Análisis y diseño de algoritmos
Tarea 4
30 de octubre de 2024
Dr. Eduardo A. Rodríguez Tello

Instrucciones: Responda las siguientes preguntas de manera breve y clara. Genere un reporte en el editor de texto de su preferencia en formato PDF donde presente los detalles del trabajo realizado, así como sus *conclusiones personales*. No olvide incluir las referencias bibliográficas empleadas y los análisis de complejidad desarrollados. Entregue además un archivo ZIP (no RAR) que incluya el código fuente de los programas desarrollados (documentado), los archivos necesarios para ejecutar sus programas (instancias de prueba), y un archivo README.txt con instrucciones para compilar y ejecutar sus programas.

Fecha de entrega: martes 11 de noviembre antes de las 23h59.

1. Dado el problema de intersección de segmentos de línea, desarrolle los siguientes puntos:

   a) Programe en C++ un algoritmo de fuerza bruta para resolver el problema. El programa debe recibir por redirección de la línea de comandos los casos de prueba (archivos texto) con el siguiente formato: primera línea un entero n que indica el número total de segmentos de línea del problema, después hay n líneas cada una con cuatro enteros separados por espacios que representan las coordenadas x y y de los dos puntos extremos de cada línea del problema. El programa regresa un archivo con el siguiente formato: primera línea con un entero k que representa cuántas intersecciones existen entre las líneas del problema, después hay k lineas con dos enteros separados por espacio que indican las coordenadas de los puntos de intersección respectivos.

   b) Analice matemáticamente el algoritmo usando las metodologías vistas en clase (i.e., usando sumatorias o relaciones de recurrencia).

   c) Efectúe al menos 10 pruebas con el algoritmo desarrollado generando para cada prueba instancias de tamaños 25, 50, 100, 200, 400, etc. Ejecute 10 corridas de su algoritmo con cada tamaño de instancia generado y registre sus resultados promedio.

   d) Presente capturas de pantalla para una corrida con cada uno de los tamaños de instancia siguientes: 25, 100, 400 y 1600.

   e) Genere una tabla con información estadística de su programa: tamaño de la instancia, tiempo promedio para efectuar 10 ejecuciones, desviación estándar del tiempo de ejecución, y número de operaciones básicas promedio. Estas tablas deben permitir analizar cómo afecta el crecimiento del tamaño de los casos de prueba al desempeño del algoritmo implementado.

   f) A partir de la tabla anterior grafique para el algoritmo analizado los siguientes resultados de sus ejecuciones: tamaño de entrada (eje X) contra el tiempo de cómputo consumido (eje Y). Genere la línea de tendencia para los datos, utilizando la opción (exponencial, lineal, logarítmica, etc) que permita obtener el mejor ajuste. Reporte la ecuación de la línea de tendencia, así como el valor de R^2. (Un valor de R^2 más cercano a 1.0 es indicativo de un mejor ajuste).

   g) Responder la siguiente pregunta: Con base en los experimentos realizados y considerando un tiempo máximo de ejecución sobre su computadora de 7 días, ¿Cuál es el tamaño máximo de entrada que puede resolver el algoritmo analizado (con una sola ejecución independiente)? Justifique su respuesta.

h) Analice y discuta sus observaciones de la información contenida en la tabla y gráfica. Trate de correlacionar sus observaciones con los resultados del análisis matemático de la complejidad del algoritmo que implementó.

2. La empresa *Transportes Eficientes* dedicada a la logística tiene almacenes situados en diversos estados del país. Dado que se conocen las distancias de las carreteras comunican ciertos pares de almacenes, entonces es posible modelar esta situación como un grafo ponderado no dirigido. El gerente de la empresa desea conocer la ruta más larga entre cualquier par de almacenes ya que esto le permitirá tomar decisiones estratégicas para hacer más eficiente la operación de su empresa. Dado este problema práctico, desarrolle los siguientes puntos:

   a) Diseñe e implemente en C++ un algoritmo **recursivo** basado en DFS para encontrar la ruta más larga entre cualquier par de almacenes de la empresa. El programa recibe por redirección de la línea de comandos el nombre de un archivo de texto que contiene un grafo ponderado dirigido con el formato que se utilizó en clase (sesión 9). El resultado que se imprimirá en pantalla es la distancia total de la ruta más larga entre cualquier par de almacenes de la empresa, la lista de almacenes que forman dicha ruta y el tiempo total de ejecución. Asegurese de utilizar una lista de adyacencia para representar el grafo.

   b) Analice matemáticamente la complejidad temporal y espacial de su algoritmo.

   c) Presente capturas de pantalla con ejemplos de sus corridas para los tamaños de instancia $|V| = n = \{25, 50, 100\}$ generados aleatoriamente. Para cada tamaño presente una corrida.

   d) Sin realizar modificaciones a su programa, analice cómo se vería afectada la complejidad temporal y espacial de su algoritmo en el caso de que se utilizara una matriz de adyacencia para representar el grafo. Presente su análisis y justifique sus afirmaciones.

3. Un grafo es *bipartita* si todos sus vértices pueden ser divididos en dos subconjuntos disjuntos X y Y de forma tal que cada arco conecte un vértice en X con uno en Y. Dado el problema de identificación de grafos bipartitas, desarrolle los siguientes puntos:

   a) Diseñe e implemente en C++ un algoritmo basado en BFS para verificar si un grafo es *bipartita* o no. Este recibe por redirección de la línea de comandos el nombre de un archivo de texto que contiene el grafo a verificar con el formato que se utilizó en clase (sesión 9). El resultado que despliega en pantalla es "Bipartita" o "No bipartita" y el tiempo total de ejecución. Asegurese de utilizar una lista de adyacencia para representar el grafo.

   b) Analice matemáticamente la complejidad temporal y espacial de su algoritmo.

   c) Presente capturas de pantalla con ejemplos de sus corridas para los tamaños de instancia $n = \{50, 100, 150\}$. Para cada tamaño presente dos corridas una con un grafo bipartita y otra con un grafo no bipartita. El conjunto de sus instancias de prueba debe contener grafos dirigidos y no dirigidos.

   d) Sin realizar modificaciones a su programa, analice cómo se vería afectada la complejidad temporal y espacial de su algoritmo en el caso de que se utilizara una matriz de adyacencia para representar el grafo. Presente su análisis y justifique sus afirmaciones.