Ejercicios Tema 4

Algoritmia para problemas difíciles

Departamento de Informática e Ingeniería de Sistemas
Escuela de Ingeniería y Arquitectura – Universidad de Zaragoza

5 de diciembre de 2018

Ejercicios sobre Árboles de sufijos

Ejercicio 1 (Ejercicio de investigación biliográfica) Explicar detalladamente como funciona el algoritmo de compresión de Lempel-Ziv (LZ77) y cómo se implementa utilizando los árboles de sufijos. Averiguar cómo puede implementarse de forma muy eficiente (para ello puede ser útil averiguar algo sobre el algoritmo de Ukkonen de construcción de árboles de sufijos).

Ejercicio 2 La relación entre el árbol de sufijos de una cadena y el árbol de sufijos de la cadena inversa (en orden contrario) no es obvia. Sin embargo, hay una relación significativa entre esos dos árboles, encontrarla, enunciarla y demostrarla. (Ejercicio no trivial)

Ejercicio 3 El árbol de sufijos generalizado de N cadenas $t_1, \ldots, t_N$ es el árbol de sufijos de la cadena $t_1\$t_2\$\ldots\$t_N$ (es decir, las N cadenas concatenadas con el símbolo de separación \$ distinto a cualquier otro símbolo). Las hojas del árbol generalizado se indexan como (i, j) para indicar "la posición j dentro de t_i".
Considérese un árbol de sufijos generalizado para almacenar los sufijos de N cadenas $t_1, \ldots, t_N$. ¿Cómo se pueden añadir otra cadena t_{N+1}? ¿Y borrar una de las cadenas t_i?

Ejercicio 4 Considera el árbol de sufijos de t con longitud n. Sea p un string de longitud m. Busca como aumentar dicho árbol de sufijos para permitir las siguientes operaciones:

1. *stringmatchingAntes*(p, t, e) que encuentra una aparición de p en t que comienza en una posición cualquiera anterior a e, si existe. Esta operación debe funcionar en tiempo $O(m \log n)$ y usar memoria adicional $O(n \log n)$.
2. *stringmatchingCerca*(p, t, e) que encuentra la aparición de p en t que comienza en una posición anterior a e, si existe. Si p ocurre varias veces antes de e debe devolver la más cercana a e. Esta operación debe funcionar en tiempo $O(m \log n)$ y usar memoria adicional $O(n^2)$.
3. *stringmatchingEntre*(p, t, s, e) que encuentra la aparición de p en t que comienza en una posición cualquiera posterior a s y anterior a e, si existe. Esta operación debe funcionar en tiempo $O(m \log n)$ y usar memoria adicional $O(n^2)$.

Ejercicio 5 Diseñar un algoritmo en tiempo $O(n \log n)$ que, dado una cadena t de longitud n, calcule el número de subcadenas distintas de t.

Ejercicio 6 ¿Cómo podría hallarse muy eficientemente el palíndromo más largo que sea subcadena de una cadena dada?

Ejercicio 7 El problema de string matching es demasiado restrictivo para muchas aplicaciones en genómica donde, ya sea debido a la secuencia de errores o mutaciones al azar, una cadena patrón p podría no coincidir exactamente con ninguna posición del texto t aunque casi coincida. Consideramos la siguiente variante aproximada: dada una cadena t de longitud n, una cadena p de longitud m, y un valor $k \geq 0$, determinar si hay una subcadena de longitud m de t que coincide con p en todo menos en como mucho en

k posiciones. Diseñar un algoritmo en tiempo $O((kn + m) \log n)$ para este nuevo problema y demostrar que el algoritmo es correcto.

Ejercicio 8 Un palíndromo con k errores es un string xy donde $|x| = |y|$ y el reverso de y es igual a x excepto en k posiciones. Dar un algoritmo que encuentre todos los palíndromos con k errores dentro de S, string de longitud n (es decir, palíndromos con k errores de la forma $S[i..j]$ para algún i, j).

Ejercicio 9 Dado un string s de longitud n, escribir un algoritmo en tiempo $O(n \log n)$ que encuentre el string t más largo tal que tanto t como su reverso ocurren en s. Debes usar árboles de sufijos (normales o generalizados).
Por ejemplo si $s = yabcxqcbaz$, tu algoritmo debe devolver $t = abc$ ó $t = cba$ porque abc y su reverso cba aparecen en s y ningún string más largo lo cumple.

Notas: El **árbol de sufijos generalizado** de N cadenas $t_1, \ldots, t_N$ es el árbol de sufijos de la cadena $t_1\$t_2\$\ldots\$t_N$ (es decir, las N cadenas concatenadas con el símbolo de separación \$ distinto a cualquier otro símbolo). Las hojas del árbol generalizado se indexan como (i, j) para indicar "la posición j dentro de t_i". En algunas aplicaciones pueden utilizarse $N - 1$ símbolos de separación distintos $\$_1, \ldots \$_{N-1}$ en lugar de uno solo \$.

El **reverso** de una cadena $s = a_1 \ldots a_n$ es la misma cadena leída de fin a principio, es decir, $s' = a_n a_{n-1} \ldots a_1$.

En caso de entregar alguno de estos ejercicios, la fecha límite es el jueves 10 de enero.

Antes de realizar cualquiera de estos ejercicios el alumno debe enviar un correo a `elvira@unizar.es` indicando qué ejercicio desea realizar.

Cualquier fuente utilizada en la resolución de estos ejercicios debe ser indicada claramente en la solución.