Part 13 Global Min-Cut Problem

(**Definition 1**) Given an undirected graph G=(V, E), find a cut (A, V-A) such that the number of edges across the cut is minimum.

(**Algorithm 1**) Let n=|V| be the number of vertices. There're n^2 possible pairs of vertices. Treat each pair of vertices as the source and sink vertex in a flow network. Run the **F-F Algorithm** (for Max Flow) to find the **minimum cut** of such a pair. Check the minimum cut of each s-t pair in the n^2 possible cases.

Notes: In the flow network reduced by a specific s-t pair, the maximum flow value should be at most |V|, since the capacities of all the edges are 1. In each iteration of the **F-F Algorithm**, the value of flow increases at least 1, so the number of iterations at most |V|. In each iteration, we apply the BFS to find an augmenting path, so the complexity is within $O(|E|)$. Hence, the total time of the **F-F Algorithm** is within $O(|V||E|)$. Since there're n^2 possible pairs, the total time of **Algorithm 1** is within $O(|V|^3|E|)$.

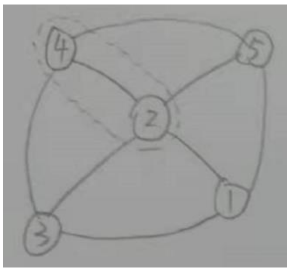
Notes: The complexity of **Algorithm 1** can be further improved, by only considering the selection of the sink (or source) vertex.

(**Algorithm 2**) Select one vertex as the source. Treat each of the rest vertices as the sink respectively, which reduce the original graph to (n-1) flow networks. Run the **F-F Algorithm** to each of the reduced flow network to find the **minimum cut**. Check the min-cut of the (n-1) reduced flow networks.

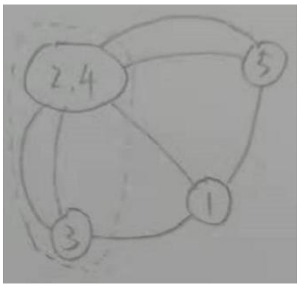
Notes: Compared with **Algorithm 1**, the complexity of **Algorithm 2** is reduced to $O(|V|^2|E|)$.

(**Algorithm 3**) (**Edge-Contraction Algorithm**) Given a graph G=(V, E), randomly selected an edge (u, v) and merge (contract) the vertex u and v to as supervertex, which derive a multi-graph. Note that the multi-graph allows each vertex pair to have multiple edges. Repeat the procedure of merging vertices based on a selected edge, until there's only 2 (super)verties in the graph. The two supervertices are treated as two groups, i.e., the cut result. The edges between the two supervertices are the edges cross the cut.

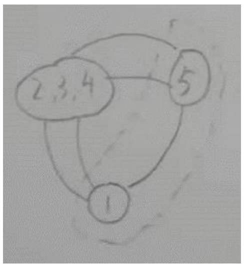
(**Example 1**) Consider the following graph and run **Algorithm 3**.



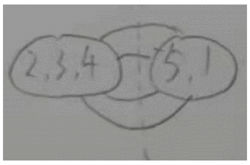
Initially, there're 8 edges. Each edge has the same probability (i.e., 1/8) to be selected. In iteration 1, suppose edge (2, 4) is selected. We can derive the following multi-graph.



Note that there're two edges between the vertex pair ({2, 4}, 5) (also similar to the pair ({2, 4}, 3)). There 7 edges in the derived multi-graph. Each edge has the same probability (i.e., 1/7) to be contracted. In iteration 2, suppose one of the two edges between {2, 4} and 3 is selected. We can derive the following multi-graph.



In iteration 3, suppose the edge (1, 5) is selected. We can derive the following multi-graph.



The algorithm stops the iteration, since there're only 2 supervertices. Because there're 4 edges between the 2 supervertices, we finally derive a cut with size 4.

(**Claim 1**) Let C denote a global min-cut of size k. The **degree** of each vertex should be at least (i.e., $\geq$) k.

Proof of **Claim 1**. Suppose there exist a vertex u with degree less than (i.e., $<$) k and the size of min-cut is k. There exists a cut ($\{u\}$, V-$\{u\}$) (where we put u in one side and the rest vertices in another side) with size less than k. It contradicts with the condition that the size of the min-cut is k. Hence, the degree of each vertex should be at least k.

(**Theorem 1**) The **probability** that the **Algorithm 3** generates a **global min-cut** is at least $\Omega(1/n^2)$, where n=$|V|$ is the number of vertices.

Proof of **Theorem 1**. Let C denote a global min-cut of size k. By **Claim 1**, the **degree** of each vertex should be at least (i.e., $\geq$) k. The number of edges should be at least (i.e., $\geq$) $nk/2$, i.e., the sum of degree is as twice as the number of edges.

Suppose an edge (u, v) isn't contracted by **Algorithm 3**. First, consider the probability that (u, v) is contracted in iteration 1, where we have

$$P[(u,v) \text{ is contracted in iteration } 1] \leq \frac{k}{nk/2} = \frac{2}{n} \cdot$$

Further, suppose (u, v) is not contracted in iteration 1. The probability that (u, v) is contracted in iteration 2 satisfiies

$$P[(u,v) \text{ is contracted in iteration 2}] \le \frac{k}{(n-1)k/2} = \frac{2}{n-1}.$$

Hence, the probability that (u, v) isn't contracted by **Algorithm 3** satisfies

$$\begin{aligned}
&P[(u,v) \text{ isn't contracted in all iterations}] \\
&\le (1-\frac{2}{n})(1-\frac{2}{n-1})\cdots(1-\frac{2}{3}) \\
&= \frac{n-2}{n}\cdot\frac{n-3}{n-1}\cdot\frac{n-4}{n-2}\cdots\frac{2}{4}\cdot\frac{1}{3} \\
&= \frac{2}{n(n-1)} = \Omega(\frac{1}{n^2})
\end{aligned}.$$

Notes: **Algorithm 3** is a polynomial-time algorithm. Although $\Omega(1/n^2)$ is a low probability, we can repeat **Algorithm 3** $\Theta(n^2)$ times, which can derive a global min-cut with the probability $O(1)$.