Write your answers in the separate answer booklet.

You have 180 minutes (after you get the answer booklet) to answer seven questions.

1. For each statement below, there are two boxes in the answer booklet labeled "Yes" and "No". Check "Yes" if the statement is ***always*** true and "No" otherwise, and give a ***brief*** (at most one short sentence) explanation of your answer. **Assume *P* ≠ *NP*.** If there is any other ambiguity or uncertainty about an answer, check "No". For example:

   - $x + y = 5$

     [X] Yes [ ] No Suppose $x = 3$ and $y = 4$.

   - 3SAT can be solved in polynomial time.

     [ ] Yes [X] No 3SAT is NP-hard.

   - If P = NP then Jeff is the Queen of England.

     [X] Yes [ ] No The hypothesis is false, so the implication is true.

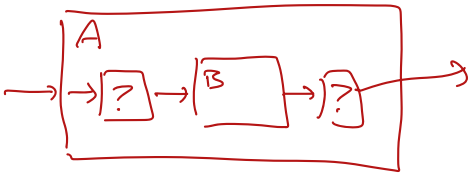
   Read each statement *very* carefully; some of these are deliberately subtle!

   (a) Which of the following statements are true for ***at least one*** language $L \subseteq \{0,1\}^*$?

   - $L^* = (L^*)^*$
   - L is decidable, but L^* is undecidable.
   - L is neither regular nor NP-hard.
   - L is in P, and L has an infinite fooling set.
   - The language $\{\langle M \rangle \mid M \text{ accepts } L\}$ is undecidable. Rice?

   (b) Suppose there is a ***polynomial-time*** reduction from some language A over the alphabet $\{0,1\}$ to some other language B over the alphabet $\{0,1\}$. Which of the following statements are ***always*** true, assuming P ≠ NP?

   - A is a subset of B.
   - If $B \in P$, then $A \in P$.
   - If B is NP-hard, then A is NP-hard.
   - If B is regular, then A is regular.
   - If B is regular, then A is decidable.



Problems 2–7 are on the following pages.

2. ***More of the same.*** For each statement below, there are two boxes in the answer booklet labeled "Yes" and "No". Check "Yes" if the statement is ***always*** true and "No" otherwise, and give a ***brief*** (at most one short sentence) explanation of your answer. **Assume $P \neq NP$.** If there is any other ambiguity or uncertainty about an answer, check "No".

   (a) Which of the following statements are true?

   - The solution to the recurrence $T(n) = 8T(n/2) + O(n^2)$ is $T(n) = O(n^2)$.
   - The solution to the recurrence $T(n) = 2T(n/8) + O(n^2)$ is $T(n) = O(n^2)$.
   - Every directed acyclic graph contains at least one sink. no outedges
   - Given *any* undirected graph G, we can compute a spanning tree of G in $O(V+E)$ time using whatever-first search.
   - Suppose $A[1..n]$ is an array of integers. Consider the following recursive function:

$$What(i,j) = \begin{cases} 0 & \text{if } i < 0 \text{ or } i > n \\ 0 & \text{if } j < 0 \text{ or } j > n \\ \max \begin{Bmatrix} What(i, j-1) \\ What(i-1, j) \\ A[i] \cdot A[j] + What(i+1, j+1) \end{Bmatrix} & \text{otherwise} \end{cases}$$

     We can memoize this function into an array $What[0..n, 0..n]$ in $O(n^2)$ time, by increasing i in the outer loop and increasing j in the inner loop.

   (b) Consider the following pair of languages:

   - DirHamPath $:= \{G \mid G \text{ is a directed graph with a Hamiltonian path}\}$
   - Acyclic $:= \{G \mid G \text{ is a directed acyclic graph}\}$

   (For concreteness, assume that in both of these languages, graphs are represented by their adjacency matrices.) Which of the following statements are true, assuming P $\neq$ NP?

   - Acyclic $\in$ NP
   - Acyclic $\cap$ DirHamPath $\in$ P
   - DirHamPath is decidable.
   - A polynomial-time reduction from DirHamPath to Acyclic would imply P=NP.
   - A polynomial-time reduction from Acyclic to DirHamPath would imply P=NP.

Problems 3–7 are on the following pages.

3. Describe and analyze an algorithm to determine whether the language accepted by a given DFA is finite or infinite. You can assume the input alphabet of the DFA is $\{0, 1\}$. *[Hint: DFAs are directed graphs.]*

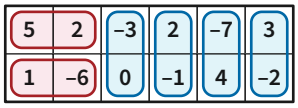
4. Suppose you are asked to tile a $2 \times n$ grid of squares with dominos (1×2 rectangles). Each domino must cover exactly two grid squares, either horizontally or vertically, and each grid square must be covered by exactly one domino.

   Each grid square is worth some number of points, which could be positive, negative, or zero. The ***value*** of a domino tiling is the sum of the points in squares covered by vertical dominos, ***minus*** the sum of the points in squares covered by horizontal dominos.

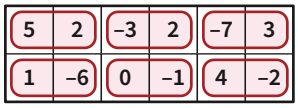
   Describe an algorithm to compute the largest possible value of a domino tiling of a given $2 \times n$ grid. Your input is an array $Points[1..2, 1..n]$ of point values.

   As an example, here are three domino tilings of the same 2×6 grid, along with their values. The third tiling is optimal; no other tiling of this grid has larger value. Thus, given this 2×6 grid as input, your algorithm should return the integer 16.

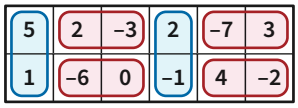
5	2	−3	2	−7	3
1	−6	0	−1	4	−2



value = −6



value = 2

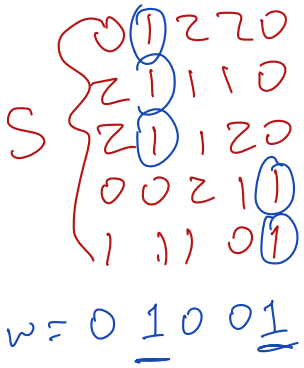


value = 16

5. ***Prove*** that the following problem (which we call MATCH) is NP-hard. The input is a finite set S of strings, all of the same length n, over the alphabet $\{0, 1, 2\}$. The problem is to determine whether there is a string $w \in \{0, 1\}^n$ such that for every string $s \in S$, the strings s and w have the same symbol in at least one position.

   For example, given the set $S = \{01220, 21110, 21120, 00211, 11101\}$, the correct output is TRUE, because the string $w = 01001$ matches the first three strings of S in the second position, and matches the last two strings of S in the last position. On the other hand, given the set $S = \{2002, 2112, 2012, 2102\}$, the correct output is FALSE.

   [Hint: Describe a reduction from SAT (or 3SAT)]



Problems 6 and 7 are on the next page.

6. Suppose you are given a height map of a mountain, in the form of an $n \times n$ grid of evenly spaced points, each labeled with an elevation value. You can safely hike directly from any point to any neighbor immediately north, south, east, or west, but only if the elevations of those two points differ by at most Δ. (The value of Δ depends on your hiking experience and your physical condition.)

   Describe and analyze an algorithm to determine the longest hike from some point s to some other point t, where the hike consists of an uphill climb (where elevations must increase at each step) followed by a downhill climb (where elevations must decrease at each step). Your input consists of an array $Elevation[1..n, 1..n]$ of elevation values, the starting point s, the target point t, and the parameter Δ.

7. Recall that a ***run*** in a string $w \in \{0,1\}^*$ is a maximal substring of w whose characters are all equal. For example, the string `00011111110000` is the concatenation of three runs:

$$\texttt{00011111110000} = \texttt{000} \bullet \texttt{1111111} \bullet \texttt{0000}$$

   (a) Let L_a denote the set of all strings in $\{0,1\}^*$ where every `0` is followed immediately by at least one `1`.

   For example, L_a contains the strings `010111` and `1111` and the empty string ε, but does not contain either `001100` or `1111110`.

   - Describe a DFA or NFA that accepts L_a ***and***
   - Give a regular expression that describes L_a.

   (You do not need to prove that your answers are correct.)

   (b) Let L_b denote the set of all strings in $\{0,1\}^*$ whose run lengths are increasing; that is, every run except the last is followed immediately by a *longer* run.

   For example, L_b contains the strings `0110001111` and `1100000` and `000` and the empty string ε, but does not contain either `000111` or `100011`.

   Prove that L_b is not a regular language.

CS/ECE 374 A ✧ Fall 2025

Practice Final Exam 1

December 7, 2023

Name:	Jeff E
NetID:	jeffe

- ***Don't panic!***
- You have 180 minutes to answer seven questions. The questions are described in more detail in a separate handout.
- If you brought anything except your writing implements, your two **hand-written** double-sided 8½" × 11" cheat sheets, and your university ID, please put it away for the duration of the exam. In particular, please turn off and put away *all* medically unnecessary electronic devices.
- Please clearly print your name and your NetID in the boxes above.
- Please also print your name at the top of every page of the answer booklet, except this cover page. We want to make sure that if a staple falls out, we can reassemble your answer booklet. (It doesn't happen often, but it does happen.)
- Greedy algorithms require formal proofs of correctness to receive any credit, even if they are correct. Otherwise, proofs or other justifications are required for full credit if and only if we explicitly ask for them, using the word ***prove*** or ***justify*** in bold italics.

- **Please do not write outside the black boxes on each page.** These indicate the area of the page that our scanners can actually scan. If the scanner can't see your work, we can't grade it.
- If you run out of space for an answer, please use the overflow/scratch pages at the back of the answer booklet, but **please clearly indicate where we should look**. If we can't find your work, we can't grade it.
- **Only work that is written into the stapled answer booklet will be graded**. In particular, you are welcome to detach scratch pages from the answer booklet, but any work on those detached pages will not be graded. We will provide additional scratch paper on request, but any work on that scratch paper will not be graded.
- Breathe in. Breathe out. You've got this.

CS/ECE 374 A ✧ Fall 2025 Practice Final 1 Problem 1	Name:

(a) Which of the following statements are true for ***at least one*** language $L \subseteq \{0,1\}^*$?

- $L^* = (L^*)^*$

  [X] Yes [ ] No

  (Foo·bar)·(baz·qux) = foo·bar·baz·qux

  concat is associative for all L

- L is decidable, but L^* is undecidable.

  [ ] Yes [X] No

  text splitting DP ⟵ L^* is decidable for all decidable L

- L is neither regular nor NP-hard.

  [X] Yes [ ] No

  $\{0^n 1^n \mid n \geq 0\}$

- L is in P, and L has an infinite fooling set.

  [X] Yes [ ] No

- The language $\{\langle M \rangle \mid M \text{ accepts } L\}$ is undecidable.

  [X] Yes [ ] No

  Rice's Theorem!

(b) Suppose there is a ***polynomial-time*** reduction from some language A over the alphabet $\{0,1\}$ to some other language B over the alphabet $\{0,1\}$. Which of the following statements are ***always*** true, assuming P $\neq$ NP?

- A is a subset of B.

  [ ] Yes [X] No

  Reductions can do weird stuff. / $A = 0^*$ $B = 1^*$

- If $B \in P$, then $A \in P$.

  [X] Yes [ ] No

  $T_A(n) \leq \text{poly}(n) + T_B(n)$

- If B is NP-hard, then A is NP-hard.

  [ ] Yes [X] No

  wrong way

- If B is regular, then A is regular.

  [ ] Yes [X] No

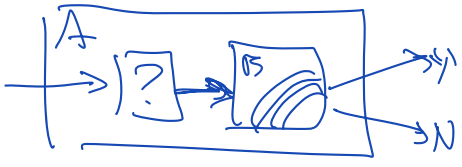
  $A = \{0^n 1^n \mid n \geq 0\}$ $B = 0^*$ reduction: FirstHalf → B

- If B is regular, then A is decidable.

  [X] Yes [ ] No

  DFA for B is algo!

  secondHalf ∈ 1^*



CS/ECE 374 A ✧ Fall 2025 Practice Final 1 Problem 2	Name:

(a) For each statement below, check "YES" if the statement is ***always*** true and "NO" otherwise, and give a ***brief*** (at most one short sentence) explanation of your answer. **Assume $P \neq NP$.** If there is any other ambiguity or uncertainty about an answer, write "NO".

- The solution to the recurrence $T(n) = 8T(n/2) + O(n^2)$ is $T(n) = O(n^2)$.

  Yes ☐ No ☒ $T(n) = O(n^3)$ / incr geom series

- The solution to the recurrence $T(n) = 2T(n/8) + O(n^2)$ is $T(n) = O(n^2)$.

  Yes ☒ No ☐ desc. geom. series

- Every directed acyclic graph contains at least one sink. = no out edges

  Yes ☒ No ☐ last node in top order / walk until stuck

- Given *any* undirected graph $G = (V, E)$, we can compute a spanning tree of G in $O(V + E)$ time using whatever-first search.

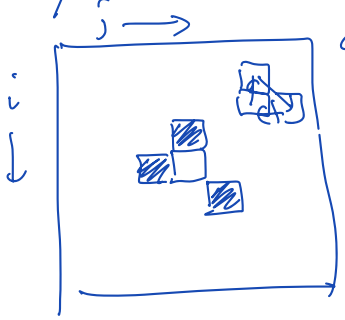
  Yes ☐ No ☒ G might be disconnected!

- Suppose $A[1..n]$ is an array of integers. Consider the following recursive function:

$$What(i,j) = \begin{cases} 0 & \text{if } i < 0 \text{ or } i > n \\ 0 & \text{if } j < 0 \text{ or } j > n \\ \max \begin{Bmatrix} What(i, j-1) \\ What(i-1, j) \\ A[i] \cdot A[j] + What(i+1, j+1) \end{Bmatrix} & \text{otherwise} \end{cases}$$

We can memoize this function into an array $What[0..n, 0..n]$ in $O(n^2)$ time, by increasing i in the outer loop and increasing j in the inner loop.

Yes ☐ No ☒ dependency cycles! recurrence on loops! dep-graph is not a dag!



Problem 2 continues on the next page.

(b) Consider the following pair of languages: hits every vertex

- DIRHAMPATH := $\{G \mid G$ is a directed graph with a Hamiltonian path$\}$
- ACYCLIC := $\{G \mid G$ is a directed acyclic graph$\}$

(For concreteness, assume that in both of these languages, graphs are represented by their adjacency matrices.) Which of the following statements are true, assuming P $\neq$ NP?

- ACYCLIC $\in$ NP

  [X] Yes [ ] No — DFS is poly-time algo and P $\subseteq$ NP

- ACYCLIC $\cap$ DIRHAMPATH $\in$ P

  [X] Yes [ ] No — ~~else~~: if dag, compute longest path $O(V+E)$ time

- DIRHAMPATH is decidable.

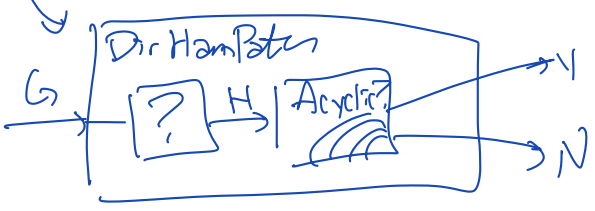
  [X] Yes [ ] No — Try all permutations of vertices / Every language in NP is decidable

- A polynomial-time reduction from DIRHAMPATH to ACYCLIC would imply P=NP.

  [X] Yes [ ] No — because DirHamPath is NP-hard and Acyclic $\in$ P

- A polynomial-time reduction from ACYCLIC to DIRHAMPATH would imply P=NP.

  [ ] Yes [X] No — reduction is in wrong direction



CS/ECE 374 A ✧ Fall 2025 Practice Final 1 Problem 3	Name:

Describe and analyze an algorithm to determine whether the language accepted by a given DFA is finite or infinite. You can assume the input alphabet of the DFA is {0,1}. *[Hint: DFAs are directed graphs.]*

---

graph

string ⟷ walk in graph from s to some other state

accepting ⟷ walk from s to acc. state

Intuition

infinite # accepted strings "⟷" cycle!

Finite "⟷" DAG

For all accepting states $t \in A$
mark all vertices reachable from s (WFS)
mark all vertices that can reach t (WFS in rev(G))

Given $M=(Q,s,A,\delta)$

Let $G'=(V',E')$
V' = double-marked states/verts
$E' = \{u \to v \in E \mid u,v \in V'\}$

if G' is not a dag (DFS)
return "Infinite!"

return "Finite!"

$O(V) \times O(V+E)$
$= O(V^2 + VE)$ time

$A \times O(Q+2Q)$
$O(AQ) = O(Q^2)$ time

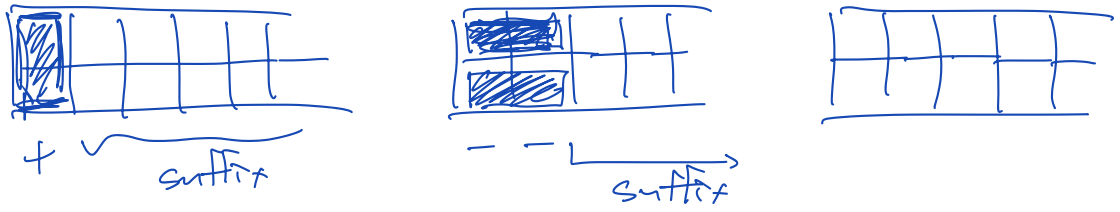
CS/ECE 374 A ✧ Fall 2025 Practice Final 1 Problem 4	Name:

Suppose you are asked to tile a $2 \times n$ grid of squares with dominos (1×2 rectangles). Each domino must cover exactly two grid squares, either horizontally or vertically, and each grid square must be covered by exactly one domino.

DP!

Each grid square is worth some number of points, which could be positive, negative, or zero. The ***value*** of a domino tiling is the sum of the points in squares covered by vertical dominos, ***minus*** the sum of the points in squares covered by horizontal dominos.

Describe an algorithm to compute the largest possible value of a domino tiling of a given $2 \times n$ grid. Your input is an array $Points[1..2, 1..n]$ of point values.



Define MaxValue(i) = largest value of a tiling of Pts[1..2, i...n]

$$MaxValue(i) = \begin{cases} 0 & \text{if } i > n \\ Pts[1,n] + Pts[2,n] & \text{if } i = n \\ \max \begin{cases} Pts[1,i] + Pts[2,i] + MaxValue(i+1) \\ -Pts[1,i] - Pts[2,i] \\ \quad -Pts[1,i+1] - Pts[2,i+1] \\ \quad + MaxValue(i+2) \end{cases} & \text{otherwise} \end{cases}$$

We want MaxValue(1)

Memoize into 1D array MaxValue[1..n]

Fill ← decreasing i

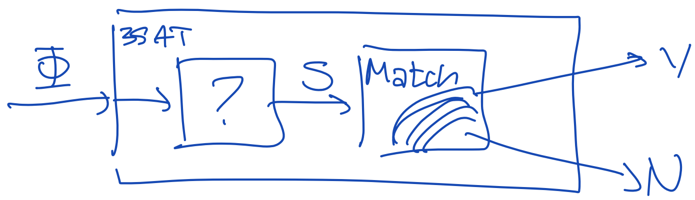
O(n) time

CS/ECE 374 A ✧ Fall 2025 **Practice Final 1 Problem 5**	Name:

3SAT? ***Prove*** that the following problem (which we call MATCH) is NP-hard. The input is a finite set S of strings, all of the same length n, over the alphabet $\{0, 1, 2\}$. The problem is to determine whether there is a string $w \in \{0, 1\}^n$ such that for every string $s \in S$, the strings s and w have the same symbol in at least one position.

↳ assignment w satisfies clause s?

Reduce from 3SAT



Given $\Phi = C_1 \wedge C_2 \wedge C_3 \wedge \cdots \wedge C_m$ variables: $x_1, x_2, \ldots, x_n$

$\downarrow$

$(x_i \vee \overline{x_j} \vee x_k)$

clause → string?

Construct $S = \{s_1, s_2, \ldots, s_m\}$ as follows:

For each $i = 1$ to m:

each string s_i has length n

Suppose clause $C_i = (x_j \vee \overline{x_k} \vee x_\ell)$ ← For example. Other clauses are similar.

$\rightarrow s_i = 222\,1\,222\,0\,222\,1\,222$ (1 at position j, 0 at position k, 1 at position ℓ)

1 for each raw variable

0 for each negated variable

2 = "don't care"

we get m strings of length n → S

Polynomial time by brute force ✓

$O(nm)$ time

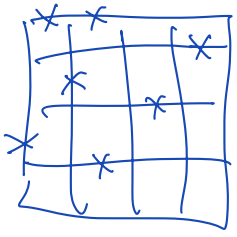
See page 8 for proof!

CS/ECE 374 A ✧ Fall 2025 Practice Final 1 Problem 6	Name:

dag?

Suppose you are given a height map of a mountain, in the form of an $n \times n$ grid of evenly spaced points, each labeled with an elevation value. You can safely hike directly from any point to any neighbor immediately north, south, east, or west, but only if the elevations of those two points differ by at most Δ. (The value of Δ depends on your hiking experience and your physical condition.)

Describe and analyze an algorithm to determine the longest hike from some point s to some other point t, where the hike consists of an uphill climb (where elevations must increase at each step) followed by a downhill climb (where elevations must decrease at each step). Your input consists of an array $Elevation[1..n, 1..n]$ of elevation values, the starting point s, the target point t, and the parameter Δ.



Build $G = (V, E)$

V = grid pts

E = adj grid pts Elev[ ] differing by $\leq \Delta$

↓

Build dir. graph $G' = (V', E')$

$V' = V \times \{up, down\}$

$E' = \{(u, up) \to (v, up) \mid uv \in E, Elev[u] < Elev[v]\}$

$\cup \{(u, dn) \to (v, dn) \mid uv \in E, Elev[u] > Elev[v]\}$

$\cup \{(v, up) \to (v, dn) \mid v \in V\}$

Elev only increases

Elev only decreases

G' is a dag

we want longest path in G' from (s, up) to (t, dn)

use textbook LongestPath algo in $O(V+E)$ time

$= O(n^2)$ time

CS/ECE 374 A ✧ Fall 2025 Practice Final 1 Problem 7	Name:

(a) Let L_a denote the set of all strings in $\{0,1\}^*$ where every 0 is followed immediately by at least one 1. Describe a DFA or NFA that accepts L_a ***and*** give a regular expression that describes L_a. (You do not need to prove that your answers are correct.)

Fooling (b) Let L_b denote the set of all strings in $\{0,1\}^*$ whose run lengths are increasing; that is, every run except the last is followed immediately by a *longer* run. ***Prove*** that L_b is not a regular language.

(a)

$(1+01)^*$

(b) Let $F = 0^*$

Let $x \neq y$ be arbitrary strings in F

$\Rightarrow x = 0^i$ and $y = 0^j$ for some $i \neq j$

WLOG $i < j$ (else swap $x \leftrightarrow y$ $i \leftrightarrow j$)

Let $z = 1^j$

Then $xz = 0^i 1^j \in L_b$ because $i < j$

$yz = 0^j 1^j \notin L_b$ because $j \not< j$

So F is an infinite fooling set for L_b □

(overflow / scratch paper)

Problem 5 continued

Proof:

($\Rightarrow$) Suppose Φ has a sat. assignment

values for $x_1 \ldots x_n$

↓

Build $w[1..n]$. $w[i] = \begin{cases} 1 & \text{if } x_i = T \\ 0 & \text{if } x_i = F \end{cases}$

Every clause C_i has ≥ 1 true literal

If $C_i = (x_j \vee \overline{x_k} \vee x_\ell)$

then $x_j = T$ or $x_k = F$ or $x_\ell = T$

$\Rightarrow w[j] = 1$ or $w[k] = 0$ or $w[\ell] = 1$

Def of $s_i \Rightarrow s_i[j] = 1$ and $s_i[k] = 0$ and $s_i[\ell] = 1$

so $w[r] = s_i[r]$ for some r

So MATCH(S) = True

($\Leftarrow$) Spose MATCH(S) = True

$\Rightarrow$ There is a string $w[1..n]$

s.t. for all i $s_i[r] = w[r]$ for some r.

Define assignment $x_i = \begin{cases} T & \text{if } w[i] = 1 \\ F & \text{if } w[i] = 0 \end{cases}$

Then for each i:

Suppose $C_i = (x_j \vee \overline{x_k} \vee x_\ell)$

Then $s_i[j] = 1$ and $s_i[k] = 0$ and $s_i[\ell] = 1$

$\Rightarrow$ Either $w[j] = 1$ or $w[k] = 0$ or $w[\ell] = 1$

$\Rightarrow x_j = T$ or $x_k = F$ or $x_\ell = T$

$\Rightarrow C_i$ is satisfied

$\Rightarrow$

Φ is satisfiable!

(overflow / scratch paper)

(overflow / scratch paper)

(overflow / scratch paper)

(overflow / scratch paper)

Some useful NP-hard problems. You are welcome to use any of these in your own NP-hardness proofs, except of course for the specific problem you are trying to prove NP-hard.

CircuitSat: Given a boolean circuit, are there any input values that make the circuit output True?

3Sat: Given a boolean formula in conjunctive normal form, with exactly three distinct literals per clause, does the formula have a satisfying assignment?

MaxIndependentSet: Given an undirected graph G, what is the size of the largest subset of vertices in G that have no edges among them?

MaxClique: Given an undirected graph G, what is the size of the largest complete subgraph of G?

MinVertexCover: Given an undirected graph G, what is the size of the smallest subset of vertices that touch every edge in G?

MinSetCover: Given a collection of subsets $S_1, S_2, \ldots, S_m$ of a set S, what is the size of the smallest subcollection whose union is S?

MinHittingSet: Given a collection of subsets $S_1, S_2, \ldots, S_m$ of a set S, what is the size of the smallest subset of S that intersects every subset S_i?

3Color: Given an undirected graph G, can its vertices be colored with three colors, so that every edge touches vertices with two different colors?

ChromaticNumber: Given an undirected graph G, what is the minimum number of colors required to color its vertices, so that every edge touches vertices with two different colors?

HamiltonianPath: Given graph G (either directed or undirected), is there a path in G that visits every vertex exactly once?

HamiltonianCycle: Given a graph G (either directed or undirected), is there a cycle in G that visits every vertex exactly once?

TravelingSalesman: Given a graph G (either directed or undirected) with weighted edges, what is the minimum total weight of any Hamiltonian path/cycle in G?

LongestPath: Given a graph G (either directed or undirected, possibly with weighted edges), what is the length of the longest simple path in G?

SteinerTree: Given an undirected graph G with some of the vertices marked, what is the minimum number of edges in a subtree of G that contains every marked vertex?

SubsetSum: Given a set X of positive integers and an integer k, does X have a subset whose elements sum to k?

Partition: Given a set X of positive integers, can X be partitioned into two subsets with the same sum?

3Partition: Given a set X of $3n$ positive integers, can X be partitioned into n three-element subsets, all with the same sum?

IntegerLinearProgramming: Given a matrix $A \in \mathbb{Z}^{n \times d}$ and two vectors $b \in \mathbb{Z}^n$ and $c \in Z^d$, compute $\max\{c \cdot x \mid Ax \leq b, x \geq 0, x \in \mathbb{Z}^d\}$.

FeasibleILP: Given a matrix $A \in \mathbb{Z}^{n \times d}$ and a vector $b \in \mathbb{Z}^n$, determine whether the set of feasible integer points $\max\{x \in \mathbb{Z}^d \mid Ax \leq b, x \geq 0\}$ is empty.

Draughts: Given an $n \times n$ international draughts configuration, what is the largest number of pieces that can (and therefore must) be captured in a single move?

SuperMarioBrothers: Given an $n \times n$ Super Mario Brothers level, can Mario reach the castle?