Universidad de Chile
Departamento de Ciencias de la Computación
CC4102 - Diseño y Análisis de Algoritmos

Tarea 1 - "Distancia de Edición en Memoria Externa"

Profesor: Gonzalo Navarro
Auxiliares: Dustin Cobas
Bernardo Subercaseaux
Ayudantes: Javier Morales
Matías Rojas

1 Problema

Dados dos strings $\mathcal{X} = x_1x_2\ldots x_n$ y $\mathcal{Y} = y_1y_2\ldots y_m$ sobre un alfabeto finito Σ, la *distancia de edición* entre $\mathcal{X}$ y $\mathcal{Y}$ es el número mínimo de *operaciones* requeridas para transformar una cadena en la otra. Estas operaciones pueden ser: `delete`(x_i) que elimina x_i de $\mathcal{X}$; `insert`(y_j) que inserta y_j en $\mathcal{X}$; y `substitute`(x_i, y_j) que reemplaza x_i por y_j en $\mathcal{X}$.

El objetivo de esta tarea es analizar, implementar y evaluar experimentalmente los algoritmos descritos para computar la distancia de edición entre dos cadenas con longitud n en **memoria externa**. Cada algoritmo debe se analizado detalladamente, haciendo énfasis en: complejidad computacional; espacio utilizado tanto en memoria principal como en memoria secundaria; y, en especial, operaciones de lectura/escritura en memoria externa. Deberá confeccionar un informe donde indique claramente los siguientes puntos:

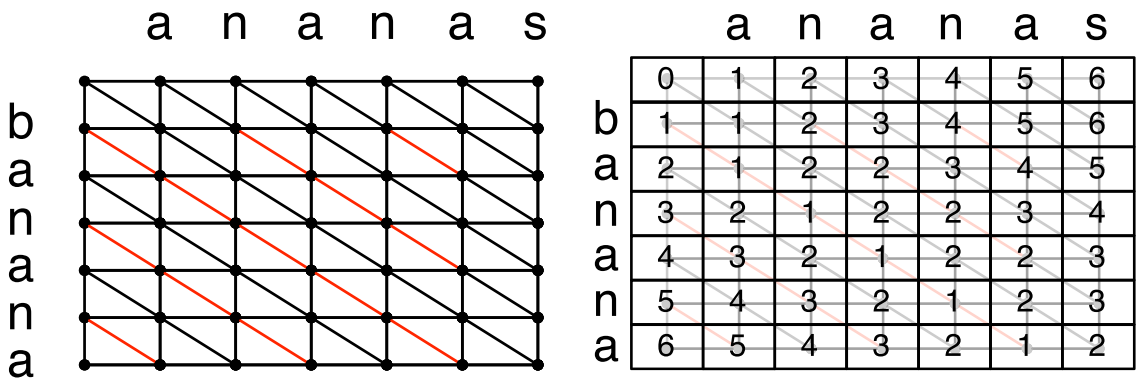
1. El *análisis teórico* de los algoritmos incluyendo los puntos mencionados anteriormente.
2. El *diseño experimental*, incluyendo los detalles de la implementación de los algoritmos, la generación de las instancias y las medidas de rendimiento utilizadas.
3. La *presentación de los resultados* en forma de una descripción textual, tablas y/o gráficos.
4. El *análisis e interpretación* de los resultados.

2 Grafo de Cuadrícula Implícito

Sean $\mathcal{X}$ and $\mathcal{Y}$ dos strings de longitud n, el *grafo de cuadrícula implícito* para $\mathcal{X}$ y $\mathcal{Y}$ se define como:

- Una cuadrícula o grilla 2D de $(n+1) \times (n+1)$ nodos.
- Todo nodo tiene una arista a sus vecinos en el Sur(S), en el Este(E) y en el Sureste(SE).
- Las E-aristas y S-aristas tienen peso 1.
- La SE-arista $(i-1, j-1) \to (i, j)$ tiene peso 0 si $\mathcal{X}[i] = \mathcal{Y}[j]$, y peso 1 en otro caso.

Teorema 1. *La longitud del camino más corto desde la celda $(0,0)$ a la (n,n) en el grafo de cuadrícula implícito para $\mathcal{X}$ y $\mathcal{Y}$ es la distancia de edición entre $\mathcal{X}$ y $\mathcal{Y}$.*



(a) Aristas Sur, Este y Sureste

(b) Caminos que parten desde $(0, 0)$

Figure 1: Grafo de cuadrícula implícito para las cadenas *banana* y *ananas*

Ejemplo 1. Sean $\mathcal{S} = banana$ y $\mathcal{T} = ananas$, la figura 1a muestra el grafo de cuadrícula implícito correspondiente. Las aristas **negras** tienen peso 1 y las **rojas** tienen peso 0. En la figura 1b se muestran los pesos de todos los caminos que parten desde la celda $(0, 0)$. La distancia de edición entre las cadenas *banana* y *ananas* es 2 y aparece en la celda $(6, 6)$.

3 Algoritmos

3.1 RAM

3.1.1 Programación Dinámica

1. Construir una matriz de $(n + 1) \times (n + 1)$, donde la casilla (i, j) contendrá la longitud del camino más corto desde $(0, 0)$ hasta (i, j).

2. Llenar la primera fila y la primera columna con los valores $0, 1, \ldots, n$ consecutivamente.

3. Calcular el valor para cada casilla secuencialmente, de izquierda a derecha, y de arriba hacia abajo. La casilla (i, j) es calculada usando las casillas $(i - 1, j)$ y $(i, j - 1)$, y la SE-arista $(i - 1, j - 1) \to (i, j)$.

4. Retornar el valor en la casilla (n, n).

Nota. Para calcular la matriz de $K \times K$, no se requiere almacenar las K^2 celdas en RAM en todo momento. Si efectuamos el cómputo de la matriz por filas (columnas), solamente se requiere acceder a la fila (columna) anterior para calcular la actual.

3.2 Memoria Externa

Los string $\mathcal{X}$ y $\mathcal{Y}$ son almacenados en n/B bloques consecutivos en memoria externa.

3.2.1 Algoritmo para RAM Adaptado

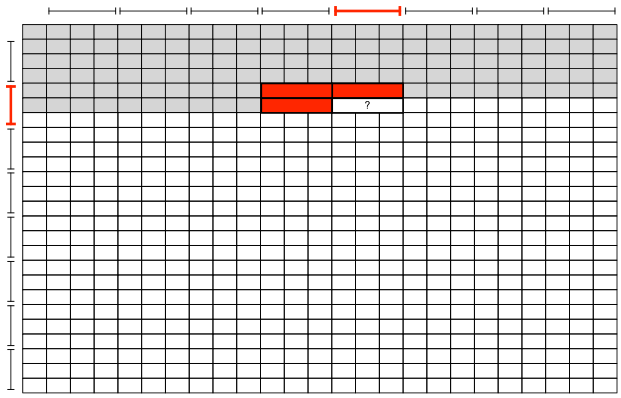


Figure 2: Algoritmo para RAM adaptado a memoria externa

El algoritmo para RAM adaptado a memoria externa es básicamente el mismo algoritmo descrito en la sección 3.1.1. La diferencia está en que al computar la matriz fila por fila, leerá cada fila bloque a bloque de la memoria externa. Por cada bloque leído, computará los valores del bloque correspondiente en la nueva fila y los escribirá a memoria externa (figura 2). Note que el bloque en la nueva fila puede reescribir el bloque en la fila anterior.

3.2.2 Particionando Cuadrilla

1. Dividir la cuadrilla en subtablas con fronteras solapadas (figura 3).
2. Procesar las subtablas de izquierda a derecha y de arriba hacia abajo. Por cada subtabla:
   (a) Leer de memoria externa los substrings y fronteras de entrada correspondientes.
   (b) Computar los valores para la subtabla usando el algoritmo para RAM (sección 3.1.1).
   (c) Escribir las fronteras de salida a memoria externa.

4 Implementación

Debe implementar los 2 algoritmos descritos para computar distancia de edición sobre memoria externa. La descripción dada para los algoritmos es bastante general, por lo que es requerido un

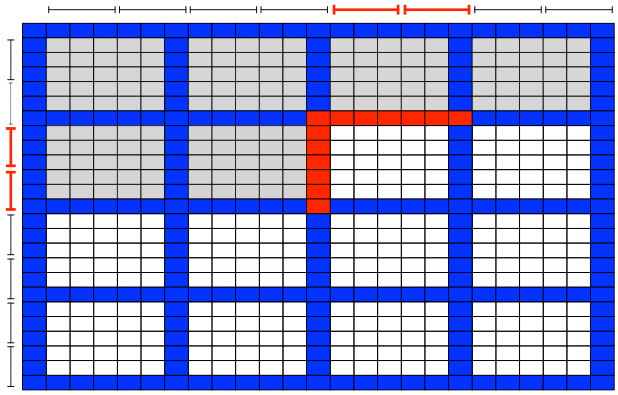


Figure 3: Algoritmo particionando la cuadrilla

análisis y diseño de los detalles faltantes.

Note que el algoritmo 3.2.2 requiere conocer el tamaño M de la RAM para determinar como particionar la cuadrilla, es decir, cual será el tamaño de las subtablas. Es recomendable que esto sea un parámetro en su implementación para facilitar la experimentación.

El objetivo es evaluar la eficiencia de los algoritmos sobre el modelo de memoria externa visto en clases, por lo que es primordial que las implementaciones se ejecuten usando operaciones de lectura/escritura a memoria externa siempre y cuando el algoritmo lo requiera (independientemente de que cuente con RAM suficiente para almacenar todas las estructuras de datos).

5 Experimentación

La experimentación se realizará utilizando datos sintéticos. Genere aleatoriamente los strings $\mathcal{X}$ y $\mathcal{Y}$ de tamaños $N \in \{2^{10}, \ldots, 2^{13}\}$.

Para cada par $\mathcal{X}$ y $\mathcal{Y}$ de tamaño N, deberá ejecutar los algoritmos 3.2.1 y 3.2.2 con $M = \{10 \times \frac{n}{2^4}, 10 \times \frac{n}{2^7}, 10 \times \frac{n}{2^{10}}\}$ con el objetivo de comparar sus comportamientos. Utilice como tamaño de bloque $B = 2^{10}$.

Además deberá evaluar y graficar el desempeño del algoritmo 3.2.2 para los strings anteriores y para nuevos strings de largo $N \in \{2^{14}, \ldots, 2^{16}\}$, mostrando como se comporta a medida que la RAM decrece. Use para esto, los mismos valores de M propuestos anteriormente.

Documente las características del hardware utilizado (CPU, RAM, HDD o SDD, ...), el sistema operativo, así como el lenguaje de programación y la versión del compilador usado.

6 Entrega de la Tarea

- La tarea puede realizarse en grupos de a lo más 3 personas.
- Para la implementación puede utilizar C, C++ o Java. Para el informe se recomienda utilizar LaTeX.
- Siga buenas prácticas (*good coding practices*) en sus implementaciones.
- Escriba un informe claro y conciso. Las ponderaciones del informe y la implementación en su nota final son las mismas.
- Tenga en cuenta las sugerencias realizadas en las primeras clases sobre la forma de realizar y presentar experimentos.
- La entrega será a través de U-Cursos y deberá incluir el informe junto con el código fuente de la implementación (y todas las indicaciones necesarias para su ejecución).
- Se permiten atrasos con un descuento de 1 punto por día.