Homework 2 (Due Date: Friday 6 October)

Please write the following on your homework:

- Name
- Collaborators (write none if no collaborators)
- Source, if you obtained the solution through research, e.g. through the web.

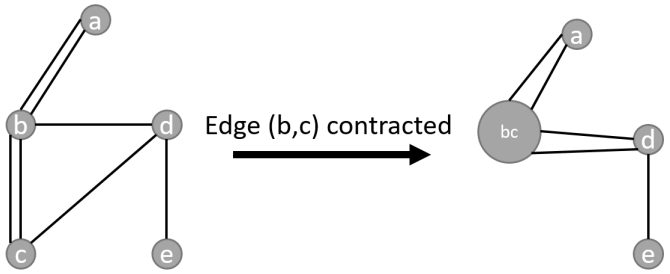
While you may collaborate, you *must write up the solution yourself* and you are asked to try doing the questions yourself first before discussing with others.

If you are asked to describe an algorithm, you should first provide a short description of the problem and what your main results are. Then you should provide a description of the algorithm in English, and if helpful, in pseudo-code. An example, or diagram can often be helpful in helping describe the algorithm. You should provide a proof (or indication) of correctness of the algorithm, and an analysis of the running time of the algorithm. Remember that you need to communicate clearly. Grades cannot be awarded if the grader is unable to understand what you are writing.

1 Karger's Algorithm

Given a graph $G = (E, V)$ with at least 2 nodes, a **cut** is a partition of the vertices of the graph into two disjoint (non-empty) subsets. For example, let $V = \{1, 2, 3, 4, 5, 6\}$, then $V_1 = \{1, 3, 5\}; V_2 = \{2, 4, 6\}$ is a cut of the graph. Notice each group needs to be non-empty and they have no intersection. A **min-cut** of the graph is a cut (V_1, V_2) where the number of edges across V_1 and V_2 is minimized.

Edge contraction is a graph operation. When an edge in a graph is *contracted*, the two nodes of the edge is merged to one node, and the edges with same endpoints of the contracted edge are removed.



In the above example, when contraction happens along an edge (b, c), all the edge with endpoints b, c are removed and b, c are merged into one point bc.

Finding min-cut in a graph is useful in solving network problems. For example, in wartime the min-cut of transportation network of hostile states is determined, so the military force could disrupt the supply with minimal effort. For ISPs, identify the min-cut of their network helps to locate the bottleneck.

Here is a randomized algorithm used to find the minimum cut size of a graph (Karger's Algorithm):

KARGER(E, V)

```
1  mincut ← INT_MAX
2  for k times
3      Make a copy G' = (V', E') ← G
4      while |V'| > 2
5          Contract along a random edge e ∈ E'
6      update mincut ← min(mincut, |E'|)
7  return mincut
```

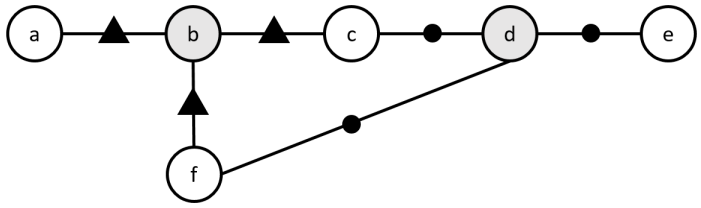
1. Analyze the success probability of each while loop
   (a) Show that the size of the min-cut of a graph is no more than the degree of any node in the graph. (The degree of a node is the number of edges connected to the node)
   (b) Show that the sum of the degree of all nodes in a graph is $2|E|$. Hence or otherwise, show that some node must have degree at most $\frac{2|E|}{|V|}$
   (c) Fix a min-cut of the graph, show that when a random edge is picked in the edge set, the probability that the edge is across the min-cut of current graph is at most $\frac{2}{|V|}$
   (d) Show that after a contraction:
      i. if the contracted edge is crossing the min-cut, the size of min-cut in the new graph is not smaller than the size of the min-cut of the old graph.
      ii. otherwise, the size of min-cut in the new graph is the same as the size of min-cut of the old graph.
   (e) Show that line 3-6 correctly update the variable min-cut to the size of min-cut of the graph G with probability at least $\frac{2}{|V|(|V|-1)}$
2. Estimate the asymptotic number of runs needed to achieve low error rate
   (hint: For all $x > 1$, we have $(1 - x^{-1})^x < e^{-1}$)
   (a) Given a coin of head probability p, find the probability that we have n continuous tosses of tail.
   (b) Using the result of the previous question or otherwise, show that in order to have the error probability of the algorithm to be less than α, setting $k = O(|V|^2 \log \alpha^{-1})$ would be sufficient. (k is the number of times we repeat the algorithm, as written in line 1)

2 Min Vertex-Cover

Given a graph $G = (E, V)$, a **vertex cover** is a set of nodes $C \subseteq V$ such that for each edge in E, at least one endpoint of the edge is in C. You have encountered *vertex-cover* in HW1. Find a vertex-cover with minimum number of nodes is NP-hard, however, we may use randomized algorithm to find an approximate solution.

VERTEX-COVER(E, V)
1 $C = \emptyset$
2 **for** $e \in E$
3 **if** e is not covered by C
4 put a randomly selected endpoint of e into C
5 **return** C

Let the Min-Vertex Cover be C^* and the return value of the algorithm be C. Partition the edges in E as follows: for each edge, assign it to one of the vertices v_i^* in C^* that covers the edge. Let P_i be the partition corresponding to v_i^*



In the above example, $\{b, d\}$ is a min vertex cover, where the edges labelled by triangle are included in P_b, and the edges labelled by circle are included in P_d.

1. Let n_i be the number of edges belonging to P_i that are examined in line 4 before all the edges in P_i has been covered. Argue that $|C| = \sum_{i=1}^{|C^*|} n_i$
2. Show that $E[|C|] < 2|C^*|$