Problem 1. Consider a complete bipartite graph $G = (V, E)$:

- V has $2n$ vertices, including n black vertices and n white vertices.
- E has n^2 edges, including an edge between every black vertex and every white vertex.

Use G to explain why 2 is the best approximation ratio that we can prove for the vertex cover approximation algorithm.

Proof. An optimal solution is to include all the n black vertices. However, our approximation algorithm selects all $2n$ vertices, meaning that the approximation ratio has to be at least 2. □

Problem 2. Let $G = (V, E)$ be an input graph to the vertex cover problem. If G is a tree, describe an $O(|V|)$ time algorithm that finds an optimal vertex cover of G.

Solution. Root the tree G at an arbitrary node. For each node of the tree, define $T(u)$ as the subtree rooted at u. In addition, define $\mathrm{OPT}(u, yes)$ as the size of an optimal vertex cover of u, provided that u belongs to the vertex cover; and $\mathrm{OPT}(u, no)$ as the size of an optimal vertex cover of $T(u)$, provided that u does not belong to the vertex cover.

If u is a leaf, then let $\mathrm{OPT}(u, yes) = 1$ and $\mathrm{OPT}(u, no) = 0$. Otherwise, u is an internal node, and

$$\mathrm{OPT}(u, yes) = 1 + \sum_{\text{child } v \text{ of } u} \min\{\mathrm{OPT}(v, yes), \mathrm{OPT}(v, no)\},$$

$$\mathrm{OPT}(u, no) = \sum_{\text{child } v \text{ of } u} \mathrm{OPT}(v, yes).$$

With dynamic programming, compute $\mathrm{OPT}(root, yes)$ and $\mathrm{OPT}(root, no)$ in a bottom-up order, which can be done in $O(|V|)$ time. Then, the optimal vertex cover size is $\min\{\mathrm{OPT}(root, yes), \mathrm{OPT}(root, no)\}$. The optimal vertex cover can also be found by the piggyback technique in $O(|V|)$ time.

Problem 3. Prof. Goofy proposes the following algorithm to find a vertex cover of $G = (V, E)$.

```
procedure ApproxMaxDegVC(G)
    S ← ∅
    while E is not empty do
        v ← a vertex with the maximum degree in the current G
        add v to S
        remove from E all the edges of v
    end while
end procedure
```

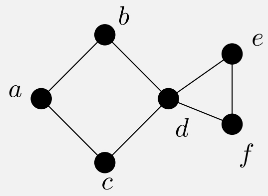
Show that the approximation ratio of this algorithm is greater than 2.

Proof. Construct a bipartite graph, where A is the set of n vertices, and $B_2, \ldots, B_n$ be sets of vertices where B_i has the size $|B_i| = \lfloor n/i \rfloor$. Each vertex in A is connected to exactly one vertex of B_i, for all i. Also, let the edges into B_i be uniformly distributed so that the degree of any vertex in B_i is as close to i as possible. Then, note that the degree of the vertex in B_n is n and the degree of the vertices in part A is $n - 1$. Also, observe that B_n, with only one vertex, is the first to be removed, this subtracts one from the degree of each vertex in A. We then have forced the algorithm to choose all vertices in part B, which has the size $\sum_{i=2}^{n} \lfloor n/i \rfloor$, while the optimal solution is to choose the vertices from part A, which has the size n. Set $n = 16$ and part B has 34 vertices, while part A has only 16 vertices. The approximation ratio of this algorithm is hence greater than 2. □

Problem 4. Let $G = (V, E)$ be a simple undirected graph. Given a subset $S \subseteq V$, a *cut* induced by S is the set of edges $e \in E$ such that e has a vertex in S and another vertex in $V \backslash S$. Let OPT_G be the maximum size of a cut that can be induced by any $S \subseteq V$. Design a poly($|V|$)-time algorithm that returns a cut of size at least $\mathrm{OPT}_G/2$ in expectation.

Solution. Start with an empty S. For each vertex $u \in V$, add u to S with probability $1/2$. Each edge $\{u, v\} \in E$ contributes to the cut induced by S with probability $1/2$. Hence, in expectation, the cut has size $|E|/2$ in expectation, which is at least $\mathrm{OPT}_G/2$ due to $|E| \geq \mathrm{OPT}_G$.

Problem 5. Consider the undirected graph G below.



(a) What is the size of a smallest vertex cover of G.

(b) Is it possible for our vertex cover algorithm to output a vertex cover of size 4?

(c) How about size 6?

Solution.

(a) 3. Suppose that there is a smaller vertex cover of G, with size 2. Then, d must be in the cover, since all other vertices have degree of at most 2, which is impossible to cover all 7 edges by only selecting two vertices. By a similar argument, two extra vertices are necessary for covering the rest of the edges, combining with the fact that d is chosen, which is a contradiction.

(b) First, pick $\{d, e\}$ and eliminate $\{d, e\}, \{d, f\}, \{d, b\}, \{d, c\}, \{e, f\}$. Then, pick $\{a, b\}$ and eliminate $\{a, b\}, \{a, c\}$. Now, E is empty. The vertex cover discovered is a, b, d, e.

(c) First, pick $\{a, b\}$ and eliminate $\{a, b\}, \{a, c\}$. Then, pick $\{c, d\}$, and eliminate $\{c, d\}, \{d, b\}, \{d, e\}, \{d, f\}$. Finally, $\{e, f\}$ is selected and E is empty. This gives the vertex cover with vertices a, b, c, d, e, f.

Problem 6. Define "variable" and "literal" in the same way as we did for the MAX-3SAT problem. However, re-define a clause as the OR of an arbitrary number of literals subject to the constraint that all literals need to be defined on distinct variables. Prove: by independently setting each variable to 0 or 1 with 50% probability, we ensure that the clause should evaluate to 1 with probability at least 1/2.

Proof. Suppose that a clause contains a literal x_i and its negation $\bar{x}_i$. Then, the clause should tautologically evaluate to 1.

Hence, without loss of generality, consider any clause of length n, $z = z_1 \vee z_2 \vee \ldots \vee z_n$. Then, if the clause evaluates to zero, it must be that $z_1, z_2, \ldots, z_n$ are all zero. The probability for this case is $1/2^n$. Hence, the probability that the clause is evaluated to true is $1 - 1/2^n \geq 1/2$. $\square$