Escuela Superior de Cómputo.
Instituto Politécnico Nacional, México.

Práctica 8: Subsecuencia común más larga.

Blancas Pérez Bryan Israel
orionmunecaycanica@gmail.com

Resumen: Implementar el algoritmo de la subsecuencia común más larga, con el enfoque de programación dinámica, para hallar que tan similares son dos archivos analizados.

Palabras Clave: Subsecuencia común más larga, Programación dinámica, Similitud entre cadenas, Complejidad Computacional.

1. Introducción

La programación dinámica, es un método para reducir el tiempo de ejecución de un algoritmo.[1]. Este método, proporciona un procedimiento sistemático para encontrar la combinación de decisiones que maximice la efectividad total, descomponiendo el problema en etapas, y al final enlazando cada etapa a través de recursividad.[2]
En está práctica, se aplicará este método de programación para resolver el problema de encontrar la subsecuencia común más larga entre dos cadenas de texto, en un tiempo razonable a comparación del algoritmo por fuerza bruta.

2. Conceptos Básicos

Subsecuencia Común Más Larga

El problema de la Subsecuencia Común Más Larga, (Longest Common Subsequence, LCS por sus siglas en inglés) trata de encontrar una subsecuencia más larga que es común en un conjunto de secuencias. El problema de LCS es uno de los problemas clásicos de las ciencias computacionales y es la base de programas que comparan datos, y tiene varios usos, como en la bioinformática. [3]
Cabe destacar, que para un número grande se secuencias a comparar, este problema se considera NP-hard. En esta práctica, solo obtendremos el LCS de dos secuencias; el algoritmo implementado aquí, está hecho bajo el paradigma de programación dinámica, con un tiempo computacional $O(mn)$, donde m y n son las longitudes de las dos secuencias.

Ejemplo de una subsecuencia común:

Sea $A :< XMJYAUZ >$ y $B :< MZJAWXU >$, entonces el LCS (llamado C), sería $C :< MJAU >$.

Con este ejemplo, se aprecia, que no importa que las letras estén juntas, la única condición es que deben de estar en el mismo orden (secuencia).

Usando la programación dinámica para encontrar el LCS, el algoritmo construye una matriz L[0..m, 0..n] donde L(i, j) representa el LCS para x1 . . . i y y1 . . . j. Cada valor en la matriz se calcula basándose en los valores de las posiciones vecinas anteriores.[4]

$$L(i,j) = \begin{cases} 0 & \text{si } i = 0 \text{ o } j = 0 \\ L(i-1,\, j-1) + 1 & \text{si } x_i = y_j \\ \text{máx.}\,\{L(i-1,\, j), L(i,\, j-1)\} & \text{si } x_i \neq y_j \end{cases}$$

Figura 1: Ecuación para el algoritmo.

La matriz generada para el ejemplo de la página anterior se muestra en la figura 2, en donde está remarcado de color amarillo el camino de la LCS, y con números remarcados, el carácter correspondiente.

		0	1	2	3	4	5	6	7
		Ø	M	Z	J	A	W	X	U
0	Ø	0	0	0	0	0	0	0	0
1	X	0	0	0	0	0	0	1	1
2	M	0	1	1	1	1	1	1	1
3	J	0	1	1	2	2	2	2	2
4	Y	0	1	1	2	2	2	2	2
5	A	0	1	1	2	3	3	3	3
6	U	0	1	1	2	3	3	3	4
7	Z	0	1	2	2	3	3	3	4

Figura 2: Ecuación para el algoritmo.

3. Experimentación y Resultados

Ejercicio 1.
Implementar un programa que como entrada tenga dos archivos en modo de texto y como salida muestre en porcentaje que tan parecidos o similares son lo archivos. Para ello:
i) Omita espacios, es decir, los espacios no se consideran.
ii) Reporte 5 parejas de ejemplos donde muestre el porcentaje de similitud.

Pseudocódigo del algoritmo:

```
1  LCS(X[1..m], Y[1..n])
2    C = array(0..m, 0..n)
3    for i = 0 to m do
4      C[i,0] = 0
5    for j = 0 to n do
6      C[0,j] = 0
7    for i = 1 to m do
8      for j = 1 to n do
9        if X[i] == Y[j]
10         C[i,j] = C[i-1,j-1] + 1
11       else
12         C[i,j] = max(C[i,j-1], C[i-1,j])
13 return C[m,n]
14
15 findLCS(C[0..m,0..n], X[1..m], Y[1..n], i, j)
16   if i = 0 or j = 0
17     return ""
18   else if X[i] = Y[j]
19     return findLCS(C, X, Y, i-1, j-1) + X[i]
20   else
21     if C[i,j-1] > C[i-1,j]
22       return findLCS(C, X, Y, i, j-1)
23     else
24       return findLCS(C, X, Y, i-1, j)
```

La función LCS, crea la matriz solución, mientras que la función findLCS, encuentra la subsecuencia común más larga.

Pareja 1.

Input.
A: (100 de semejanza).
B: (100 de semejanza).
Output.

```
bryan@bryan-ubuntu:~/Documentos/Análisis de Algoritmos/practica8/codigo$ ./ejercicio1 input1a.t
xt input1b.txt
Contenido Archivo 1: 100 de semejanza
Contenido Archivo 2: 100 de semejanza
LCS (sin considerar espacios): 100desemejanza
Porcentaje de semejanza: 100.00
```

Figura 3: Pareja 1.

Como las dos cadenas son exactamente iguales, el porcentaje de semejanza es 100.
Ejemplo de matriz solución generada (por estética del reporte, en las siguientes ejecuciones no pondré la matriz):

```
0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   1   1   1   1   1   1   1   1   1   1   1   1   1
0   1   2   2   2   2   2   2   2   2   2   2   2   2
0   1   2   3   3   3   3   3   3   3   3   3   3   3
0   1   2   3   4   4   4   4   4   4   4   4   4   4
0   1   2   3   4   5   5   5   5   5   5   5   5   5
0   1   2   3   4   5   6   6   6   6   6   6   6   6
0   1   2   3   4   5   6   7   7   7   7   7   7   7
0   1   2   3   4   5   6   7   8   8   8   8   8   8
0   1   2   3   4   5   6   7   8   9   9   9   9   9
0   1   2   3   4   5   6   7   8   9   10   10   10   10
0   1   2   3   4   5   6   7   8   9   10   11   11   11
0   1   2   3   4   5   6   7   8   9   10   11   12   12
0   1   2   3   4   5   6   7   8   9   10   11   12   13
```

Figura 4: Matriz solución-Pareja 1.

Pareja 2.

Input.

A: (veamos que tan semejantes con este archivo 2a).

B: (con este otro archivo 2b).

Output.

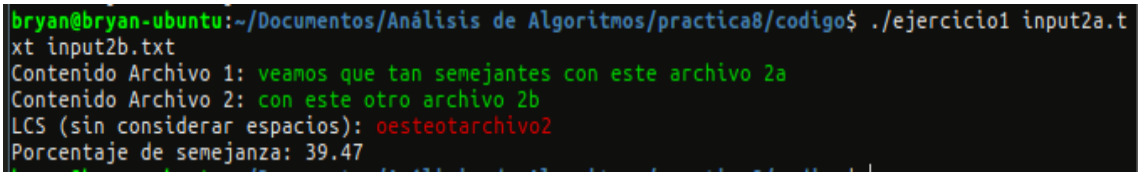

Figura 5: Pareja 2.

Pareja 3.

Input.

A: (There's a lady who's sure All that glitters is gold And she's buying a stairway to heaven When she gets there she knows If the stores are all closed With a word she can get what she came for Oh oh oh oh and she's buying a stairway to heaven).

B: (Nobody gonna take my car I'm gonna race it to the ground Nobody gonna beat my car It's gonna break the speed of sound Oooh it's a killing machine It's got everything Like a driving power big fat tires And everything).

Output.

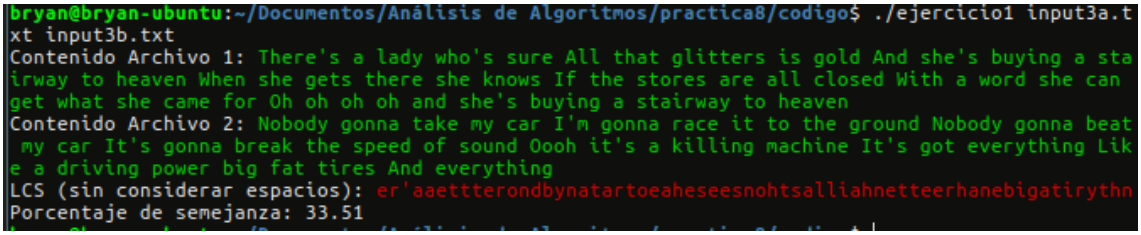

Figura 6: Pareja 3.

Pareja 4.

Input.

A: (El Longest common subsequence problem (Problema de subsecuencia común más larga) también conocido como LCS problem, se trata de encontrar una subsecuencia más larga que es común en un conjunto de secuencias (Aunque en la mayor parte solamente se toman dos secuencias).).

B: (La subsecuencia es cualquier subconjunto de elementos que mantienen el orden, entonces el problema de encontrar la subsecuencia común mas larga se define como : Dadas 2 cadenas , sean X e Y, encontrar la longitud de la cadena mas larga Z, tal que los caracteres de Z aparezcan en en ese mismo orden dentro de las cadenas X e Y.).

Output.

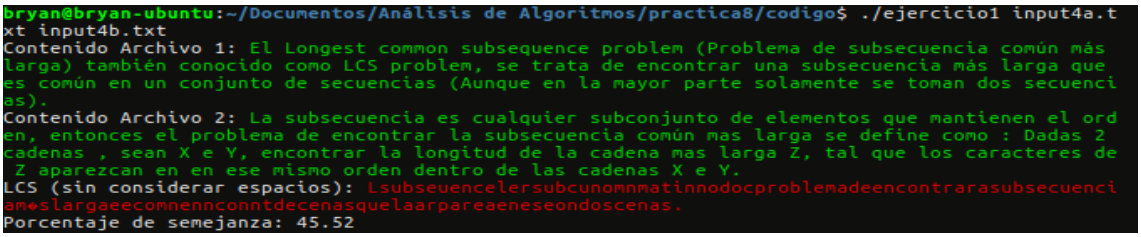

Figura 7: Pareja 4.

Pareja 5.

Input.

A: (bryan israel blancas pérez).

B: (antonio issac blancas pérez).

Output.

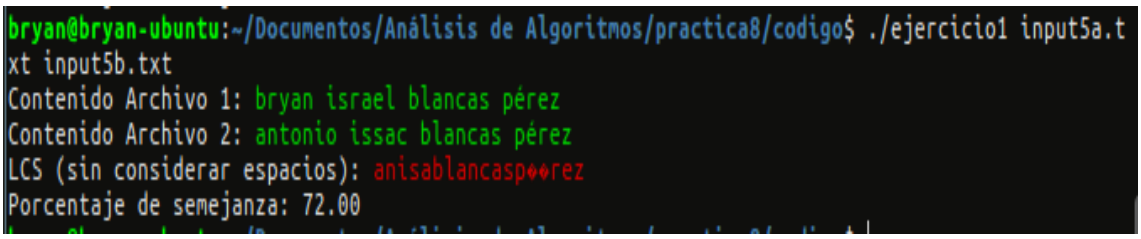

Figura 8: Pareja 5.

4. Conclusiones

Esta práctica me gustó mucho, porque programar el algoritmo tiene cierto grado de complejidad, sobretodo en fijarse bien en los índices de los arreglos. Además, este ejemplo es muy bueno para ejemplificar el uso de la programación dinámica, pasando de un tiempo computacional muy grande para el algoritmo de fuerza bruta, a un tiempo razonable aplicando ésta técnica.
Pienso que esta práctica es buena para mejorar las técnicas de programación y para la lógica detrás de encontrar un algoritmo eficiente para la resolución de un problema.

5. Bibliografía

[1] https://es.wikipedia.org/wiki/Programación_dinámica.

[2] http://www.ingenieria.unam.mx/sistemas/PDF/Avisos/Seminarios/SeminarioV/Sesion6_IdaliaFlores_20abr15.pdf

[3] https://es.wikipedia.org/wiki/Problema_de_Subsecuencia_Comú_mas_Larga

[4] http://www.bdigital.unal.edu.co/2754/1/299716.2010.pdf