

CSCI 0500: Data Structures, Algorithms, and Intractability (Fall 2025)

Assignment 5

Due at 11:59pm ET, Monday, Nov 24

1. (1 point) In this question, we explore the connections between maximum matching, minimum vertex cover, and maximum independent set in bipartite graphs.

For Parts (b) and (d), you can use algorithms discussed in class without implementing them. You do not need to prove the correctness or analyze the runtime of your algorithms.

Definition 1 (Vertex Cover). A vertex cover of an undirected graph $G = (V, E)$ is a subset of nodes $C \subseteq V$ such that every edge in E has at least one endpoint in C . A minimum vertex cover is a vertex cover with the smallest possible size.

Theorem 1 (König's Theorem). *For any bipartite graph, the number of nodes in a minimum vertex cover is equal to the number of edges in a maximum matching.*

- (a) Fix any undirected graph G . Let M be a matching in G . Let C be a vertex cover of G . Prove that $|C| \geq |M|$. (Hint: Can one node in C cover multiple edges in M ?)

Let $G = (L \cup R, E)$ be a bipartite graph, where L and R are left and right nodes, respectively. Let M^* be a maximum matching in G . Let C^* be a minimum vertex cover of G .

Part (a) implies that $|C^*| \geq |M^*|$. To prove that $|C^*| \leq |M^*|$, we will construct a vertex cover C of size $|M^*|$, and hence $|C^*| \leq |C| = |M^*|$.

As discussed in class, we construct a directed graph $G' = (V', E')$ where $V' = V \cup \{s, t\}$, such that there is an s -to- t path in G' if and only if there is an augmenting path with respect to M^* in G . Recall that G' has the following edges:

- (s, u) for all unmatched $u \in L$.
- (v, t) for all unmatched $v \in R$.
- (u, v) for all $(u, v) \in (E \setminus M^*)$ with $u \in L$ and $v \in R$.
- (v, u) for all $(u, v) \in M^*$.

Lemma 1. *Let S be the set of nodes reachable from s in G' . Then, $C = (L \setminus S) \cup (R \cap S)$ is a vertex cover of G , and $|C| = |M^*|$.*

You are encouraged to try to prove this lemma, but it is not required for this assignment.

- (b) Two departments are jointly hosting a PhD visit day. Each student lists two faculty they would like to meet with: one from Department A and one from Department B. Faculty members can meet with any number of students.

Suppose the goal is to select the minimum number of faculty to participate so that every student gets to meet with at least one of the two faculty they requested. The input starts with two integers $f > 0$ and $s > 0$, where f is the number of faculty in each department, and s is the number of students. This is followed by s pairs of integers (a_i, b_i) , where student i requested faculty a_i in Department A and faculty b_i in Department B.

Design an algorithm that outputs the minimum number of faculty required. Briefly describe your algorithm. Your algorithm should run in $O(f^9 s^9)$ time.

(Hint: Reduce this problem to computing a minimum vertex cover in a bipartite graph.)

Definition 2 (Independent Set). An independent set of an undirected graph $G = (V, E)$ is a subset of nodes $I \subseteq V$ such that no two nodes in I are connected by an edge in E . A maximum independent set is an independent set with the largest possible size.

(c) Prove that C is a vertex cover of G if and only if $(V \setminus C)$ is an independent set of G .

Therefore, C is a minimum vertex cover if and only if $(V \setminus C)$ is a maximum independent set.

(d) Consider a $k \times k$ chessboard where some squares are blocked.

The goal is to place as many knights as possible in the unblocked squares so that no two knights can attack each other. The input starts with two integers $k > 0$ and $b \geq 0$, followed by b coordinates of the blocked squares.

Design an algorithm that outputs the maximum number of knights that can be placed. Briefly describe your algorithm. Your algorithm should run in $O(k^9 b^9)$ time.

(Hint: Reduce this to computing a maximum independent set in a bipartite graph.)

2. (1 point) In class, we showed that a maximum matching in a bipartite graph with n nodes and m edges can be computed in $O(mn)$ time.¹ In this question, we study the Hopcroft-Karp algorithm, which can compute a maximum bipartite matching in $O(m\sqrt{n})$ time.

The key idea is to find shortest augmenting paths, and more specifically, a maximal set of node-disjoint shortest augmenting paths.

Algorithm 1: Hopcroft-Karp Algorithm

Input : A bipartite graph G

Output: A maximum matching M of G

$M \leftarrow \emptyset$

while *there exists an augmenting path in G with respect to M* **do**

$\ell \leftarrow$ the length of a shortest augmenting path

$(P_1, \dots, P_k) \leftarrow$ a maximal set of length- ℓ node-disjoint augmenting paths

$M \leftarrow M \oplus (P_1 \cup \dots \cup P_k)$, where $\oplus$ is the XOR of two sets

return M

Lemma 2. *After each iteration of the while loop in Algorithm 1, the length of the shortest augmenting path increases by at least 2.*

¹We assume without loss of generality that $m = \Omega(n)$.

Lemma 3. *Each iteration of the while loop in Algorithm 1 can be implemented to run in $O(m)$ time.*

You are encouraged to try to prove these lemmas, but it is not required for this assignment.

- (a) Let M be a matching and let M^* be a maximum matching in G . Suppose the shortest augmenting path with respect to M in G has ℓ edges. Prove that $\ell \cdot (|M^*| - |M|) \leq n$. (Hint: Consider the graph $M \oplus M^*$. You can use any facts discussed in class without proving them.)
- (b) Use Lemma 2 and Part (a) to prove that the number of iterations of the while loop in Algorithm 1 is $O(\sqrt{n})$. (Hint: Show that $|M|$ must be close to $|M^*|$ after $O(\sqrt{n})$ iterations.)

Consequently, by Part (b) and Lemma 3, the runtime of Algorithm 1 is $O(m\sqrt{n})$.

3. (1 point) Recall that the implication graph of a 2-CNF formula with n variables and m clauses is a directed graph with $2n$ nodes and $2m$ edges. Each node represents a literal. For each clause $(u \vee v)$ where u and v are literals, we add two directed edges $(\bar{u}, v)$ and $(\bar{v}, u)$.

Consider the following 2-CNF formula:

$$(x_1 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee x_5) \wedge (x_2 \vee x_3) \wedge (\bar{x}_2 \vee x_4) \wedge (x_4 \vee \bar{x}_5) \wedge (\bar{x}_4 \vee \bar{x}_5) .$$

- (a) Draw the implication graph G of this formula.
- (b) List all strongly connected components of G .
- (c) Contract each strongly connected component into a single node and choose any of its literals as its representative. Draw the resulting directed acyclic graph.
- (d) Decide whether the given formula is satisfiable. That is, show a satisfying assignment or prove that no satisfying assignment exists.