Campus Monterrey

Mónica Andrea Ayala Marrero	*A01707439*
José Eduardo Puentes Martínez	*A01733177*
Hedguhar Domínguez González	*A01730640*
Axel Uzeta Gómez	*A00829417*

E1. Actividad Integradora 1

Análisis y diseño de algoritmos avanzados (Gpo 570)

Profesor:

Salvador E. Venegas-Andraca

Julio 21, 2023

Descripción del funcionamiento de los algoritmos utilizados

Dentro de la solución de nuestra actividad integradora 1, se hizo uso de dos principales algoritmos los cuales fueron clave para la solución del problema, cada uno con distinto objetivo dentro del programa, estos dos algoritmos fueron el algoritmo KMP (Knuth-Morris-Pratt) y el algoritmo de Manacher, ambos siendo de suma importancia historica asi como eficientes. primeramente hablaremos del algoritmo KMP adentrándonos a su funcionamiento paso a paso, el objetivo del algoritmo y como este nos ayudó en la solución de nuestro problema. Ahora bien conozcamos un poco más sobre la historia del algoritmo KMP, este algoritmo tiene como objetivo el de toda presencia de un patrón dado dentro de un texto de tamaño significativo y que a su vez este sea eficiente realizando dicha acción, el algoritmo KMP fue dado a conocer en 1977 por Donald Knuth, Vaughan Pratt y James H. Morris, de aquí el nombre de dicho algoritmo.

KMP

Ahora bien ahora que conocemos un poco de su historia y del objetivo general por el cual este algoritmo fue creado podemos adentrarnos más en su funcionamiento, como este algoritmo realiza las búsqueda y procesa la información, para este desglosamos el funcionamiento del algoritmo en pasos de manera que sea más intuitivo y fácil de comprender.

- **Tabla LPS(Longest Prefix Suffix)**
    - La tabla LPS tiene el objetivo de no ser redundante durante las búsquedas, de esta manera no se realizarán comparaciones que no añadan valor durante el proceso y así aumentando su eficiencia. Esta tabla tendrá distinta información del patrón el cual estamos buscando como la longitud del prefijo y sufijo del mismo asignando posiciones.

    - Existe una equivalencia dentro de la tabla LPS y es que cada posición de la tabla LPS representa el índice correspondiente en el patrón a buscar, y el valor que está en dicha posición corresponde a la longitud del prefijo más largo.

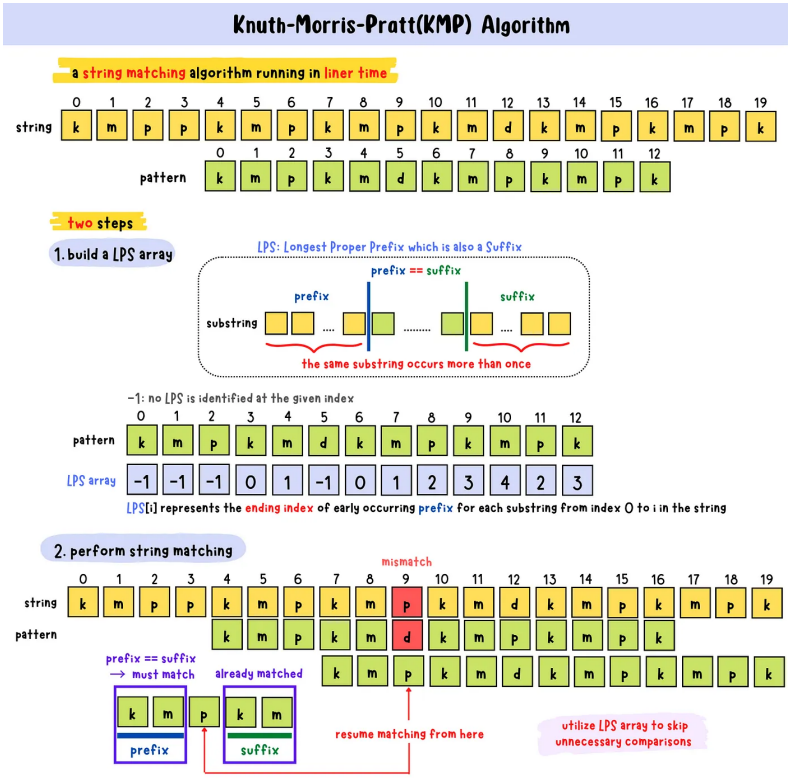
- **Búsqueda**
    - Como el nombre del paso bien lo indica una vez construida la tabla LPS se iniciara el proceso de busqueda del patron para lo cual se inicializar dos índices de iteración en 0, con el objetivo de comparar los caracteres iniciales tanto del patrón como del texto en el cual se desea encontrar el patrón. Ahora bien en este caso de comparar los caracteres del patrón con el texto pueden suceder dos escenarios en el cual o puede haber coincidencia entre caracteres o no hay coincidencia dichos escenarios se explicaran a continuación.

- **Coincidencia**
    - En caso de coincidencia y que el segundo índice de iteración sea igual a la longitud del patrón deseado menos uno (Es decir si el segundo índice de iteración es 4 y la longitud de mi patrón es 5), tenemos que exitosamente se encontró una presencia del patrón en el texto, si esto sucede entonces después actualizamos el segundo índice con ayuda de la tabla LPS y se incrementa el primer índice correspondiente al texto para seguir con la busqueda de mas coincidencias.

- **No Coincidencia**
    - En caso de no existir coincidencia y que el segundo índice correspondiente al patrón no sea igual a cero entonces se actualizará el segundo índice con ayuda de la tabla LPS en caso contrario de que el segundo índice sea igual a cero, nos iremos a incrementar el primer índice correspondiente al texto para continuar con la busqueda del patron.

- **Repetición**
    - Finalmente una vez se haga todo este proceso por primera vez los pasos de búsqueda que a su vez implican los de coincidencia y no coincidencia se repetirán hasta haber analizado todo el texto.

Para poder hacer más fácil la comprensión de dicho algoritmo nos gustaría añadir una imagen la cual a nuestra consideración representa de manera sencilla y bastante visual con distintos colores el como es el funcionamiento de este algoritmo paso a paso.



(Lee, 2022)

Dentro de la imagen podemos observar de manera visual la construcción del LPS, el uso de los sufijos y prefijos así como la búsqueda del patrón dentro del texto.

Manacher

Una vez conocido el algoritmo de KMP, iremos con el siguiente algoritmo utilizado dentro de nuestra solución el cual es el algoritmo de Manacher, para esto primeramente conoceremos un poco sobre su historia para poder ver su funcionamiento. El objetivo del algoritmo de Manacher es el de encontrar la

subcadena palindroma más larga dentro de una cadena dada por supuesto todo esto de la forma más eficiente posible, dicho algoritmo fue realizado en 1975 por Michel Manacher de ahí el nombre de este algoritmo, ahora bien pasaremos a explicar su funcionamiento por pasos para comprender de mejor manera como este nos fue de ayuda en la solución de nuestra actividad.

- **Modificación en la cadena**
    - Antes que todo se modificara la cadena original para que de esta manera cualquier palíndromo tenga su centro bien establecido, posteriormente a esto se insertarán caracteres especiales entre cada carácter de la cadena y en los extremos de manera que se simplificará la identificación de los palíndromos.

- **Inicialización**
    - Se inicializan dos variables para la expansión de los palíndromos, la primera variable será asignada a representar el centro derecho del palíndromo más largo (conocido en ese momento de la iteración), la segunda variable indicará el extremo derecho del palíndromo antes mencionado y a manera de historial se creará una matriz la cual nos será de ayuda para añadirle la información de los palíndromos conocidos.

- **Expansión**
    - Empezaremos a recorrer la cadena a la cual le añadimos los caracteres especiales, si el iterador esta dentro del rango de el palíndromo conocido en ese momento, entonces se hará uso de la simetría de manera que utilizaremos la información de la matriz la cual guarda la información de los palíndromos conocidos para así poder realizar una estimación de la longitud del palíndromo. posteriormente se realizará la expansión del palíndromo hacia los extremos izquierdo y derecho y en base a esto se actualizará la información de la matriz en caso de que se encuentren palíndromos más grandes, si el palíndromo se extiende más allá de los bordes que definimos previamente en las dos variables entonces actualizaremos las dos variables dando como resultado que existe un palíndromo más grande.

- **Búsqueda**
    - Finalmente y en base al proceso visto anteriormente mientras más avancemos en la cadena se irá registrando el palíndromo más grande, así al finalizar obtendremos la subcadena palindroma más grande.

LCS

Y ya para finalizar con la explicación de los algoritmos utilizados en la solución de la actividad explicaremos un poco de la historia y del funcionamiento del algoritmo LCS(Longest Common Subsequence), dicho algoritmo tiene el objetivo de encontrar la longitud de la subsecuencia común más larga entre dos o más cadenas dadas. Este algoritmo fue introducido inicialmente en 1957 por Michael O. Rabin y Richard M. Karp.

Ya que conocemos sobre el objetivo y la historia del algoritmo LCS procederemos a explicar su funcionamiento paso por paso para comprender cómo dicho algoritmo fue de gran ayuda para elaborar la solución de nuestro problema.

- **Matriz**
    - Se creará una matriz la cual tendrá un tamaño de n + 1 x m +1, “n” corresponde a la longitud de la cadena A y “m” será la longitud de la cadena B. (se inicializan los espacios de la matriz en 0).

- **Matriz llena**
    - Ahora se recorren tanto la cadena A como la B para llenar la matriz y calcular la longitud de la subsecuencia común más larga para cada subcadena. Dentro de este proceso pueden existir dos casos el primero sería en el la posición de los caracteres de las cadenas A y B coinciden, si esto sucede se incrementará el valor de la celda en base al valor de la celda más uno. En el caso de que los caracteres sean diferentes se tomará el valor máximo entre el primer iterador y el segundo iterador,(Se continuará el proceso hasta recorrer toda la matriz).

- **Longitud LCS**
    - Finalmente en donde se encontrará la subsecuencia común más larga será en la ubicación (n,m) las cuales definimos anteriormente siendo “n” la longitud de la cadena A y “m” de la B.

Complejidad de los algoritmos utilizados

KMP

Dado que este algoritmo es una comparación constante, de caracteres, y esta comparación es lineal, y lo que hace mejor es almacenar información para poder hacer una comparación futura mejor, pero al final de cuentas es una comparación lineal, por lo que se pasa una vez por el string, siendo una complejidad final O(n).

Manacher

En este algoritmo se transforma constantemente un string de comparación palíndrica, pero de igual manera, esta función itera de manera lineal por el string mayor, resultando de igual manera en una complejidad lineal O(n).

LCS

Dado que en este algoritmo de utiliza una matriz para almacenar las cadenas comparadas y se revisan constantemente estas en base a la programación dinámica, esta matriz que almacena las longitudes y las posiciones de las substrings se representa una iteración en función de las longitudes de las cadenas a comparar, representando las longitudes de las cadenas como m y n, la complejidad resulta en O(m*n)

Distintas aplicaciones de los algoritmos utilizados

Los algoritmos mencionados anteriormente tienen diversas aplicaciones en el campo de la informática y la ciencia de datos:

KMP

Búsqueda de patrones de texto, este algoritmo es ampliamente utilizado en aplicaciones que requieren la búsqueda eficiente de patrones, como editores de texto, motores de búsqueda, análisis de texto, procesamiento de lenguaje natural, etc. Además de utilizarse también para comprimir archivos y datos, y poder analizar mejor repeticiones en secuencias genómicas.

Manacher

Este algoritmo es bueno para encontrar subcadenas palindrómicas que sean de mayor tamaño, en este caso, sus aplicaciones son parecidas a las del algoritmo KMP, pero dando la posibilidad de análisis palíndromos y su eficiencia con este tipo de textos, desbloquea un mayor abanico de posibilidades para el análisis de datos.

LCS

Con la posibilidad de poderse enfocar en una herramienta de encontrar el substring más largo, este algoritmo desbloquea la posibilidad de poder aplicarlo en sentidos como la detección de plagio, tango en documento de texto como otras expresiones de datos comparables, a su vez, toma un rol más importante en la compresión de datos al poder relacionar substrings que se compartan con la mayor longitud, consiguiendo una mayor compresión por relación.

Referencias

Lee, C. (2022, October 16). Knuth-Morris-Pratt(KMP) algorithm: String matching - Claire Lee - medium. Medium. https://yuminlee2.medium.com/knuth-morris-pratt-kmp-algorithm-string-matching-fb2a3ec6d682

DAA Knuth-Morris-Pratt Algorithm - javatpoint. (n.d.). www.javatpoint.com. https://www.javatpoint.com/daa-knuth-morris-pratt-algorithm

Shiksha Online. (2023, July 7). Matching Strings: An Introduction to Knuth-Morris-Pratt (KMP) Algorithm. Retrieved July 21, 2023, from https://www.shiksha.com/online-courses/articles/introduction-to-kmp-algorithm/

Manacher’s Algorithm - Finding all sub-palindromes in O(N) - Algorithms for Competitive Programming. (n.d.). https://cp-algorithms.com/string/manacher.html

Longest Common Subsequence - javatpoint. (n.d.). www.javatpoint.com. https://www.javatpoint.com/longest-common-subsequence

Ivanovs, A. (2023, July 3). Longest Common Subsequence (LCS) - Glossary & definition. Stack Diary. https://stackdiary.com/glossary/longest-common-subsequence-lcs/