TAREA 2_3
ALGORITMOS Y COMPLEJIDAD

«Explorando la Distancia entre Cadenas, una Operación a la Vez»

Fecha límite de entrega 10 de noviembre de 2024

Equipo ALGORITMOS Y COMPLEJIDAD 2024-2

28 de octubre de 2024
08:41
Versión 1.0

Índice

1. Objetivos

El objetivo de esta tarea 2_3 es introducirnos en el Diseño de Algoritmos. Para esto, utilizaremos dos paradigmas de programación: fuerza bruta y programación dinámica. El primer paradigma es un enfoque fundamental en el diseño de algoritmos y es especialmente útil para resolver problemas donde todas las posibles soluciones deben ser evaluadas exhaustivamente y lo compararemos con otro enfoque. El segundo paradigma es programación dinámica.

En este contexto, utedes deben implementar 2 algoritmos que calculen la **distancia mínima de edición** entre dos cadenas utilizando los 2 enfoques, incorporando operaciones como inserciones, eliminaciones, sustituciones y transposiciones, con costos variables.

La **distancia de edición** entre dos cadenas es el número mínimo de operaciones necesarias para transformar una cadena en la otra.

2. Contexto: Distancia de edición extendida con costos variables y transposiciones

En el problema estándar de distancia de edición, las operaciones permitidas son inserciones, eliminaciones y sustituciones, cada una típicamente asignada con un costo uniforme. En esta tarea, ampliarás el problema con las siguientes mejoras:

1) Costos Variables de Operación:
Cada operación de edición (inserción, eliminación, sustitución) puede tener diferentes costos, que pueden variar dependiendo de los caracteres específicos involucrados.

2) Operación de Transposición:
Una operación adicional donde dos caracteres adyacentes se intercambian, también con un costo asociado.

Detalles de las Funciones de Costos:

Costo de Sustitución: Esta función calcula el costo de sustituir el carácter 'a' por el carácter 'b'.

```
// Calcula el costo de sustituir el carácter 'a' por 'b'.
// Parámetros:
// - a: carácter original
// - b: carácter con el que se sustituye
// Return: costo de sustituir 'a' por 'b'
int costo_sub(char a, char b) {
    int costo;
    // Implementación
    return costo;
}
```

Costo de Inserción: Esta función devuelve el costo de insertar el carácter 'b' en la cadena.

```
// Calcula el costo de insertar el carácter 'b'.
// Parámetros:
// - b: carácter a insertar
// Return: costo de insertar 'b'
int costo_ins(char b) {
    int costo;
    // Implementación
    return costo;
}
```

Costo de Eliminación: Esta función devuelve el costo de eliminar el carácter 'a' de la cadena.

```
// Calcula el costo de eliminar el carácter 'a'.
// Parámetros:
// - a: carácter a eliminar
// Return: costo de eliminar 'a'
int costo_del(char a) {
    int costo;
    // Implementación
    return costo;
}
```

Costo de Transposición: Esta función devuelve el costo de intercambiar los caracteres 'a' y 'b' si son adyacentes.

```
// Calcula el costo de transponer los caracteres 'a' y 'b'.
// Parámetros:
// - a: primer carácter a transponer
// - b: segundo carácter a transponer
// Return: costo de transponer 'a' y 'b'
int costo_trans(char a, char b) {
    int costo;
    // Implementación
    return costo;
}
```

3. Especificaciones

En esta sección se presentará la estructura que tiene que tener su informe, en particular, se detallarán las secciones que debe contener y los elementos que deben ser incluidos en cada una de ellas. En el siguiente repositorio podrá encontrar el Template que deben utilizar en esta entrega:

`https://github.com/pabloealvarez/AlgoReportTemplate`

- No se debe modificar la estructura del informe.
- Las indicaciones se encuentran en el archivo `README.md` del repositorio y en la plantilla de LaTeX.
- Se actualizó el repositorio con la plantilla de LaTeXel día 27 de octubre de 2024 (se agregó aclaración en readme y sección de implementación, ver commit).

3.1. Introducción (máximo 2 páginas)

La introducción de este tipo de informes o reportes, tiene como objetivo principal **contextualizar el problema que se va a analizar**, proporcionando al lector la información necesaria para entender la relevancia del mismo.
Es fundamental que en esta sección se presenten los antecedentes del problema, destacando investigaciones previas o principios teóricos que sirvan como base para los análisis posteriores. Además, deben explicarse los objetivos del informe, que pueden incluir la evaluación de un algoritmo, la comparación de métodos o la validación de resultados experimentales.

3.2. Diseño y Análisis de Algoritmos (máximo 5 páginas)

Diseñar un algoritmo por cada técnica de diseño de algoritmos mencionada en la sección de objetivos. Cada algoritmo debe resolver el problema de distancia mínima de edición extendida, dadas dos cadenas `S1` y `S2`, utilizando las operaciones y costos especificados.

- Describir la solución diseñada.
- Incluir pseudocódigo (ver ejemplo Template)
- Proporciones un ejemplo paso a paso de la ejecución de sus algoritmos que ilustren cómo sus algoritmos manejan diferentes escenarios, particularmente donde las transposiciones o los costos variables afectan el resultado. Haga referencias a los programas expresados en pseudocódigo (además puede hacer diagramas).
- Analizar la Complejidad temporal y espacial de los algoritmos diseñados en términos de las longitudes de las cadenas de entrada $S1$ y $S2$.
- Discute cómo la inclusión de transposiciones y costos variables impacta la complejidad.

Recuerde que lo importante es diseñar algoritmos que cumplan con los paradigmas especificados.

3.2.1. Fuerza Bruta

Diseñar algorimto basado en el paradigma de Fuerza Bruta.

3.2.2. Programación Dinámica

1) Describa la solución recursiva.
2) Escriba la relación de recurrencia, incluyendo condiciones y casos base.
3) Identifique subproblemas.
4) Defina estructura de datos a utilizar y especifique el orden de calculo que realiza su programa que utiliza programación dinámica.

Todo lo anterior es, digamos, en "lapiz y papel", en el sentido de que no necesita de implementaciones ni resultados experimentales.

3.3. Implementar el Algoritmo (máximo 1 página)

Cada algoritmo debe ser implementado en C++, siguiendo los siguientes puntos:

Entrada:

- Leer dos cadenas de entrada (S1 y S2) desde la entrada estándar o desde un archivo.
- Leer los costos para cada operación (que pueden ser constantes o definidos mediante una tabla de costos).
  - Las implementaciones de cada algoritmo no incluirán las tablas de costos como entrada.
  - Las implementaciones de cada algoritmo sólo deben llamar a las funciones de costos y estas serán las encargadas de obtener los costos de cada operación desde las tablas.
  - Las tablas de costos deben ser definidas de manera global, por lo tanto no deben ser pasadas como argumento a las funciones de costos.
  - Los valores que se incluirán en las tablas de costo deben ser leídos desde archivos de texto con los siguiente nombres: `cost_insert.txt`, `cost_delete.txt`, `cost_replace.txt`, `cost_transpose.txt`.
  - El alfabeto considerado para esta tarea es el alfabeto inglés y sólo letras minúsculas.
  - Cada archivo de texto tiene números enteros, espacios en blanco y saltos de línea un ejemplo a continuación (es parte de la tarea saber las dimensiones que deben tener las tablas para cada función de costo, incluyendo la verificación de que las dimensiones de la siguiente tabla son o no correctas):

```
0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0
```

Cuadro 1: Ejemplo de archivo de texto con los costos de una de las operaciones.

  - En el caso de las funciones que involucran sólo un carácter, los archivos serán de sólo una fila y en el caso de las funciones que involucran dos caracteres, los archivos serán de una matriz cuadrada. En ambos casos el formato de los archivos será como el mostrado en el ejemplo anterior.

Salida:

- La distancia mínima de edición entre las cadenas `S1` y `S2`.
- Opcionalmente, la secuencia de operaciones con el costo más bajo.

Aquí deben explicar la estructura de sus programas haciendo referencias a los archivos y funciones de su entrega. No adjunte código en esta sección.

3.4. Experimentos (máximo 6 páginas)

En la sección de Experimentos, es fundamental detallar la infraestructura utilizada para asegurar la reproducibilidad de los resultados, un principio clave en cualquier experimento científico. Esto implica especificar tanto el hardware (por ejemplo, procesador Intel Core i7-9700K, 3.6 GHz, 16 GB RAM DDR4, almacenamiento SSD NVMe) como el entorno software (sistema operativo Ubuntu 20.04 LTS, compilador g++ 9.3.0, y cualquier librería relevante). Además, se debe incluir una descripción clara de las condiciones de entrada, los parámetros utilizados y los resultados obtenidos, tales como tiempos de ejecución y consumo de memoria, que permitan a otros replicar los experimentos en entornos similares. *La replicabilidad es un aspecto crítico para validar los resultados en la investigación científica computacional* [1].

3.4.1. Datasets (máximo 2 páginas)

Diseña al menos **cinco casos de prueba** que cubran varios escenarios, incluyendo:

- Casos donde las cadenas están vacías.
- Casos con caracteres repetidos.
- Casos donde las transposiciones son necesarias.

Documentación de los Casos de Prueba:

- Proporciona las cadenas de entrada y los costos para cada operación.
- Muestra la salida esperada para cada caso de prueba, incluyendo la secuencia de operaciones y el costo total.
- Explica por qué la salida es correcta.

Es importante generar varias muestras con características similares para una misma entrada, por ejemplo, variando tamaño del input dentro de lo que les permita la infraestructura utilizada en ests informe, con el fin de capturar una mayor diversidad de casos y obtener un análisis más completo del rendimiento de los algoritmos.

> Aunque la implementación de los algoritmos debe ser realizada en C++, se recomienda aprovechar otros lenguajes como Python para automatizar la generación de casos de prueba, ya que es más amigable para crear gráficos y realizar análisis de los resultados. Python, con sus bibliotecas como `matplotlib` o `pandas`, facilita la visualización de los datos obtenidos de las ejecuciones de los distintos algoritmo bajo diferentes escenarios.
>
> Además, debido a la naturaleza de las pruebas en un entorno computacional, los tiempos de ejecución pueden variar significativamente dependiendo de factores externos, como la carga del sistema en el momento de la ejecución. Por lo tanto, para obtener una medida más representativa, siempre es recomendable ejecutar múltiples pruebas con las mismas características de entrada y calcular el promedio de los resultados.

3.4.2. Resultados de los Experimentos (máximo 4 páginas)

En esta sección, los resultados obtenidos, como las gráficas o tablas, deben estar respaldados por los datos generados durante la ejecución de sus programas. Es fundamental que, junto con el informe, se adjunten los archivos que contienen dichos datos para permitir su verificación. Además, se debe permitir y especficiar como obtener esos archivos desde una ejecución en otro computador (otra infraestructura para hacer los experimentos).
No es necesario automatizar la generación de las gráficas, pero sí es imprescindible que se pueda confirmar que las visualizaciones presentadas son producto de los datos generados por sus algoritmos, aunque la trazabilidad de los datos hasta las visualizaciones es esencial para garantizar que su validez: describa cómo se generaron los datos, cómo se procesaron y cómo se visualizaron de manera que pueda ser replicado por quien lea su informe.

3.5. Conclusiones (máximo 1 página)

La conclusión de su informe debe enfocarse en el resultado más importante de su trabajo. No se trata de repetir los puntos ya mencionados en el cuerpo del informe, sino de interpretar sus hallazgos desde un nivel más abstracto. En lugar de describir nuevamente lo que hizo, muestre cómo sus resultados responden a la necesidad planteada en la introducción.

4. Condiciones de entrega

- La tarea se realizará individualmente (esto es grupos de una persona), sin excepciones.
- La entrega debe realizarse vía `http://aula.usm.cl` en un tarball en el área designada al efecto, en el formato `tarea-2_3-rol.tar.gz` (`rol` con dígito verificador y sin guión).

  Dicho tarball debe contener las fuentes en LaTeX 2ε (al menos `tarea-2_3.tex`) de la parte escrita de su entrega, además de un archivo `tarea-2_3.pdf`, correspondiente a la compilación de esas fuentes.
- Si se utiliza algún código, idea, o contenido extraído de otra fuente, este **debe** ser citado en el lugar exacto donde se utilice, en lugar de mencionarlo al final del informe.
- Asegúrese que todas sus entregas tengan sus datos completos: número de la tarea, ramo, semestre, nombre y rol. Puede incluirlas como comentarios en sus fuentes LaTeX (en TeX comentarios son desde % hasta el final de la línea) o en posibles programas. Anótese como autor de los textos.
- Si usa material adicional al discutido en clases, detállelo. Agregue información suficiente para ubicar ese material (en caso de no tratarse de discusiones con compañeros de curso u otras personas).
- No modifique `preamble.tex`, `tarea_main.tex`, `condiciones.tex`, estructura de directorios, nombres de archivos, configuración del documento, etc. Sólo agregue texto, imágenes, tablas, código, etc. En el códigos funte de su informe, no agregue paquetes, ni archivos .tex (a excepción de que agregue archivos en `/tikz`, donde puede agregar archivos .tex con las fuentes de gráficos en `TikZ`).
- La fecha límite de entrega es el día 10 de noviembre de 2024.

NO SE ACEPTARÁN TAREAS FUERA DE PLAZO.

- Nos reservamos el derecho de llamar a interrogación sobre algunas de las tareas entregadas. En tal caso, la nota de la tarea será la obtenida en la interrogación.

NO PRESENTARSE A UN LLAMADO A INTERROGACIÓN SIN JUSTIFICACIÓN PREVIA SIGNIFICA AUTOMÁTICAMENTE NOTA 0.

A. Observaciones: Ejemplo de Entrada y Salida

Las funciones de costos para este ejemplo son las siguientes:

- `costo_sub(a, b)` = 2 si $a \neq b$, y 0 si $a = b$.
- `costo_ins(b)` = 1 para cualquier carácter b.
- `costo_del(a)` = 1 para cualquier carácter a.
- `costo_trans(a, b)` = 1 para transponer los caracteres adyacentes a y b.

Las dos cadenas de entrada son las siguientes:

- S1 = “intention”
- S2 = “execution”

y la salida esperada es la siguiente:

```
Distancia Mínima de Edición: 9
```

A continuación se muestra una secuencia posible de operaciones con costo total 5:

1) $i \rightarrow e$: Sustituir ’i’ por ’e’ en posición 1 (costo 2).
2) $n \rightarrow x$: Sustituir ’n’ por ’x’ en posición 2 (costo 2).
3) $t \leftrightarrow e$: Transponer ’t’ y ’e’ en posiciones 3 y 4 (costo 1).
4) $t \rightarrow c$: Sustituir ’i’ por ’c’ en posición 5 (costo 2).
5) $n \rightarrow u$: Sustituir ’o’ por ’u’ en posición 6 (costo 2).

El costo total es $2+2+1+2+2=9$.

Referencias

[1] Jorge Fonseca y Kazem Taghva. «The State of Reproducible Research in Computer Science». En: ene. de 2020, págs. 519-524. ISBN: 978-3-030-43019-1. DOI: 10.1007/978-3-030-43020-7_68.