TRABAJO PRÁCTICO 3

PROBLEMA 1 - Algoritmos de Aproximación

Tenemos un conjunto de enteros positivos $A=\{a_1, a_2, ..., a_n\}$ y otro entero positivo B. Llamaremos *subconjunto factible* a un subconjunto S de A tal que la suma de sus elementos sea a lo sumo el valor B:

$$\sum_{a_i \in S} a_i \leq B$$

Queremos encontrar un subconjunto factible S de A tal que la suma de sus elementos sea la mayor posible. Por ejemplo, si $A=\{9,3,5\}$ y $B=13$, entonces la solución óptima es $S=\{9,3\}$.

El siguiente es un algoritmo para encontrar un subconjunto factible (no necesariamente el óptimo):

```
subconjunto_factible(A, B)
  S={}
  T=0
  Para i=1,2,...,n
    Si T + A[i] <= B, entonces:
      S = S ∪ A[i]
      T = T + A[i]
    fin si
  fin para
  devolver S
```

Se pide:

1. Presentar una instancia de A y B tal que la suma de los elementos del conjunto factible S devuelto por este algoritmo sea menor que la mitad de la suma total de algún otro conjunto factible de A.
2. Diseñar y desarrollar un algoritmo de aproximación que encuentre un subconjunto factible con la garantía de que su suma total sea al menos la mitad de la suma total de cualquier subconjunto factible de A.
3. Supuestos: identificar supuestos, condiciones, limitaciones y/o premisas bajo los cuales funcionará el algoritmo desarrollado
4. Diseño:
    a. Mostrar cómo se cumple la garantía solicitada
    b. Incluir un Pseudocódigo
    c. Detallar las estructuras de datos utilizadas
5. Seguimiento: Ejemplo de seguimiento con un set de datos reducido
6. Complejidad: Análisis de la complejidad temporal a partir del pseudocódigo
7. Sets de datos: diseñar sets de datos apropiados. Se pueden generar utilizando una función random con una semilla fija, para permitir la reproducibilidad de los resultados, o ser generados manualmente e incluidos como archivos que lee el programa.
    a. Cada set de datos debe ser incluido en la entrega, junto con el resultado obtenido en cada caso.

   b. El programa entregado debe generar los sets de datos o leerlos desde archivos.
8. Tiempos de Ejecución: medir los tiempos de ejecución de cada set de datos y presentarlos en un gráfico. Verificar que la curva de tiempos medida se corresponde con la complejidad temporal establecida en el punto 6.
9. Informe de Resultados: redactar un informe analizando los resultados obtenidos. ¿Se cumple con la garantía solicitada?

PROBLEMA 2 - Algoritmos Randomizados

Tenemos un sitio web de subastas. Supongamos que un vendedor pone a la venta un producto, y que existen n compradores, cada uno con una oferta diferente. Supongamos también que los compradores se presentan en un orden aleatorio, y que el vendedor conoce el número n de compradores. A medida que cada comprador se presenta, el vendedor analiza su oferta y decide si la toma o la descarta: si la toma, el producto se vende y termina la subasta (el vendedor no analizará otras ofertas); si la descarta, el vendedor analizará otra oferta y nunca más podrá volver a tomar la oferta que descartó.

Queremos diseñar una *estrategia* donde el vendedor tenga una probabilidad razonable de aceptar la mayor de las n ofertas. Por *"estrategia"* nos referimos a una regla según la cual el vendedor decide aceptar una oferta basado únicamente en el valor de n y las ofertas presentadas hasta ese momento. Por ejemplo, el vendedor podría aceptar siempre la primera oferta presentada: esto implicaría que la probabilidad de aceptar la mayor oferta es de $1/n$, ya que requiere que la mayor de todas las ofertas sea la primera.

Se pide:

1. Diseñar y desarrollar una estrategia donde el vendedor acepta la mayor de las n ofertas con una probabilidad de al menos 1/4, independientemente del valor de n (se puede suponer que n es un número impar).
2. Supuestos: identificar supuestos, condiciones, limitaciones y/o premisas bajo los cuales funcionará el algoritmo desarrollado
3. Diseño:
    a. Mostrar cómo se cumple la garantía solicitada
    b. Incluir un Pseudocódigo
    c. Detallar las estructuras de datos utilizadas
4. Seguimiento: Ejemplo de seguimiento con un set de datos reducido
5. Complejidad: Análisis de la complejidad temporal a partir del pseudocódigo
6. Sets de datos: diseñar sets de datos apropiados. Se pueden generar utilizando una función random con una semilla fija, para permitir la reproducibilidad de los resultados, o ser generados manualmente e incluidos como archivos que lee el programa.
    a. Cada set de datos debe ser incluido en la entrega, junto con el resultado obtenido en cada caso.
    b. El programa entregado debe generar los sets de datos o leerlos desde archivos.
7. Tiempos de Ejecución: medir los tiempos de ejecución de cada set de datos y presentarlos en un gráfico. Verificar que la curva de tiempos medida se corresponde con la complejidad temporal establecida en el punto 5.
8. Informe de Resultados: redactar un informe analizando los resultados obtenidos. ¿Se cumple con la garantía solicitada?

PROBLEMA 3 - Autómatas

Tenemos dos lenguajes que corresponden a la intersección (representada mediante la conjunción "y") de dos lenguajes más simples. Los símbolos de los mismos corresponden a los números 0 y 1:

Lenguaje A: *{w/ w (comienza con "1") y (tiene cuando mucho un "0")}*
Lenguaje B: *{w/ w (tiene un número par de "1") y (tiene uno o dos "0")}*

Por ejemplo:
Lenguaje A: "10111", "11", "110"
Lenguaje B: "1010", "11110"

Se pide:

1. Describir formalmente un autómata finito determinístico para cada uno de los lenguajes simples:
    a. Indicar el conjunto de estados
    b. Presentar la función de transición en forma de tabla
    c. Indicar cuál es el estado inicial
    d. Indicar cuáles son los estados de aceptación
2. Basándose en los autómatas descritos, describir formalmente los dos autómatas finitos determinísticos de los lenguajes A y B.
3. Describir cada uno de los seis autómatas mediante su representación gráfica.
4. Para cada uno de los seis autómatas, brindar un ejemplo de una cadena que acepta y de otra que rechaza. Explicar paso a paso cómo decide si acepta o rechaza.

Condiciones Generales de Entrega

- El trabajo debe ser entregado en un archivo zip conteniendo:
    - Documento con carátula, índice y numeración de páginas. La carátula debe incluir nombre y padrón de los integrantes del grupo. Debe presentarse en formato PDF.
    - Archivos con los fuentes de los algoritmos desarrollados para los problemas 1 y 2
    - Archivo README indicando el lenguaje de programación utilizado, versión mínima y bibliotecas requeridas, e instrucciones para ejecutar.
    - Archivos con los sets de datos utilizados
    - Archivos con resultados obtenidos para cada set de datos
- IMPORTANTE: el tamaño de los sets de datos (cuando corresponda) debe ser tal que permita obtener suficiente información como para graficar los tiempos de ejecución y verificar la complejidad temporal.
- Se calificará positivamente la inclusión de referencias bibliográficas relevantes. Para ello, se pueden incluir citas en el texto del informe siguiendo el modelo propuesto por Rivas (2022) y luego incorporar el listado completo en un anexo al final, usando normas APA 7ma edición.

Referencias:

- Rivas, A. (2022, mayo 25). Cómo hacer una lista de referencias con Normas APA. Guía Normas APA. https://normasapa.in/como-hacer-la-lista-de-referencias/