St. Francis Xavier University
Department of Computer Science

CSCI 550: Advanced Algorithms
Assignment 1
Due October 2, 2026 at 8:30am

Assignment Regulations.

- This assignment may be completed individually or in a group of two people. If you are collaborating on an assignment as a group, your group must submit exactly one joint set of answers.
- Please include your full name and email address on your submission. For groups, every member must include their full name and email address on the joint submission.
- You may either handwrite or typeset your submission. If your submission is handwritten, please ensure that the handwriting is neat and legible.

[10 marks] 1. Consider the *pebble game*: Alice and Bob are given an undirected graph $G = (V, E)$ and two distinguished vertices $s, t \in V$. They then do the following:

- A pebble is placed on vertex s.
- Alice goes first by moving the pebble from s to an adjacent vertex u.
- Bob then goes, moving the pebble from u to an adjacent vertex w.
- The pebble may be moved to any vertex at most once.
- Play goes back and forth until either the pebble is moved to vertex t, or no legal move is available.

The Pebble-Game problem asks: does Alice always have a winning strategy?

(a) Show that Pebble-Game is in PSPACE.

(b) Suppose we modify the problem so that Bob goes first. Is Pebble-Game still in PSPACE? Why or why not?

(c) Suppose we modify the problem in a different way so that G is a *directed acyclic* graph. Show that Directed-Pebble-Game is now in P.

[7 marks] 2. Recall the QSAT problem from our lectures. Here, we will consider a variant of the problem where the given Boolean formula has no negated variables.

Let $\phi(x_1, x_2, \ldots, x_n)$ be a Boolean formula in conjunctive normal form; that is, a Boolean formula of the form $C_1 \wedge C_2 \wedge \cdots \wedge C_m$, where each clause C_i is a disjunction of three variables. Furthermore, suppose ϕ is *monotone*; that is, each variable in ϕ is of the form x_i and not $\overline{x_i}$.

The Monotone-QSAT problem asks whether the formula $\exists x_1\ \forall x_2 \cdots \forall x_{n-1}\ \exists x_n\ \phi(x_1, x_2, \ldots, x_n)$ is true. Interestingly, while QSAT is in PSPACE, Monotone-QSAT is quite a bit easier to solve.

Give an algorithm that decides instances of Monotone-QSAT in polynomial **time** in terms of n.

[7 marks] 3. The Hitting-Set problem is defined as follows. Suppose $A = \{a_1, a_2, \ldots, a_n\}$ is a set, and let $B_1, B_2, \ldots, B_m$ be a collection of subsets of A. We say that a subset $H \subseteq A$ is a *hitting set* for the collection $B_1, B_2, \ldots, B_m$ if H contains at least one element from each subset B_i; in other words, H "hits" each of the subsets B_i.

Hitting-Set is well-known to be NP-complete, but much like we showed in lecture with Vertex-Cover, we can make this problem tractable in special cases. Suppose we want to determine whether

there exists a hitting set of size at most k, and furthermore, we know that each subset B_i contains at most c elements for some constant c.

Using ideas similar to those we used for VERTEX-COVER, describe an algorithm that solves HITTING-SET efficiently. Namely, just like our algorithm for VERTEX-COVER took time $O(2^k \cdot kn)$, your algorithm for HITTING-SET should take time $O(f(k, c) \cdot p)$, where f is an arbitrary function that depends on k and c and p is a polynomial expression. (You do not need to formally analyze your algorithm; a high-level justification of the runtime will suffice.)

[6 marks] 4. You have been recruited to help new students move into their residences at the beginning of the school year. The move-in process works in the following way: the moving truck arrives, containing n boxes each of weight w_1, w_2, ..., w_n. Outside of the residence building is a set of moving carts, each of which can hold K kilograms of weight. (You may assume w_i for all i as well as K are all natural numbers.)

You can stack multiple boxes onto one moving cart, as long as you do not exceed its weight limit of K kilograms. The residence staff have told you to minimize the number of moving carts you need to move all n boxes. However, you (as a clever algorithms student) realize this problem is NP-complete!

Fortunately, there is a greedy algorithm you can try. Start with an empty moving cart, and then begin piling boxes 1, 2, 3, ..., onto the cart until you reach a box that would make the cart exceed its weight limit. Then, send this cart into the residence and continue the process with another empty cart.

(a) This greedy algorithm gives us a solution, but not necessarily an optimal one. Give an example of a set of box weights $\{w_1, \ldots, w_n\}$ and a value K where the greedy algorithm does not use the minimum-possible number of moving carts.

(b) Show that the number of moving carts used by the greedy algorithm is within a factor of 2 of the minimum, for any set of box weights $\{w_1, \ldots, w_n\}$ and any value K, thereby showing that the greedy algorithm is a 2-approximation algorithm.