Similitud entre secuencias de ADN

Ejercicio N° 1

Objetivos	• Buenas prácticas en programación de Tipos de Datos Abstractos (TDAs) • Modularización de sistemas • Correcto uso de recursos (memoria dinámica y archivos)
Instancias de Entrega	**Entrega 1:** clase 4. **Entrega 2:** clase 6.
Temas de Repaso	• Uso de structs y typedef • Uso de macros y archivos de cabecera • Funciones para el manejo de Strings
Criterios de Evaluación	• Criterios de ejercicios anteriores • Cumplimiento de la totalidad del enunciado del ejercicio • Ausencia de variables globales • Correcta encapsulación en TDAs y separación en archivos • Uso de interfaces para acceder a datos contenidos en TDAs • Empleo de memoria dinámica de forma ordenada y moderada • Acceso a información de archivos de forma ordenada y moderada

Índice

Introducción

El ADN (ácido desoxirribonucleico) es una cadena de bases nitrogenadas que pueden ser de subtipos

- Adenina (A)
- Timina (T)
- Citosina (C)
- Guanina (G)

Dado que cada eslabón de la secuencia se diferencia de otro por su posición y su subtipo de base, las secuencias de ADN se las representan simplemente como cadenas de texto, siendo cada caracter la inicial de su subtipo. [1]

Por ejemplo, la siguiente secuencia de ADN consta de 4 eslabones, 2 Adeninas, 1 Timina y una Guanina:

```
ATAG
```

Con el paso del tiempo, una secuencia de ADN puede sufrir mutaciones puntuales cambiando una base nitrogenada por otra, puede sufrir remociones (pérdida de eslabones) o inserciones (nuevos eslabones son insertados).

La comparación entre 2 secuencias de ADN para determinar que secciones son iguales en ambas es una pieza fundamental en los estudios e investigaciones actuales.

En una comparación eslabón a eslabón se puede ver

```
ATAG
ATAA
===X   ← resultado de la comparación
```

Las mutaciones puntuales no suponen un gran problema a la hora de comparar, no así las inserciones y remociones.

```
ATAG
 ATA
?XXX
```

Como puede verse, la segunda secuencia perdió su último eslabón (G) haciendo que la comparación falle en todo momento.
Es claro que en este ejemplo bastaría con alinear la segunda secuencia de ADN hacia la izquierda con el fin de maximizar la similitud de las secuencias.

```
ATAG
ATA
===?
```

La generalización de este problema se conoce como *Alineamiento de secuencias de ADN.* [2]
Dos secuencias de ADN serán *similares* si tienen grandes subsecuencias de bases nitrogenadas en ambas cadenas y en la mismas posiciones o bien, en posiciones cercanas de tal manera que se requiere el mínimo alineamiento.
El ejemplo anterior muestra dos secuencias similares donde la subsecuencia compartida es ATA y solo se requirió un único desplazamiento para alinearlas.

En cambio, dos secuencias con pocas y chicas subsecuencias compartidas o que requieren un mayor número de movimientos durante la alineación son menos similares entre sí.

Algoritmo de Smith-Waterman

Existen numerosos algoritmos e implementaciones que automatizan este proceso, alineando y comparando secuencias de ADN y dando el grado de similitud entre estas.
El algoritmo de Smith-Waterman es uno de ellos y se explicará a continuación. [3]

Sean las dos secuencias a comparar S1=a1,a2,.... y S2=b1,b2,…. de largo L1 y L2, se construye una matriz H de dimensiones (L1+1) x (L2+1) y se inicializa la primer fila y la primer columna con ceros (H(0, j) = 0 para todo j y H(i, 0) para toda i). (En esta descripción se usará la nomenclatura de índices de C, esto es, el primer elemento de la matriz esta en (0;0) y no en (1;1)).

Luego de la inicialización, el resto de los valores de la matriz se computan como:

H(i, j) = máximo valor de:

- 0
- H(i-1, j-1) + W(ai, bj)
- H(i-1, j) + W(ai, -)
- H(i, j-1) + W(-, bj)

La función W(ai, bj) determina la bonificación por tener una coincidencia (los elementos ai y bj son iguales) o una penalización por ser diferentes. Los casos especiales W(ai, -) y W(-, bj) permiten cuantificar la penalización por cada desplazamiento que tengamos que hacer (alineación).

Esta función permite asignar pesos distintos según el subtipo (podríamos querer valorar más una coincidencia entre dos Adeninas que entre las otras)
Una simplificación inmediata y de la que se hará uso hace que sólo existan tres pesos posibles

- W(ai, bj) = W_iguales si ai == bj, independientemente del subtipo que sea ai o bj
- W(ai, bj) = W_distintos si ai != bj
- W(ai, -) = W(-, bj) = W_gap

Finalmente, en el último valor de la matriz H(L1, L2) se encuentra el puntaje final, una cuantificación de la similitud entre ambas secuencias.

Ejemplo

S1 = ATAG
S2 = CATA
L1 = L2 = 4

W_iguales = 2
W_distintos = -1
W_gap = -1

Matriz H inicial. (Se le agregó tanto a cada columna como a cada fila su correspondiente subtipo de eslabón):

```
. . A T A G
. 0 0 0 0 0
C 0 ? ? ? ?
A 0 ? ? ? ?
T 0 ? ? ? ?
A 0 ? ? ? ?
```

Veamos el primer caso:

H(1,1) = máximo de
- 0
- H(1-1, 1-1) + W(C, A) = H(0,0) + W(C, A) = 0 + W(C, A) = 0 + W_distintos = -1
- H(1-1, 1) + W_gap = H(0, 1) + W_gap = 0 - 1 = -1
- H(1, 1-1) + W_gap = H(1, 0) + W_gap = 0 - 1 = -1

Es claro que entonces H(1, 1) = 0

```
. . A T A G
. 0 0 0 0 0
C 0 0 ? ? ?
A 0 ? ? ? ?
T 0 ? ? ? ?
A 0 ? ? ? ?
```

Veamos ahora H(2, 1)

H(2,1) = máximo de
- 0
- H(2-1, 1-1) + W(A, A) = H(1,0) + W(A, A) = 0 + W_iguales = 2
- H(2-1, 1) + W_gap = H(1, 1) + W_gap = 0 - 1 = -1
- H(2, 1-1) + W_gap = H(2, 0) + W_gap = 0 - 1 = -1

Es claro que entonces H(2, 1) = 2, resultando en

```
. . A T A G
. 0 0 0 0 0
C 0 0 ? ? ?
```

```
A 0 2 ? ? ?
T 0 ? ? ? ?
A 0 ? ? ? ?
```

Si continuamos con todos los elementos de la matriz, el resultado final es:

```
. . A T A G
. 0 0 0 0 0
C 0 0 0 0 0
A 0 2 1 2 1
T 0 1 4 3 2
A 0 2 3 6 5
```

Finalmente el valor de H(4, 4) = 5 es el puntaje final con que se cuantifica la similitud entre dos secuencias.

Descripción

El presente trabajo consta de implementar un ordenador de secuencias de ADN según su similitud respecto a una secuencia en particular.
Esto es, dada una única secuencia S y dadas las secuencias S1, S2, …. se desea ordenar de mayor a menor las secuencias S1, S2, …. respecto a su similitud con S.
En caso de que dos secuencias Sa y Sb tengan el mismo grado de similitud con S, el orden entre Sa y Sb será el mismo que en el archivo de entrada.

Los argumentos de entrada serán pasados por un archivo con el siguiente formato:

```
<W_iguales>,<W_distintos>,<W_gap>,<Max_Longitud>
<S>
<S1>
<S2>
….
```

Donde <W_iguales>, <W_distintos> y <W_gap> son números enteros con signo que parametrizan el algoritmo de alineación.
El siguiente parámetro es <Max_Longitud> y determina cual es la longitud de la cadena más larga.
Como puede verse la primer línea consta de números separados por comas, donde despues de cada coma puede venir 0 o más espacios en blanco, y finaliza con un salto de línea. Esta primer línea del archivo no superará los 255 bytes ***incluyendo*** al salto de línea.

La segunda línea consta de la secuencia de ADN usada como referencia, una sucesión de caracteres (A, C, G, T) consecutivos que finaliza con un salto de línea.
Las siguientes líneas describen el resto de las secuencias, una por cada línea.
Todas estas tienen como máximo <Max_Longitud> bytes ***sin incluir*** el salto de línea. Las cadenas pueden tener longitudes distintas.

La última secuencia también termina en un salto de línea.

En todos los casos, el byte “salto de línea” hace referencia al byte **‘\n’** y no al los dos bytes **‘\r\n’**.

El formato de los resultados es simplemente las secuencias S1, S2 … ordenadas, una por cada línea. Estos resultados se escribirán en un archivo o bien se escribirán directamente en la salida estándar.

Recomendaciones

- Entender primero el problema evitando resolverlo.
- Leer las libs stdlib, stdio y string.
- Luego de la investigación, proceder a la resolución.

Formato de Línea de Comandos

Para correr el programa y que el resultado se guarde en un archivo

```
./tp entrada.txt salida.txt
```

Si el archivo de salida no se especifica, los resultados se deben escribir en la salida estándar

```
./tp entrada.txt
```

Códigos de Retorno

El programa debera retornar

- 0 en caso de éxito
- 1 si hubo argumentos incorrectos (faltan argumentos o hay de más)
- 2 si el archivo de entrada o el de salida no pudo ser leído/escrito.

Entrada y Salida Estándar

La entrada estándar no es usada. La salida estándar es usada sólo si no se especifica un archivo de salida.

Ejemplos de Ejecución

Sea el archivo de entrada entrada.txt:

```
2, -1, -1, 4
ATAG
CATA
ATAG
ATAC
```

Entonces los parámetros son

- W_iguales = 2
- W_distintos = -1
- W_gap = -1

- Máxima longitud de cadena = 4

La secuencia de referencia es ATAG y las secuencias a comparar y ordenar son CATA, ATAG, ATAC.

Al ejecutar

```
./tp entrada.txt salida.txt
```

El código de retorno debe ser 0 y el contenido del archivo salida.txt debe ser

```
ATAG
CATA
ATAC
```

Esto se debe a que la similitud entre ATAG y ATAG es de 8, mientras que las similitudes entre ATAG y ATAC y entre ATAG y CATA son ambas de 5. En este caso de empate el orden se resuelve usando el mismo orden que se encontró en el archivo de entrada (CATA aparece primero que ATAC en el archivo de entrada).

Restricciones

La siguiente es una lista de restricciones técnicas exigidas:

1. El sistema debe desarrollarse en ISO C (C99)
2. Está prohibido el uso de variables globales.
3. Se podrá usar cualquier biblioteca estándar de C99 pero ninguna externa. [4]
4. No se podrá usar el stack para almacenar toda la matriz H. Se deberá usar el heap.
5. El archivo de entrada puede leerse más de una vez pero solo un número finito, constante y chico de veces (de hecho se podría implementar una solución que requiera solo 2 recorridos completos del archivo, el primero secuencial y el otro de acceso directo.)
6. No se puede cargar todas las secuencias en memoria.

Referencias

[1] ADN Wikipedia: http://es.wikipedia.org/wiki/%C3%81cido_desoxirribonucleico
[2] Alinemiento ADN Wikipedia: http://es.wikipedia.org/wiki/Alineamiento_de_secuencias
[3] Smith-Waterman Wikipedia: http://en.wikipedia.org/wiki/Smith%E2%80%93Waterman_algorithm
[4] C Library: http://www.cplusplus.com/reference/clibrary/