A. Generalidades sobre la teoría

1. Mostrar que los siguientes problemas pertenecen a P:

   a) Determinar si una lista de n números se encuentra ordenada.
   b) Determinar si un grafo es conexo.
   c) Determinar si un grafo tiene un ciclo.

2. Considerar el problema de determinar si un grafo contiene a un K_4 como subgrafo inducido. ¿Pertenece a P este problema? ¿Pertenece a NP?

3. Está usted tratando de resolver un determinado problema cuando le comunican que el mismo está en NP. ¿Es un motivo de alegría o de tristeza?

4. Demuestre que si un problema Π pertenece a NP, entonces puede ser resuelto en tiempo $O(2^{p(n)})$ mediante un algorítmico determinístico, con $p(n)$ un polinomio.

5. Demuestre que $P = coP$. Es decir, si una máquina de Turing determinística puede aceptar una entrada positiva en tiempo acotado polinomialmente entonces es posible construir una máquina de Turing determinística para poder aceptar entradas negativas en tiempo polinomial.

6. Diseñar algoritmos no determinísticos polinomiales para los siguientes problemas. No es neceasrio construir una máquina de Turing no determinística, puede escribir en un pseudo-código ampliado con la función *Guess()*. Determine la complejidad.

   a) Determinar si un grafo es conexo.
   b) Determinar si dos grafos son isomorfos.
   c) Determinar si un grafo contiene un circuito hamiltoniano.

7. Sean X e Y dos problemas de decisión tales que $X \leq_p Y$. ¿Qué se puede inferir?

   a) Si X está en P entonces Y también.
   b) Si Y está en P entonces X también.
   c) Si X es NP-completo entonces Y también.
   d) Si X es NP-hard entonces Y también.
   e) No pueden ser ambos NP-completos.

8. ¿Qué se puede decir del hecho de que el problema de viajante de comercio (TSP) es NP-completo, suponiendo $P \neq NP$?

   a) No existe un algoritmo que resuelva instancias arbitrarias de TSP.
   b) No existe un algoritmo que eficientemente resuelva instancias arbitrarias de TSP.
   c) Existe un algoritmo que resuelve eficientemente instancias arbitrarias de TSP pero nadie lo ha encontrado.
   d) TSP no está en P.
   e) Todos los algoritmo que resuelven TSP corren en tiempo exponencial para todas las entradas.

B. Reducciones polinomiales y problemas NP-completos

Para esta práctica, consideraremos los siguients problemas (además de los ya vistos en clase).

MATCHING

INPUT: dos conjuntos A ("los chicos") y B ("las chicas"). Un conjunto C de pares de $A \times B$ ("las parejas posibles").
OUTPUT: ¿existe un subconjunto de C de manera tal que cada elemento de A y de B aparezca una y sólo una vez?

MATCHING 3D

INPUT: tres conjuntos A ("los chicos"), B ("las chicas") y C ("las mascotas"). Un conjunto D de ternas de $A \times B \times C$.
OUTPUT: ¿existe un subconjunto de D de manera tal que cada elemento de A, de B y de C aparezca una y sólo una vez?

ZOE

INPUT: una matriz A de $m \times n$, con entradas binarias.
OUTPUT: ¿existe un vector binario $x = (x_1, \ldots, x_m)$ tal que $Ax = 1$ (vector m-dimensional de unos)?

SUBSET-SUM

INPUT: un conjunto S de números enteros y k un entero.
OUTPUT: ¿existe un subconjunto de S que sume exactamente k?

CAMINO MÁS LARGO

INPUT: un grafo G, un entero k.
OUTPUT: ¿existe un camino simple en G de longitud mayor o igual a k?

TAXICAB RIP-OFF ("la estafa del taxi")

INPUT: un grafo G con costos enteros en las aristas, un entero k.
OUTPUT: ¿existe un camino simple en G de costo mayor o igual a k?

1. a) Sea $G = (V, X)$ un grafo y $W \subset V$. Demostrar la equivalencia de las siguientes afirmaciones:
      1) $V - W$ es un recubrimiento por arcos de G (es decir, cualquier eje es incidente a algún nodo de $V - W$).
      2) W es un conjunto independiente de G (es decir, ningún par de vértices de W son adyacentes entre sí).
      3) W como subgrafo inducido es un clique de G^c.

   b) A partir de lo anterior, encuentre reducciones polinomiales entre los siguientes problemas: VERTEX-COVER (¿existe un cubrimiento de las aristas con a lo sumo k vértices?), Conjunto Independiente (¿existe un conjunto independiente con al menos k vértices?) y Clique (¿existe un subgrafo completo con al menos k vértices?).

   c) ¿A qué clase de complejidad perteneces estos tres problemas?

2. Sabiendo que el problema CIRCUITO HAMILTONIANO es NP-completo, demuestre que el problema TSP también es NP-completo.

3. Demuestre que el problema CIRCUITIO HAMILTONIANDO DIRIGIDO es NP-completo.

4. El problema VAGABUNDO es similar al TSP, pero el viajante no tiene que retornar a la ciudad de origen. Demuestre que este problema es NP-completo.

5. Se sabe que el problema SUBSET-SUM es NP-completo. Utilice este problema para probar que el problema 0-1 KNAPSACK es NP-completo.

6. Usando ZOE, pruebe que SUBSET-SUM es NP-completo.

7. Usando MATCHING 3D, demuestre que ZOE es NP-completo.

8. Demuestre que tanto CAMINO MÁS LARGO como TAXICAB RIP-OFF son NP-completos.

9. Clasifique los siguientes problemas en P("fácil") o NP-completo ("difícil"): SAT, 3-SAT, 2-SAT, TSP, ÁRBOL GENERADOR MÍNIMO, CAMINO MÁS LARGO, CAMINO MÁS CORTO, MATCHING, MATCHING 3D, 0-1 KNAPSACK, KNAPSACK SIMPLIFICADO, CONJUNTO INDEPENDIENTE, PROGRAMACIÓN LINEAL ENTERA, PROGRAMACIÓN LINEAL, CAMINO EULERIANO, CAMINO HAMILTONIANO, CORTE MÍNIMO, CORTE MÁXIMO, TAXICAB RIP-OFF, VERTEX-COVER, CLIQUE MÁXIMO, EDGE-COVER, COLOREO MÍNIMO.

10. Problema CONJUNTO DOMINANTE.
    INPUT: un grafo $G = (V, E)$ y un entero k.
    OUTPUT: ¿existe un subconjunto de vértices V' de cardinal menor o igual que k tal que todo nodo del grafo es adyacente a algún elemento de V'?

    Probar que CONJUNTO DOMINANTE es NP-completo. Sugerencia, usar que VERTEX-COVER es NP-completo.