CC4102 - Tarea 2

Prof. Gonzalo Navarro

Entrega: 3 de Diciembre de 2021

La tarea consiste en implementar un arreglo de sufijos y un árbol de sufijos para hacer búsquedas y autocompletado sobre un texto.

1. Arreglo y Árbol de Sufijos

Dado un texto $T[1..n]$ terminado en un símbolo especial \$ (que suele ser el cero en los lenguajes de programación típicos), el arreglo de sufijos es un arreglo $A[1..n]$ de posiciones de T, de manera que el sufijo $T[A[i]..]$ es lexicográficamente menor que el sufijo $T[A[i+1]..n]$ para todo i. En esta tarea lo construiremos simplemente con cualquier algoritmo de ordenamiento.

El arreglo de sufijos permite buscar las ocurrencias de una cadena de largo m en T en tiempo $O(m \log n)$, más lo necesario para reportar cada una de las ocurrencias. El procedimiento es esencialmente una búsqueda binaria.

El *árbol de sufijos*, en esta tarea, será simplemente un árbol Patricia al que se insertan todos los sufijos de T. En las hojas se guarda la posición de T donde comienza el sufijo. Note que, si los hijos del árbol Patricia se almacenan en orden lexicográfico del siguiente carácter, las hojas del árbol Patricia forman el arreglo de sufijos. En esta tarea se construirá este árbol simplemente insertando todos los sufijos de T uno a uno.

Puede leer sobre estas estructuras en la sección 4.3 del apunte. Nuestro árbol de sufijos tendrá punteros al texto para cada nodo, por lo que la búsqueda por las aristas es más sencilla: se pueden encontrar en el texto todos sus caracteres, en vez de la búsqueda de dos pasadas que se describe en el apunte.

2. A Implementar

La tarea consiste en estas cuatro etapas:

1. Hacer un programa que construya el arreglo de sufijos para el texto de un archivo. El programa recibe el nombre del archivo, lo trae a memoria, y construye su arreglo de sufijos. No es necesario que almacene el arreglo en disco.

2. Una vez construido el arreglo, el programa debe entrar en un loop en que pida al usuario una cadena a buscar. Debe usar el arreglo de sufijos para buscar las ocurrencias de la cadena, reportar cuántas ocurrencias encontró, y para cada una de ellas imprimir un *snippet* del texto donde ocurre. En este caso lo que imprimiremos será la línea del texto que contiene la ocurrencia. Esta línea se debe extraer a partir de la posición donde el arreglo de sufijos indica que está la ocurrencia, moviéndose hacia adelante y hacia atrás en el texto.

3. Hacer un programa que construya el árbol de sufijos para el texto de un archivo. El programa recibe el nombre del archivo, lo trae a memoria, y construye su árbol de sufijos. No es necesario que lo almacene el árbol en disco.

4. Una vez construido el árbol, el programa debe usarlo para permitir buscar con *autocompletado*. Para ello, preprocesar el árbol de sufijos de modo que cada nodo sepa la cantidad de hojas que tiene debajo y además tenga un puntero al texto de alguna hoja que descienda de él.

El autocompletado irá bajando en el árbol a medida que el usuario tipea, y a partir de un nodo ofrecerá continuar por los tres hijos con más descendientes. El programa debe permitir al usuario ir tipeando letra a letra lo que quiere buscar, y para cada letra, ir bajando por el árbol de sufijos. Hay dos posibilidades:

- Lo escrito hasta ahora me deja en el medio de una arista del árbol, no un nodo. Eso significa que todas las ocurrencias de lo tipeado hasta ahora continúan de la misma forma, hasta llegar al nodo hijo de la arista. En este caso el autocompletado debe directamente forzar las siguientes letras hasta llegar al nodo, pues de otro modo no habrá ocurrencias.
- Lo escrito hasta ahora me deja en un nodo del árbol. En este caso, tomamos los tres hijos con más descendientes y ofrecemos completar de esas tres formas, ordenando de más a menos frecuente. La cadena que se ofrece para completar es el string completo que representan las aristas que descienden a cada uno de esos tres hijos, el que puede contener un carácter o más.

Elija la interfaz que prefiera para realizar los puntos 2 y 4.

3. Bono

Puede obtener hasta dos puntos más en esta tarea (sobrepasando incluso la nota 7, que podrá chorrear a la primera tarea) con este punto adicional:

- Con el árbol de sufijos, haga un programa que reciba un largo mínimo ℓ y una frecuencia mínima f, e imprima todos los substrings del texto de largo al menos ℓ que aparezcan al menos f veces. No imprima un string que sea más corto que otro que también imprimirá. Esto se resuelve recorriendo el árbol de sufijos en DFS y buscando los nodos más profundos que cumplan con las condiciones.

4. Informe

Su informe debe incluir una descripción de su implementación y una explicación de cómo utilizar los programas en modo de usuario. Los programas serán probados con archivos de texto del Canterbury Corpus, `https://corpus.canterbury.ac.nz/descriptions/`.

Será suficiente que sus programas funcionen sobre archivos de hasta 1 MB, pero se premiará que soporten archivos mayores, como el `world192.txt`. o el `bible.txt` del Large Corpus. Asimismo, se invita a probar, reportar y explicar sus resultados sobre el archivo artificial `aaa.txt`.