David Santiago Velasco Triana - 2259479 – Grupo Jueves

William Alexander Franco Otero - 2259715– Grupo Martes

Jhojan Stiven Castaño Jejen - 2259476 – Grupo Jueves

3.1. Solución Ingenua

La función *reconstruirCadenaIngenuo* la cual tiene una complejidad de O(4^n) se utiliza para generar todas las posibles combinaciones de una secuencia de DNA de una longitud *n* y luego filtrar estas combinaciones usando a la función oráculo *o.*

La función interna la cual es *generarCombinaciones* se utiliza en este caso para generar todas las combinaciones posibles de la longitud *n* de los caracteres del alfabeto de DNA (‘A’,’C’,G’,’T’).

Esta se puede definir en notación matemática de esta manera donde

- A va a ser el conjunto alfabeto {'A', 'C', 'G', 'T'}.

- n como la longitud de la secuencia

- O como la función del oráculo que toma una secuencia y retorna un bolean

Generar combinaciones generará todas las posibles combinaciones de los elementos A de longitud n el cual se puede representar como A^n,

ReconstruirCadenaIngenuo luego de generar las anteriores combinaciones las filtra usando la función O y devuelve la primera que satisface esta función por lo cual se puede representar como.

reconstruirCadenaIngenuo(n, O) = head({s ∈ A^n | O(s) = true})

En este ejercicio se hace uso de manejo de la recursión, en la función interna generarCombinaciones, también se evidencia mecanismos de encapsulación debido a que generarCombinaciones está definida dentro de la función reconstruirCadenaIngenuo y funciones de alto orden en el caso del filter debido a que usa la función O como argumento

3.2. Solución Mejorada

El método reconstruirCadenaMejorado de complejidad O(n*4^n) en el peor de los casos genera la primera secuencia de longitud n del alfabeto de ADN que cumple con una condición dada por el oráculo o. Utiliza una función auxiliar recursiva GenerarCadenaMejorada que toma un entero k y una secuencia de secuencias de caracteres SC. Si k es mayor que n, devuelve una secuencia vacía. Si no, genera una nueva secuencia newSC agregando cada carácter del alfabeto a cada secuencia en SC y filtrando las que cumplen con la condición del oráculo. Si encuentra una secuencia de longitud n en newSC, la devuelve. Si no, llama a GenerarCadenaMejorada de nuevo con k + 1 y newSC. El método se inicia llamando a GenerarCadenaMejorada con 1 y una secuencia vacía.

Esta se puede definir en notación matemática de esta manera donde

- A va a ser el conjunto alfabeto {'A', 'C', 'G', 'T'}.

- n como la longitud de la secuencia

- O como la función del oráculo que toma una secuencia y retorna un bolean

- k como el contador que incrementa en cada iteración recursiva

La función GenerarCadenaMejorada genera todas las combinaciones elementos de A en longitud k y las filtra usando el oráculo O si no encuentra una secuencia se satisfaga O la devuelve con k + 1

$$\text{GenerarCadenaMejorada}(k, SC) = \{s \in A^k \mid O(s) = \text{true}, \text{length}(s) = n\}$$

La función reconstruirCadenaMejorado inicia el proceso llamando a GenerarCadenaMejorada con k = 1 y SC como una secuencia que contiene una secuencia vacía. Esto puede representarse como:

$$\text{reconstruirCadenaMejorado}(n, O) = \text{GenerarCadenaMejorada}(1, \{\varepsilon\})$$

ε representa la secuencia vacía.

2.3.3 Turbo Solución

La Función reconstruirCadenaTurbo toma dos parámetros los cuales son : n que es la longitud de las combinaciones y O que el oráculo, primero verifica si la cadena es de tamaño uno y la filtra usando la función oráculo en caso que no n no sea igual a 1 entra en la función recursiva generarCadenaTurbo la cual en cada iteración genera secuencias del alfabeto y filtra las secuencias que no cumplan con esta condición (oráculo), luego se una secuencia la cual cumpla con ser igual a la longitud de n si se encuentra retorna esa secuencia que satisface la función. Si no se encuentra la función se llama a sí misma multiplicando k * 2, El conjunto inicial de secuencias se genera mapeando cada carácter del alfabeto a una secuencia. La función generarCadenaTurbo se llama inicialmente con k igual a 1 y el conjunto inicial de secuencias.

Esta se puede definir en notación matemática de esta manera donde

-A es el alfabeto de ADN.

-k es un número entero que representa la longitud de las secuencias a generar.

-SC es una secuencia de secuencias de caracteres.

-O es el Oráculo, una función que toma una secuencia de caracteres y devuelve un booleano.

-s es una secuencia de caracteres.

-length(s) es la longitud de la secuencia s.

Donde generarCadenaTurbo se puede representar

GenerarCadenaTurbo (k, SC) = {s ∈ A^k | O(s) = true, length(s) = n}

k ← 2*k

2.3.4 Turbo Mejorada

La función reconstruirCadenaTurboMejorada en caso de que la longitud n es 1 mapea los caracteres del alfabeto de DNA a una secuencia y encuentra el primero que satisface el oráculo en caso de que no pase devuelve una secuencia vacía.

Si n es mayor que 1 entonces define una función de recursión de cola generarCadenaTurbo la cual genera las secuencias de DNA lo que hace es tomar el entero k y las secuencias de secuencias de caracteres SC y genera una nueva secuencia newSC filtrando SC con la función filtrar y la condición o.

La función filtrar toma una secuencia de secuencia de SC y una entero k, la cual genera una nueva secuencia agregando cara carácter en el alfabeto de DNA a cada secuencia de SC y luego filtra las secuencias que no contienen las posibles subsecuencias de longitud k.

GenerarCadenaTurbo luego intenta encontrar una secuencia en newSC que cumpla con que se igual a la longitud n si esto pasa se devuelve, si no, generarCadenaTurbo se llama recursivamente aumentado k+1 y newSC como la nueva secuencia de secuencias de combinaciones

Paralelizaciones

-Ingenua: En la función reconstruirCadenaIngenuo se logró la paralelización a partir del uso de task, en este se divide el Alfabeto ('A', 'C', 'G', 'T') en 4 partes iguales y luego se generan combinaciones por cada parte en paralelo iniciando cada una con task y ejecutándolas paralelamente con la función parallel lo cual permite que pueda mejorar el rendimiento en sistemas con múltiples núcleos de CPU

-Cadena Mejorada: En la función reconstruirCadenaTurboMejoradaPar se logra paralelizar a través del uso de parallel y task, en este se divide el alfabeto de DNA en 2 partes iguales y se generan combinaciones por cada parte en paralelo, cada tarea de estas crea una combinación de una parte del alfabeto y luego las une todas las combinaciones generadas por tareas parallel, Este enfoque permite que las combinaciones se generen en paralelo, lo que puede mejorar el rendimiento en sistemas con múltiples núcleos de CPU

-Turbo Solución: En el código proporcionado, la función task se utiliza en las funciones reconstruirCadenaIngenuoPar y reconstruirCadenaTurboMejoradaPar para crear tareas paralelas, el alfabeto DNA es dividido en 2 partes alfabeto1 y alfabeto2, luego se crea 2 task los cuales cada uno es responsable de generar una nueva secuencia de DNA de la secuencia existente SC y filtrándola por la función o, la

función parallel es usada para ejecutar las 2 task paralelamente, al final se usa el método join para unir los 2 resultados de los task.

-Turbo Mejorada: Esta función a diferencia de las demás se paraleliza solo con la función parallel de la biblioteca de programación paralela de Scala.

Dentro de generarCadenaTurbo se llama la función filtrar para generar nuevas secuencias de DNA agregando cada carácter del alfabeto de DNA a las secuencias existentes SC esta operación se genera de manera paralela con la operación de encontrar una secuencia de la longitud deseada n en las secuencias existentes SC, esto gracias a que parallel nos permite ejecutar estas 2 operaciones de manera concurrente, esta función parallel devuelve una tupla con los resultados de las 2 operaciones, el primero de la tupla newSC y el segundo el resultado de la operación find

BENCHMARKS

Tamaño Cadena	ReconstruirCadenaIngenuo	ReconstruitCadenaIngenuoPar	aceleracion
2	97.4	296.70	0.328
4	287.5	1315.39	0.218
6	1209.0	3383.9	0.357
8	39930.9	33690.0	1,185
10	986752.7	604590.2	1.63

Tamaño Cadena	ReconstruirCadenaMejorada	ReconstruitCadenaMejoradaPar	aceleracion
4	119.6	255.6	0.467
8	397.9	698.1	0.569
16	528.0	1037.4	0.508
32	3731.39	4645.5	0.803
64	31942.39	21400.0	1,492
128	280417.0	151372.3	1,852
256	2271786.9	1205323.59	1,884
512	1.32E+23	7632901.39	1,725

Tamaño Cadena	ReconstruirCadenaTurbo	ReconstruitCadenaTurboPar	aceleracion
4	135.2	363.3	0.372
8	204.1	442.0	0.461
16	519.0	1061.39	0.488
32	4258.20	5488.1	0.775
64	34993.7	25237.7	1,386
128	284335.5	226113.0	1,257
256	2256449.9	1688330.5	1,336
512	1.79E+03	1,376	1,298

Tamaño Cadena	ReconstruirCadenaTurboMejorada	ReconstruitCadenaTurboMejoradaPar	aceleracion
4	279,601	184.70	1,513
8	266,498	231.7	1,150
16	527.0	569.2	0.925
32	2552.6	2874.89	0.887
64	12044.79	25237.7	1,004
128	88909.79	92683.5	0.959
256	1154612.1	755866.70	1,527
512	7031472.0	6131730.30	1,298