Exercise 1. \hspace{1cm} \textbf{Better randomized algorithm for Min Cut}

Consider the following recursive algorithm for Min Cut:

Procedure \texttt{FastMinCut}(\texttt{G} : an undirected multigraph)
\[
\begin{aligned}
n &:= |V(\texttt{G})|, \\
\text{if } n \leq 6 &\quad \text{return an optimal min-cut obtained by exhaustive search}, \\
\text{else } &\quad \text{\texttt{t} := } 1 + \left\lceil \frac{n}{\sqrt{2}} \right\rceil, \\
&\makebox[0.2cm]{\text{Make two copies } \texttt{G}_1 \text{ of } \texttt{G}.,} \\
&\makebox[0.2cm]{\text{Iteratively apply } n-\texttt{t} \text{ random edge contractions on } \texttt{G}_1 \text{ (removing loops but not multi-edges} \\
&\text{after each contraction) to obtain a multigraph } \texttt{H}_1 \text{ with } \texttt{t} \text{ vertices}.} \\
&\makebox[0.2cm]{\text{Iteratively apply } n-\texttt{t} \text{ random edge contractions on } \texttt{G}_2 \text{ (removing loops but not multi-edges} \\
&\text{after each contraction) to obtain a multigraph } \texttt{H}_2 \text{ with } \texttt{t} \text{ vertices}.} \\
&\text{return the best cut between } \texttt{FastMinCut}(\texttt{H}_1) \text{ and } \texttt{FastMinCut}(\texttt{H}_2). \nonumber
\end{aligned}
\]
end if

1. Show that we apply a random edge contraction in $O(n)$. You are free to pick the data structure used to represent multigraphs (as long as it supports all operations here).
2. Show \texttt{FastMinCut} runs in $T(n) = O(n^2 \log n)$. (First give a short argument that it terminates.)
3. Show this algorithm uses at most $M(n) = O(n^2)$ memory.
4. Show that the probability that a min cut survives $n-\texttt{t}$ random edge contractions is at least $1/2$ when $n > 6$.
5. Let $P(k)$ be the probability that the algorithm outputs a min cut for a multi-graph that requires $k$ levels of recursions ($k = \Theta(\log n)$ if $G$ has $n$ vertices).

Show that $P(k) \geq p(k)$ where $p$ is defined recursively as $p(0) = 1$ and $p(k + 1) = p(k) - \frac{p(k)^2}{4}$.

6. Let $q(k) = \frac{4}{p(k)} - 1$ so that $q(0) = 3$ and $q(k + 1) = q(k) + 1 + \frac{1}{q(k)}$.

Show that for all $k \geq 0$, $0 < q(k) \leq k + H_{k-1} + 3$, where $H_k$ is the $k$th harmonic number $\sum_{i=1}^{k} \frac{1}{i}$.

7. Conclude that \texttt{FastMinCut} computes a min cut with probability $\Omega(1/\log n)$. Propose an algorithm which increases the success probability to $1 - 1/n^2$. Compare its time complexity to that of the best known deterministic algorithm (based on a max flow computation), $O(mn \log(m^2/n))$.

Exercise 2. \hspace{1cm} \textbf{Yao’s principle}

We want to minimize the energy consumption of a hard-drive. When the drive is on, it consumes $x$ per unit of time; when it is off it costs $0$, but it cost $y$ to turn it back on. Request to access the hard drive arrive at unknown times $t_1 < t_2 < \ldots$

1. Show that we can consider each interval $[t_i, t_{i+1}[$ independently and thus only need to solve the following problem.

At $t = 0$, the drive is on and we want to serve a request at time $t$ where $t$ is not known beforehand.

2. Show that the only deterministic algorithms to consider are $A_d$, for $d \geq 0$ : if the request arrives before $t = d$, serves it; otherwise, turn off the drive at time $d$ and turn it back on when the request arrives.

3. We denote by $I_t$ the instance where the request arrive at time $t$ and by $\text{OPT}(I_t)$ the optimal energy consumption to serve a request arriving at time $t$.

Show that:

\[
(A_d(I_t)) = \begin{cases} 
xt & \text{if } t \leq d \\
xd + y & \text{if } t > d 
\end{cases}
\]

and $\text{OPT}(I_t) = \min(\xt, y)$.
4. We define the competitive ratio of $A_d$ as $C_d = \max_{t \geq 0} \left( \frac{A_d(I_t)}{\text{OPT}(I_t)} \right)$. Show that the optimal deterministic competitive ratio is $C_{d/2} = 2$.

5. We now turn to randomized algorithms. Any randomized algorithm can be seen as a distribution $p : R \to R$ over deterministic algorithms: $A_p$ is the algorithm that runs $A_d$ “with probability $p(\delta)$” (more precisely, $A_p$ runs algorithm $A_\delta$ with $\delta \geq d$ with probability $\int_0^\infty p(\delta) d\delta$). Then, $\left[ (A_p(I_t)) \right] = \int_0^\infty (A_\delta(I_t)) p(\delta) d\delta$ and the randomized competitive ratio is $C_p = \max_{t \geq 0} \frac{\left[ (A_p(I_t)) \right]}{\text{OPT}(I_t)}$. Our goal is to find an optimal distribution $p^* = \arg\min_{p : R_n \to R_n : \sum_{t=1}^\infty p(\delta) d\delta = 1} \max_{t \geq 0} \frac{\left[ (A_p(I_t)) \right]}{\min(xt, y)}$. Note that we assume that the adversary choosing the instance $t$ does not known our random choice, otherwise we would be back to the deterministic case.

**Zero-sum games.** _Zero-sum games_ are defined by a cost matrix $C = (c_{d,t})_{d=1..D, t=1..T}$. At each round, the line player chooses a line $d$ and (independently) the column player chooses a column $t$ and the line player pays $c_{d,t}$ to the column player (each player earns what the other loses). Now, an optimal deterministic strategy for the line player is a line $d^{*} \in \arg\min_d \max_t c_{d,t}$ whereas an optimal deterministic strategy for the column player is a column $t^{*} \in \arg\max_t \min_c c_{d,t}$.

It turns out that an optimal strategy does not always exist. So we now focus on randomized strategies: assume that the line player chooses a line $d \in \{1, \ldots, D\}$ with probability $p_d$ and that the column player chooses a column $t \in \{1, \ldots, T\}$ with probability $q_t$. Let us see $p$ and $q$ as line and column vectors of dimension $D$ and $T$ respectively.

Show that the expected cost paid by the line player to column player is $p^T C q = \sum_{d=1}^D \sum_{t=1}^T p_d c_{d,t} q_t$.

6. An optimal randomized strategy $p^*$ for the line player must minimize the worst cost to the worst possible distribution of the column player:

$$p^* = \arg\min_{p \geq 0 : \|p\|_1 = 1} \max_{q \geq 0 : \|q\|_1 = 1} p^T C q$$

whereas an optimal randomized strategy $q^*$ for the column player must maximize the cost to the best possible distribution of the line player:

$$q^* = \arg\max_{q \geq 0 : \|q\|_1 = 1} \min_{p \geq 0 : \|p\|_1 = 1} p^T C q$$

Von Neumann showed that there always exists a dominating randomized strategy for both players, and the cost of these two strategies always match! (the minimax theorem is in fact equivalent to the duality theorem in linear programming):

$$\min_{p \geq 0 : \|p\|_1 = 1} \max_{q \geq 0 : \|q\|_1 = 1} p^T C q = \max_{q \geq 0 : \|q\|_1 = 1} \min_{p \geq 0 : \|p\|_1 = 1} p^T C q$$

Show the easy direction of this equality, that is the inequalities:

$$\min_{p \geq 0 : \|p\|_1 = 1} \max_{q \geq 0 : \|q\|_1 = 1} p^T C q = \max_{q \geq 0 : \|q\|_1 = 1} \min_{p \geq 0 : \|p\|_1 = 1} p^T C q$$

where $C_d$ and $C_t$ denote respectively the $d$-th row and the $t$-th column of the cost matrix.

7. Show that the expected competitive ratio of the randomized algorithm $A_{p^*}$ is independent of $t$ and that the expected competitive ratio for distribution of instance $q^*$ is independent of the deterministic algorithm and conclude that $A_{p^*}$ is optimal. What is the best expected competitive ratio achievable by a randomized algorithm?