Lecture Scribe Min-Cut Algorithm

L10_S2.pdf
PDF

Prepare a lecture scribe based only on the provided lecture slides and textbook context. The scribe should serve as dependable reference material for exam preparation and must accurately capture the content covered in the lecture so that it can be used effectively for revision.

While creating the scribe present definitions, notation, assumptions, theorem statements or results, formulas, algorithms, proofs, derivations and worked examples in a clear step-by-step manner following the same logical sequence as demonstrated in the lecture. Include all necessary intermediate steps, mathematical expressions and conclusions required to understand and reconstruct the concepts during exam revision.

Ensure that the content strictly follows the provided lecture slides and textbook context. Do not introduce any new explanations, examples, assumptions, formula or reasoning that are not explicitly included in the given materials.

Maintain a formal academic tone and structure the scribe in a well-organized manner so that the progression of concepts, definitions, results, algorithms, proofs and examples is clear and easy to follow for revision purposes.

Lecture Scribe

CSE400 – Fundamentals of Probability in Computing

Lecture 10: Randomized Min-Cut Algorithm

Based strictly on Lecture Slides and Textbook Context

---

1. Min-Cut Problem

1.1 Why Use Min-Cut?

The min-cut algorithm is used in applications related to:

1. Network Connectivity, Reliability, and Optimization

The min-cut algorithm helps analyze and optimize networks by identifying the minimum number of edges whose removal disconnects the graph. This helps understand the structure and robustness of networks.

2. Network Design

Min-cut helps improve communication efficiency and optimize network flow by identifying the minimum capacity cut in a network.

3. Communication Networks

Min-cut helps identify vulnerabilities in networks. It is useful for building robust and fault-tolerant communication systems.

4. VLSI Design

In Very Large Scale Integration (VLSI), min-cut is used to partition circuits into smaller components, reducing interconnectivity complexity.

---

2. Definition of Min-Cut

2.1 Cut-Set

Definition:
A cut-set in a graph is a set of edges whose removal breaks the graph into two or more connected components.

2.2 Min-Cut Problem

Given:
A graph

$$G = (V , E)$$

where

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

Problem Statement:

The minimum cut (min-cut) problem is to find a cut-set with minimum cardinality.

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

2.3 Edge Contraction Operation

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

Definition:
Edge contraction is an operation that removes an edge and merges its two endpoints into one vertex.

Formally, when contracting an edge (u, v):

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

Step 2: Remove all edges connecting u and v

Step 3: Retain all other edges in the graph

Result:

- Parallel edges may be created
- Self-loops are removed

The diagram on page 5 illustrates how two vertices are merged and edges updated accordingly.

---

3. Successful and Unsuccessful Min-Cut Runs

Min-cut algorithms, especially randomized ones, may produce different outcomes in different runs.

3.1 Successful Min-Cut Run

Definition:
A successful min-cut run refers to the outcome of the algorithm correctly identifying the minimum cut of the graph.

The diagram on page 6 illustrates a sequence of contractions that leads to the correct minimum cut.

3.2 Unsuccessful Min-Cut Run

Definition:
An unsuccessful min-cut run refers to an iteration where the algorithm fails to identify the true minimum cut.

This occurs when critical edges belonging to the minimum cut are contracted early.

The diagram on page 7 shows a contraction sequence resulting in an incorrect cut.

---

4. Max-Flow Min-Cut Theorem

4.1 Theorem Statement

Max-Flow Min-Cut Theorem:

In a flow network,

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

This means:

The maximum amount of flow from source to sink is equal to the total weight of edges in the minimum cut.

4.2 Definitions

Capacity of a Cut

Capacity of a cut is defined as:

Capacity = Σ capacity of edges from vertices in set X to set Y

Minimum Cut

Minimum cut is defined as:

The cut with smallest possible capacity.

Minimum Cut Capacity

Minimum cut capacity is:

The capacity of the minimum cut.

Maximum Flow

Maximum flow is:

The largest possible flow from source S to sink T.

The diagram on page 9 illustrates a network with flow and cut separation.

---

5. Deterministic Min-Cut Algorithm

5.1 Stoer-Wagner Min-Cut Algorithm

Let:

- G be a graph
- s, t be two vertices
- $G/\{s, t\}$ be the graph obtained by merging s and t

Theorem

A minimum cut of G is equal to the smaller of:

1. Minimum s-t cut of G

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

Reason

Two possible cases exist:

Case 1:
There exists a minimum cut separating s and t

Then,

Minimum s-t cut is the minimum cut of G

Case 2:
No minimum cut separates s and t

Then,

Minimum cut of $G/\{s, t\}$ is the minimum cut of G

5.2 Pseudocode: MinimumCutPhase

Algorithm: MinimumCutPhase(G, a)

Step 1: Initialize

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

Step 2: While

$$A \supseteq V$$

Add the most tightly connected vertex to set A

Step 3: Return cut weight as cut-of-the-phase

5.3 Pseudocode: MinimumCut(G)

Step 1: While

$$| V | \geqslant 1$$

Step 2: Choose vertex $a \in V$

Step 3: Execute MinimumCutPhase(G, a)

Step 4: If cut-of-phase is smaller than current minimum cut

Update minimum cut

Step 5: Shrink graph by merging last two vertices added

Step 6: Repeat until finished

Step 7: Return minimum cut

5.4 Time Complexity

Stoer-Wagner Algorithm complexity:

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

where

- V = number of vertices
- E = number of edges

---

6. Randomized Min-Cut Algorithm

6.1 Why Randomized Algorithms?

Randomized algorithms provide probabilistic guarantees and may improve efficiency.

Advantages:

- Efficiency
- Parallelization capability
- Approximation guarantees
- Avoidance of worst-case instances
- Heuristic nature
- Robustness

---

7. Karger’s Randomized Min-Cut Algorithm

This algorithm repeatedly performs random edge contractions.

The diagram on page 13 illustrates random edge selection and contraction.

The output cut depends on random choices.

Sometimes correct minimum cut is obtained, sometimes not.

---

8. Recursive Randomized Min-Cut Algorithm

Algorithm: Recursive-Randomized-Min-Cut(G, α)

Input

- Undirected multigraph G with n vertices
- Integer constant $\alpha > 0$

Output

- Cut C

Step-by-Step Algorithm

Step 1:

If

$$n \leqslant \alpha^3$$

then

$$C \leftarrow \text{min-cut found using brute force}$$

Step 2:

Else repeat for

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

Step 2.1:

Construct

$$G'$$

by applying

$$n - \lceil \frac{n}{\sqrt{\alpha}} \rceil$$

random contractions

Step 2.2:

Compute

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

Step 2.3:

If

$$i = 1 \text{ OR } |C'| < |C|$$

then

$$C \leftarrow C'$$

Step 3:

Return

$$C$$

---

9. Comparison: Deterministic vs Randomized Min-Cut

Deterministic Min-Cut

Properties:

- Guarantees exact minimum cut
- Higher time complexity for large graphs
- Time complexity:

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

Randomized Min-Cut

Properties:

- Provides approximate minimum cut with high probability
- Faster algorithm
- Time complexity:

$$O(V^2)$$

---

10. Theorem: Probability of Finding Minimum Cut

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

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

where

n = number of vertices

---

11. Summary of Lecture

Key concepts covered:

1. Definition of cut-set and minimum cut
2. Edge contraction operation
3. Successful and unsuccessful min-cut runs
4. Max-Flow Min-Cut Theorem
5. Stoer-Wagner deterministic algorithm
6. Randomized min-cut algorithm
7. Recursive randomized algorithm pseudocode

8. Comparison between deterministic and randomized methods
9. Probability theorem for minimum cut detection

If you want, I can also convert this into **exam-ready handwritten style notes**, or add **step-by-step worked examples exactly like the lecture diagrams** for revision.