
- $n = |V|$ = number of vertices

2.2 Definition: Cut-Set

A cut-set is defined as:

A set of edges whose removal breaks the graph into two or more connected components.

Thus, removing these edges disconnects the graph.

2.3 Definition: Minimum Cut (Min-Cut)

Given a graph

$$G = (V, E)$$

The minimum cut problem is defined as:

Finding a cut-set with minimum cardinality.

This means finding the smallest number of edges whose removal disconnects the graph.

2.4 Sensitivity of Randomized Min-Cut Algorithms

Min-cut algorithms such as Karger's algorithm are randomized and sensitive to edge selection.

If the algorithm contracts critical edges early, it may produce a different cut.

Thus, the algorithm's outcome depends on random choices.

---

3. Edge Contraction Operation

3.1 Definition: Edge Contraction

Edge contraction is the primary operation used in randomized min-cut algorithms.

Edge contraction removes an edge and merges its two endpoints into a single vertex.

3.2 Formal Description

When contracting an edge (u, v):

Step 1: Merge vertices u and v into a single vertex

Step 2: Remove the edge (u, v)

Step 3: Retain all other edges

Step 4: Remove self-loops

Step 5: Parallel edges may remain

This is illustrated in the diagram on page 5, where two vertices merge and multiple edges remain between merged vertices and other vertices.

---

4. Successful and Unsuccessful Min-Cut Runs

Since randomized algorithms involve randomness, different runs produce different results.

4.1 Successful Min-Cut Run

Definition:

A successful min-cut run is when the algorithm correctly identifies the minimum cut of the graph.

The diagram on page 6 shows the sequence of contractions resulting in the correct minimum cut.

Characteristics:

- Critical edges are not contracted prematurely
- Final result equals actual minimum cut

4.2 Unsuccessful Min-Cut Run

Definition:

An unsuccessful run occurs when the algorithm fails to identify the minimum cut.

The diagram on page 7 illustrates a case where contracting incorrect edges leads to an incorrect cut.

Characteristics:

- Critical edges were contracted early
- Final cut is not minimum

---

5. Max-Flow Min-Cut Theorem

5.1 Theorem Statement

The Max-Flow Min-Cut theorem states:

"In a flow network, the maximum amount of flow passing from the source to the sink is equal to the total weight of the edges in a minimum cut."

5.2 Definitions Related to Theorem

Capacity of a Cut

The capacity of a cut is defined as:

$$\text{Capacity of cut} = \Sigma \text{ capacity of edges from } X \text{ to } Y$$

Where:

- X and Y form the partition of vertices

Minimum Cut

The minimum cut is:

The cut with smallest possible capacity.

Minimum Cut Capacity

Defined as:

Capacity of the minimum cut.

Maximum Flow

Defined as:

Largest possible flow from source S to sink T.

5.3 Result of Theorem

$$\text{Maximum Flow} = \text{Minimum Cut Capacity}$$

This is illustrated in the diagram on page 9, showing equal flow and cut capacity.

---

6. Deterministic Min-Cut Algorithm (Stoer–Wagner Algorithm)

6.1 Theorem Used in Stoer–Wagner Algorithm

Let s and t be vertices of graph G.

Let:

$$G/s,t$$

be graph obtained by merging s and t.

Then:

Minimum cut of G is the smaller of:

- Minimum $s-t$ cut of G, OR
- Minimum cut of $G/s,t$

6.2 Explanation of Theorem

Two cases exist:

Case 1: Minimum cut separates s and t

Then minimum $s-t$ cut is minimum cut.

Case 2: Minimum cut does not separate s and t

Then minimum cut exists in contracted graph $G/s,t$.

6.3 Pseudocode: MinimumCutPhase(G, a)

Algorithm:

Step 1: Initialize

$$A \leftarrow \{a\}$$

Step 2: Repeat until

$$A \supseteq V$$

Step 3: Add most tightly connected vertex to A

Step 4: Return cut weight

6.4 Pseudocode: MinimumCut(G)

Step 1: While

$$|V| \geqslant 1$$

Step 2: Choose vertex a

Step 3: Run MinimumCutPhase(G, a)

Step 4: If cut is smaller than current minimum cut

Store cut

Step 5: Merge last two added vertices

Step 6: Repeat

Step 7: Return minimum cut

---

7. Randomized Min-Cut Algorithm

7.1 Why Randomized Algorithm?

Randomized algorithms provide:

- Probabilistic guarantee of success
- More accurate estimate with fewer iterations
- Efficiency
- Parallelization capability
- Approximation guarantees
- Avoidance of worst-case instances
- Robustness
- Heuristic flexibility

---

8. Karger's Randomized Min-Cut Algorithm

8.1 Main Idea

The algorithm repeatedly performs edge contraction randomly until only two vertices remain.

The edges between these two vertices form the cut.

8.2 Algorithm Process

Step 1: Start with graph G

Step 2: Randomly select an edge

Step 3: Contract the edge

Step 4: Repeat until only two vertices remain

Step 5: Remaining edges form the cut

The diagram on page 13 shows a sequence of random contractions leading to a non-minimum cut, illustrating randomness.

---

9. Recursive Randomized Min-Cut Algorithm

9.1 Algorithm Definition

Input:

Undirected multigraph G with n vertices

Constant $\alpha > 0$

Output:

Cut C

9.2 Algorithm Steps

If:

$$n \leqslant \alpha^3$$

Then:

Compute minimum cut using brute force

Else:

Repeat for:

$$i = 1 \text{ to } \alpha$$

Step 1: Create graph G' by random contraction

Step 2: Compute

$$C' \leftarrow RecursiveRandomizedMinCut(G', \alpha)$$

Step 3: If

$$| C' | < | C |$$

Then update

$$C \leftarrow C'$$

Step 4: Return C

10. Comparison: Deterministic vs Randomized Min-Cut

10.1 Deterministic Min-Cut

Properties:

- Guarantees exact minimum cut
- Higher time complexity

Time complexity:

$$O(V \cdot E + V^2 \log V)$$

Where:

- V = vertices
- E = edges

10.2 Randomized Min-Cut

Properties:

- Provides approximate minimum cut with high probability

Time complexity:

$$O(V^2)$$

Where:

- V = vertices

---

11. Theorem: Probability of Finding Min-Cut

The randomized algorithm outputs a minimum cut with probability at least:

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

Where:

- n = number of vertices

---

12. Summary of Key Concepts

Min-cut problem:

Find smallest set of edges disconnecting graph.

Core operation:

Edge contraction

Deterministic algorithm:

Stoer–Wagner algorithm

Randomized algorithm:

Karger’s algorithm

Max-flow min-cut theorem:

$$\text{Maximum flow} = \text{Minimum cut capacity}$$

Randomized algorithm probability:

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

If you want, I can also convert this into:

- Exam-ready handwritten style notes
- OR condensed revision notes (5-page exam format)