CSCI 0500: Data Structures, Algorithms, and Intractability (Fall 2026)

Assignment 5

Due at 11:59pm ET, Friday, Oct 30

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

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

   We now give a proof sketch of Theorem 2. Let $G = (L \cup R, E)$ be a bipartite graph. Let M^* be a maximum matching in G. By Part (a), every vertex cover of G has size at least $|M^*|$. It remains to construct a vertex cover of size $|M^*|$.

   As discussed in class, we construct a directed graph G' with node set $L \cup R \cup \{s, t\}$ and the following edges:

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

   Recall that G' has an s-to-t path if and only if G has an augmenting path with respect to M^*.

   Lemma 3. *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 $|C| = |M^*|$.*

   Together with Part (a), this lemma proves Theorem 2. You are encouraged to prove Lemma 3, but it is not required for this assignment.

(b) Departments A and B are jointly hosting a PhD visit day. Each student requests to meet with two faculty members, one from each department. Each faculty member can meet with any number of students.

The goal is to select as few faculty members as possible so that every student can meet with at least one of the two they requested. The input starts with two integers $f > 0$ and $s > 0$, where f is the number of faculty members in each department and s is the number of students. These are followed by s pairs of integers (a_i, b_i), where student i requests faculty member a_i from Department A and b_i from Department B.

Design an algorithm that outputs the minimum number of faculty members 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 4 (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) Let $G = (V, E)$ be any undirected graph and $C \subseteq V$. 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 on the unblocked squares so that no two knights can attack each other. A knight attacks any square two rows and one column away, or one row and two columns away. The input starts with two integers $k > 0$ and $b \geq 0$, followed by the coordinates of the b 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 $O(k^9)$ time.

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

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

Algorithm 1: Hopcroft-Karp Algorithm

Input: A bipartite graph G
Output: A maximum matching M in G
$M \leftarrow \varnothing$
while G *has an augmenting path with respect to* M **do**
  $\ell \leftarrow$ the length of a shortest augmenting path
  $\{P_1, \ldots, P_k\} \leftarrow$ a maximal set of node-disjoint length-ℓ augmenting paths
  $M \leftarrow M \oplus (P_1 \cup \cdots \cup P_k)$, where $\oplus$ denotes symmetric difference
return M

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

The length of a path is the number of edges in it. The key idea is to augment along a maximal set of node-disjoint shortest augmenting paths in each iteration. Here, maximal means that no additional shortest augmenting path can be added while keeping the paths node-disjoint.

Lemma 5. *If a shortest augmenting path has length ℓ before an iteration of Algorithm 1, then every augmenting path afterward has length at least $\ell + 2$.*

Lemma 6. *Each iteration of Algorithm 1 can be implemented to run in $O(m)$ time.*

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

(a) Let M be a matching in G. Let M^* be a maximum matching in G. Suppose a shortest augmenting path with respect to M has length ℓ. Prove that $\ell \cdot (|M^*| - |M|) \leq n$.
(Hint: Consider the graph $M \oplus M^*$. You can use facts discussed in class without proving them.)

(b) Use Lemma 5 and Part (a) to prove that Algorithm 1 performs $O(\sqrt{n})$ iterations.
(Hint: Show that after $O(\sqrt{n})$ iterations, the current matching is within $O(\sqrt{n})$ edges of a maximum matching.)

By Part (b) and Lemma 6, Algorithm 1 runs in $O(m\sqrt{n})$ time.

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 $(\overline{u}, v)$ and $(\overline{v}, u)$.

Consider the following 2-CNF formula:

$$(\overline{x_1} \vee \overline{x_2}) \wedge (\overline{x_1} \vee \overline{x_3}) \wedge (x_1 \vee \overline{x_4}) \wedge (\overline{x_1} \vee \overline{x_5}) \wedge (x_2 \vee x_5) \wedge (x_3 \vee x_4).$$

(a) Draw the implication graph G of this formula.

(b) List all strongly connected components of G. Contract each component into a single node and choose any of its literals as its representative. Draw the resulting directed acyclic graph.

(c) Decide whether the given formula is satisfiable. That is, show a satisfying assignment or prove that no satisfying assignment exists.

(d) Add the clause $(x_1 \vee x_4)$ to the formula. Decide whether the new formula is satisfiable. That is, show a satisfying assignment or prove that no satisfying assignment exists.
(Hint: To prove that a 2-CNF formula is unsatisfiable, it suffices to find some i such that its implication graph has a path from x_i to $\overline{x_i}$ and a path from $\overline{x_i}$ to x_i.)