Due: Tuesday, April 25, 2023, 11:59pm

Submit online at (clickable link): `https://forms.gle/oaWgXELv3szYUZMh7`

1 Class Scheduling

At some universities, class scheduling proceeds in a different manner than at UMBC. One process starts with each department producing a list of classes that will be offered the next semester. (This list just has the classes offered and does not include any information about timing.) Next, the students indicate which classes they will take. Finally, the university Registrar assigns one of the standard time slots to each class.

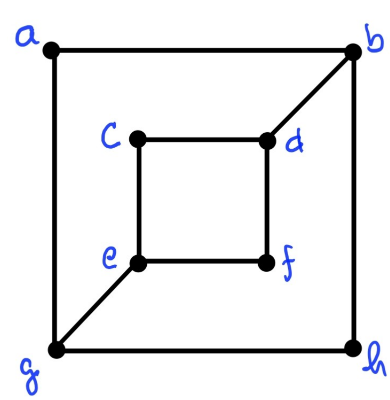
Suppose that there are n classes, m students, t standard time slots and that each student is allowed to sign up for at most k classes. The Registrar wants to determine if it is possible to assign a standard time slot to each class so that none of the students have a scheduling conflict — i.e., none of the students have two of the classes they want to take assigned to the same time slot.

Below, you will show that the Registrar's Class Scheduling Problem (RCSP) is NP-complete by selecting one of the problems we have shown NP-complete during lecture and reducing that problem to RCSP.

1. Pick X to be one of 3SAT, 3-Color, k-Clique, Vertex Cover, Partition, Traveling Salesman or 3-Dimensional Matching. Devise and describe a polynomial-time function f that takes an instance of X and produces an instance of RCSP. Do briefly justify that your function f runs in polynomial time.
2. Prove that for all instances x of X that if $x \in X$ then $f(x) \in$ RCSP.
3. Prove that for all instances x of X that if $f(x) \in$ RCSP then $x \in X$.

2 Lucky Cover

The performance of the 2-approximation algorithm for Vertex Cover is very much dependent on which edges are chosen, since the endpoints of the chosen edge are added to the approximate vertex cover. Consider the graph below.



Questions:

1. What is the best possible sequence of edges that the algorithm might have picked? (Here best means the resulting vertex cover is smallest possible for the algorithm.)

2. What is the worst possible sequence of edges that the algorithm might have picked? (Similarly, worst means the resulting vertex cover is the largest possible for the algorithm.)

3 Approximating Three-Dimensional Matching

In the three-dimensional matching problem (3DM) we are given three sets X, Y and Z such that the sizes of the three sets are the same. Let $m = |X| = |Y| = |Z|$. We are also given a set of triples $T \subseteq X \times Y \times Z$. The NP-complete problem we discussed in class asks if there exists a *perfect matching* — that is, a subset of triples $T' \subseteq T$ such that $|T'| = m$ and every $x \in X$ appears in a triple in T' exactly once, every $y \in Y$ appears in a triple in T' exactly once and every $z \in Z$ appears in a triple in T' exactly once. A *matching* $M \subseteq T$ (not necessarily a perfect one) just requires that every $x \in X$ appears in a triple in M at most once, every $y \in Y$ appears in a triple in M at most once and every $z \in Z$ appears in a triple in M at most once. So, a matching enforces that triples in M do not share components, but does not require that every x, y and z appear in some triple in M. The optimization version of 3DM asks us to find a matching that contains the largest number of triples. We are still required to make sure that none of the selected triples overlap. For example, we are not allowed to select two triples (x, y_1, z_1) and (x, y_2, z_2) that share the same value in the first component.

Assignment: Consider the following naïve (some might say stupid) approximation algorithm for 3DM: Pick an arbitrary triple (x, y, z) from T. Then, remove from T any triple that contains x, y or z. We continue picking an arbitrary triple in this manner until T is empty. Argue that this naïve algorithm achieves an approximation factor of 3.