uniandes

# Ingeniería de Sistemas y Computación

## Diseño y análisis de algoritmos

Profesor: Mateo Sanabria Ardila

Examen 3 Supletorio

Fecha de entrega: Diciembre 08 | 2024-20 | Nota máxima: 120

1. **(+50 pts)** En una fábrica, un robot debe inspeccionar un conjunto de placas de circuito ubicadas en distintos puntos de una cuadrícula. El robot comienza y termina en una estación de carga y debe visitar cada placa de circuito exactamente una vez. Se proporciona la distancia entre cada par de puntos, y el robot puede desplazarse directamente entre ellos. La configuración de las placas de circuito y la ubicación de la estación de carga cambian dinamicamente debido a la naturaleza de la fábrica. Usted es el ingeniero encargado de programar el robot para que calcule la secuencia de visitas que minimice la distancia total recorrida.

   Mientras explorae ste problema, se da cuenta de que es computacionalmente complicado , y tu jefe le solicita justificar por qué ocurre esto. Para esto debe:

   1. **(+5 pts)** Mostrar el lenguaje aceptado del problema de decisión asociado.
   2. **(+15 pts)** Mostrar que el problema pertenece a NP.
   3. **(+30 pts)** Mostrar que el problema es NPH.

   Para cualquiera de los puntos anteriores que requiera mostrar un algoritmo (o la implementación de una función) debe hacerlo usando pseudocódigo. No es valido una explicación en palabras. Si cree que el problema se relaciona a un problema NPC conocido, no es valido que justifique los puntos mencionando dicho problema, en otras palabras si cree que es un problema conocido debe mostrar NP y NPH para ese problema. Sin embargo, puede usar otros problemas NPC conocidos para demostrar que el problema es NPH.

2. **(+20 pts)** Responda las siguientes preguntas, cada pregunta depende de la anterior. Por lo tanto, si la respuesta a una pregunta es incorrecta, las siguientes serán incorrectas:

   - (**+5 pts**) Indique el lenguaje aceptado para el problema.
   - (**+5 pts**) Describa el problema de optimización asociado.
   - (**+10 pts**) Suponga que existe un algoritmo que resuelve el problema el problema de decisión en tiempo constante, muestre un algoritmo (pseudocódigo) que resuelva el problema de optimización en tiempo polinomial.

3. **(+30 pts)** El Teorema 35.3 ([Cor2009] 35) menciona que si $P \neq NP$ entonces para cualquier constante $1 \leq p$, no existe un algoritmo aproximado en tiempo polinomial con ratio de aproximación $p$ para el problema generar de TSP.

   - **(+15 pts)** Muestre como en tiempo polinomial se puede traducir una instancia del problema de TSP en una instancia donde la función de costo satisface la desigualdad triangular. Las dos instancias deben tener el mismo conjunto de tours óptimos.
   - **(+15 pts)** Asuma que $P \neq NP$ explique como el punto anterior no contradice el teorema mencionado.

4. **(+20 pts BONO)** Dado un total especificado $t$ y un multiconjunto $S$ de $n$ números enteros, encuentra todos los subconjuntos distintos de $S$ cuyos elementos sumen $t$. Por ejemplo, si $t = 4$ y $S = \{4, 3, 2, 2, 1, 1\}$, entonces existen cuatro sumas distintas que son iguales a $t$: 4, $3 + 1$, $2 + 2$, y $2 + 1 + 1$. Un número puede ser usado en una suma hasta la cantidad de veces que aparece en $S$, y un único número cuenta como una suma.

   Implemente (**Python, Java...**) la función *printSums* utilizando la técnica de **backtraking** para imprimir todas las posible sumas que se pueden obtener dados $t$ y $S$.

   **Noy hay puntos intermedios, para tener los puntos completos debe proporcionar código que ejecute correctamente la tarea, salvo errores de sintaxis.**

```
# Muestra por consola todas las posibles sumas distintas de los elementos
# de S que suman t, en cualquier formato y orden.
    printSums(t: int,s: Set) -> None
```