David Rojas B 96873
José David Delgado B 92570

Propuesta del Algoritmo

Primeramente se calcula la longitud del LCS más largo, la complejidad computacional de este algoritmo es relativamente baja comparado con imprimir cual es concretamente la subsecuencia más larga. De hecho el problema de hallar la longitud de la subsecuencia más larga ha sido ampliamente estudiado y si se implementa el algoritmo correctamente se puede alcanzar una complejidad temporal de O(mn).

Para esto se utiliza una técnica llamada memorización (“memoization”) la cual es utilizada para evitar calcular un mismo resultado múltiples veces en un método recursivo. Se utiliza una tabla implementada como una matriz donde las filas corresponden al primer string y las columnas al segundo string, además inicialmente la matriz contendrá -1 en todas sus entradas. Particularmente, en la matriz la i-ésima fila corresponde al i-ésimo carácter del primer string y la i-ésima columna hace referencia al i ésimo carácter del segundo string. Además la entrada (i,j) de dicha matriz corresponderá a la longitud de la subsecuencia más larga entre el substring que va desde [0-i] en el primer string y el substring que va desde [0,j] en el segundo string. Al hacer un llamado recursivo, se revisa en la matriz si ya el cálculo se realizó para no hacerlo otra vez.

Conociendo este número se puede mejorar significativamente el algoritmo convencional de "fuerza bruta" para hallar cúal es la subsecuencia más larga , debido a que este genera las subsecuencias de todas las longitudes posibles,
mientras que si sabemos la longitud correcta de la subsecuencia más larga, se pueden generar únicamente las subsecuencias de esa longitud en particular.

La implementación de este algoritmo tiene la ventaja de que es sencillo de paralelizar ya que se puede emplear MPI para calcular todas las subsecuencias de una longitud dada . Realmente calcular la longitud del LCS no es una tarea compleja, por lo que puede ser realizada por el primer proceso sin necesidad de paralelizar.

Ejemplo del Algoritmo

Suponga que se quiere hallar la subsecuencia más larga entre las hileras: “ACADB” y “CBDA”. El primer paso consiste en únicamente encontrar la **longitud** de la subsecuencia más larga. En este caso la longitud corresponde a 2. Sabiendo esto se calculan todas las subsecuencias de longitud 2 de la primera hilera. Es decir: AC,AA,AD,AB,CA,CD,CB,AD,AB,DB y se busca cual de ellas es una subsecuencia de la segunda hilera. En este caso,dicha subsecuencia sería CA y esta sería la solución final.

Paralelización del Algoritmo

Asumiendo que hay T trabajadores, se reparten equitativamente las tareas (siempre sea posible), es decir tamaño del string / T. Por ejemplo si hay T=3 trabajadores y la hilera “A” corresponde a “ABCDEFGHI”(tamaño 9) entonces al primer trabajador le corresponde calcular las subsecuencias de un tamaño dado(por nosotros) que empiecen en A, B Y C. Luego al segundo trabajador le corresponde calcular las subsecuencias que empiecen en D,E y F, mientras que al tercero las que comienzan en G,H,I.

Referencias Consultadas

ishayadav181. (2020). *Longest common subsequence.* Codesdope. Recuperado de: https://www.codesdope.com/blog/article/longest-common-subsequence/

Printing Longest Common Subsequence. (2020, June 10). Retrieved November 20, 2020, from https://www.geeksforgeeks.org/printing-longest-common-subsequence/