CSC 2420 Spring 2019, Assignment 2
Due date: April 4

It is certainly preferable for you to solve the questions without consulting a published source. However, if you are using a published source then you must specify the source and you should try to improve upon the presentation of the result.

If you would like to discuss any questions with someone else that is fine BUT at the end of any collaboration you must spend at least one hour playing video games or watching two periods of Maple Leaf hockey or maybe even start reading a good novel before writing anything down.

Unless stated otherwise, all subquestions (1(i), 1(ii), 1(iii), 1(iv), etc) are worth 10 points. If you do not know how to answer a question, state "I do not know how to answer this (sub) question" and you will receive 20% (i.e. 2 of 10 points) for doing so. You can receive partial credit for any reasonable attempt to answer a question BUT no credit for arguments that make no sense.

In class I can clarify any questions you may have about this assignment.

1. Show that if there is an algorithm (either deterministic or randomized) achieving competitive ratio ρ for BMM, then there is a *greedy* algorithm (respectively, deterministic or randomized) achieving competitive ratio at least ρ.

2. Consider the implementation (as an online algorithm) of the "double sided greedy algorithm" (for the USM problem) applied to the explicitly defined maximum directed cut (DICUT) problem. Here the online input items are the nodes of the directed graph. In particular, what information must be provided in the representation of each node?

3. Consider the problem of online stochastic maximum edge-weighted bipartite matching in the known i.i.d. setting. We are given an edge-weighted *type graph* $G = (U, V, E)$ and a distribution $\mathcal{D}$ over the online nodes V. Let $|U| = k$, $|V| = n$, $w(e)$ be the weight of edge e, and assume $\mathcal{D}$ is the uniform distribution. During the online phase, n i.i.d. draws of nodes $v \sim V$ arrive and we must assign v to a node $u \in U$ or not assign it at all. A node in U can only be assigned once. Let $N(a)$ be the set of edges adjacent to node a.

   The edge weighted bipartite matching problem can be formalized as follows:

   Maximize $\sum_{e \in E} w(e) \cdot p_e$

   subject to $\quad \sum_{e \in N(u)} p_e \leq 1 \quad \forall u \in U$

   $\qquad \sum_{e \in N(v)} p_e \leq 1 \quad \forall v \in V$

   $\qquad p_e \in \{0, 1\}$

   Consider the LP relaxation (i.e. $p_e \in [0, 1]$) of the IP and let p^* be the optimal solution to this LP.

   Consider the following algorithm: Whenever an online node v arrives, choose an edge $e(u, v) \in N(v)$ with probability p_e^* and if u is not already matched, include (u, v) in the matching.

   Show that this algorithm achieves a competitive ratio of $1 - \frac{1}{e}$.

4. As suggested in the lecture slides for week 10 (see slides 10 and 11), we can potentially obtain a relatively simple proof for a competitive ratio for the secretary problem by considering more events that would result in the best candidate being chosen.

   - By counting permutations, work out the precise values for $P[A^1|B^1$ and $P[B^1]$ as defined in the slides.
   - This question is somewhat open-ended so do not spend too much time on this. To what extent can you use a similar type of argument (using just basic discrete probability) to obtain a competitive ratio for the secrtary problem that is better than $\frac{1}{4}$? That is, consider sampling the first n/a (e.g. for $a = 2$ or maybe $a = 5/2$) candidates before looking for a better candidte and then explore some of the events that would result in finding the optimal candidate.

5. Consider an input sequence $a_1, \ldots, a_n$ with $a_i \in \{1, 2, \ldots, N\}$ and assume $n >> N$. Using a 2-pass streaming algorithm, determine if the input stream contains an element a such that $f(a) > n/2 - 10$ where $f(a)$ is the number of times the element a occurs in the input stream. What is the space and time complexity of your algorithm in temrs of n and N?

   Hint: Look at the Misra-Gries algorithm for the "Frequent problem" in the Chakrabarti Lecture Notes which have been posted although the notation is slightly different.