Reporte Segundo Proyecto: Programación Dinámica, Problema de la Mochina y Problema de Alineamiento de Secuencias

Brandon Josué Ledezma Fernández
Carné: 2018185574
Curso de Investigación de Operaciones
Escuela de Ingeniería en Computación, Instituto Tecnológico de Costa Rica

Abstract

El presente informe contiene información relevante acerca de los problemas de programación dinámica: mochila y alineamiento de secuencias asignados para el segundo proyecto del curso de Investigación de Operaciones, en el cual se presenta la especificación correspondiente a cada problema los cueles contienen una solución utilizando tanto fuerza bruta, como programación dinámica, los parámetros utilizados para cada problema, así como resultados obtenidos a partir de una serie de experimentos realizados por las soluciones creadas, en las cuelen se analizaron los resultados obtenidos.

I. Especificación del problema

- **Problema de la mochila:** También conocido como el problema del contenedor, trata sobre el caso en que teniendo una mochila que posee un peso de **W** fijo, en el cual se deben elegir de un total de **N** elementos disponibles, los cuales tienen un valor **b** y una disponibilidad de **c**, entonces se debe poder elegir los elementos correctos, para beneficiar de la mejor forma posible el costo y peso, sin excederse de los limites establecidos.
- **Problema de Alineamiento de Secuencias:** El estudio de las secuencias biológicas a mejorado con la inclusión de los sistemas de computación, el algoritmo de Saul Needleman y Christian Wunsch es el utilizado para este proyecto, el cual consiste en un sistema de scoring, en el que se ingresan ”Gaps”para ajustar las hileras, el sistema de scoring trata: si los dos caracteres en una posición i son iguales se suma 1, si son distintos se resta 1, mientras que el caso de que se encuentre un Gap(_) se resta 2, para finalmente obtener el resultado de comparar dos secuencias.

II. Parámetros utilizados para los problemas:

Existen dos programas una para solucionar problemas que lee archivos que contiene los datos del problema y otro que los genera y crea archivos en los cuales se guarda el resultado obtenido.

Los problemas tienen dos formas de solucionarse mediante fuerza bruta y utilizando programación dinámica, a continuación se específica los parámetros para cada caso:

II-A. Problema de la mochila:

- Solucionador:

  - Fuerza Bruta:

```
python3 solver.py 1 1 archivo
```

  donde archivo corresponde al nombre del archivo que contiene el problema.

  - Programación Dinámica:

```
python3 solver.py 1 2 archivo
```

  donde archivo corresponde al nombre del archivo que contiene el problema.

  - Formato archivo entrada:

```
Peso máximo soportado por el contenedor
elemento i (peso, beneficio, cantidad)
elemento n (peso, beneficio, cantidad)
```

  Un ejemplo es el siguiente:

```
50
5,20,4
15,50,3
10,60,3
```

- Generador:

  - Generación:

```
python3 generator.py 1 1 W N minPeso maxPeso minBeneficio
maxBeneficio minCantidad maxCantidad
```

donde los parámetros corresponden a valores enteros, de los cuales se elegirá un valor entre un rango del mínimo y máximo establecidos.

- Formato archivo resultado:

```
Peso máximo soportado por el contenedor
elemento i (peso, beneficio, cantidad)
elemento n (peso, beneficio, cantidad)
```

Un ejemplo es el siguiente:

```
50
5,20,4
15,50,3
10,60,3
```

II-B. Problema de Alineamiento de Secuencia:

- Solucionador:

  - Fuerza Bruta:

```
python3 solver.py 2 1 archivo
```

  donde archivo corresponde al nombre del archivo que contiene el problema.

  - Programación Dinámica:

```
python3 solver.py 2 2 archivo
```

  donde archivo corresponde al nombre del archivo que contiene el problema.

  - Formato archivo entrada:

```
valoriguales,valordistintos,valorgap
hilera1
```

```
hilera2
```

donde los valores corresponde a valores enteros, mientras que las hileras a cadenas de texto (strings). Un ejemplo es el siguiente:

```
1,-1,-2
ATTGTGATCC
TTGCATCGGC
```

- Generador:

  - Generación:

```
python3 generator.py 2 largoH1 largoH2
```

donde largo1 corresponde a la cantidad de caracteres de la primera hilera y largo2 al de la segunda, donde aleatoriamente se ira eligiendo entre los valores: A, T, C y G para formar las hileras.

- Formato archivo resultado:

```
valoriguales,valordistintos,valorgap
hilera1
hilera2
```

donde los valores corresponde a valores enteros, mientras que las hileras a cadenas de texto (strings). Un ejemplo es el siguiente:

```
1,-1,-2
ATTGTGATCC
TTGCATCGGC
```

III. Resultados de ejecución

A continuación se presentan los resultados de ejecutar los distintos problemas con distintos datos, es decir en el caso de la mochila con **N** elementos y el caso de ordenamiento de hileras de tamaño **N** y **M** correspondientemente.

III-A. Problema de la mochila

Se utilizo como ejemplos ya establecidos los presentados por el profesor en el documento de la asignación del proyecto, los demás son generados a partir del generador de problemas:

Fuerza Bruta	Tiempo Ejecución	Programación Dinámica	Tiempo Ejecución	Diferencia
50 5,20,4 15,50,3 10,60,3	0:00:00.000700	50 5,20,4 15,50,3 10,60,3	0:00:00.000173	527 s
100 53,46,1 59,37,1 47,37,1	0:00:00.000049	100 53,46,1 59,37,1 47,37,1	0:00:00.000106	57 s
200 57,31,5 49,37,4 48,43,3 40,39,2 41,41,4 59,29,2	0:00:00.603277	200 57,31,5 49,37,4 48,43,3 40,39,2 41,41,4 59,29,2	0:00:00.001129	602148 s
200 42,39,2 52,48,2 44,36,2 44,39,2 55,53,2 40,54,2 59,47,2 50,34,2 42,33,2 45,32,2 48,33,2 57,53,2 44,43,2 50,52,2 58,32,2	0:13:26.126200	200 42,39,2 52,48,2 44,36,2 44,39,2 55,53,2 40,54,2 59,47,2 50,34,2 42,33,2 45,32,2 48,33,2 57,53,2 44,43,2 50,52,2 58,32,2	0:00:00.001890	0.2239234194 h

TABLE I

TABLA DE RESULTADOS DE TIEMPO DE EJECUCIÓN DE DISTINTOS EJEMPLO TANTO EN FUERZA BRUTA COMO PROGRAMACIÓN DINÁMICA. ELABORACIÓN PROPIA.

III-B. Problema Alineamiento de Secuencias

De igual forma que con los ejemplos utilizados en la sección anterior, se utiliza los ejemplos presentados por el profesor, así como otros obtenidos del generador de problemas.

Fuerza Bruta	Tiempo Ejecución	Programación Dinámica	Tiempo Ejecución	Diferencia
ATTGTGATCC TTGCATCGGC	+24:00:00.000000	ATTGTGATCC TTGCATCGGC	0:00:00.000466	Desconocida
GTG CAT	0:00:00.034744	GTG CAT	0:00:00.000126	34618 s
ATTG CATC	0:01:00.954633	ATTG CATC	0:00:00.000166	60.954467 s
ATGAC TTGGC	+2:00:00.000000	ATGAC TTGGC	0:00:00.000221	+2 h

TABLE II

TABLA DE RESULTADOS DE TIEMPO DE EJECUCIÓN DE DISTINTOS EJEMPLO TANTO EN FUERZA BRUTA COMO PROGRAMACIÓN DINÁMICA. ELABORACIÓN PROPIA.

IV. COMPLEJIDAD TEMPORAL

IV-A. Problema mochila

- Fuerza bruta:

  Para realizar este algoritmo es necesario decidir si incluir o no un elemento en la mochila, lo cual va creando un tipo de grafo de decisiones el cual tiene un costo computacional de:

  $$O(n) = 2^n$$

  para lo cual también hay que tomar en cuenta el peso soportado por la mochila.
- Programación Dinámica: Al utilizar programación dinámica la complejidad disminuye, a tal punto que, disminuye considerablemente:

  $$O(n) = n \cdot w$$

IV-B. Problema Alineamiento de secuencias

- Fuerza bruta:

  Para realizar esta solución lo que se realiza es comprobar todas las posibles soluciones añadiendo a las hileras la misma cantidad de Gaps que el largo correspondiente de cada hilera, para lo cual es necesario comparar la primera hilera sin Gaps con todas las posibles combinaciones de la

hilera 2 y Gaps y así avanzar realizando todas las posibles permutaciones por lo tanto el tiempo computacional se puede calcular a partir de:

$$O(n) = n \cdot m \cdot 2^n \cdot 2^m$$

asumiendo que n es similar a m seria:

$$O(n) = n^2 \cdot 4^n$$

- Programación Dinámica:

  Al igual que el caso de la mochila, la complejidad computacional disminuye considerablemente al punto de ser:

$$O(n) = n \cdot m$$

V. Análisis y Conclusiones

La comparación entre optar por usar algoritmos de fuerza bruta y programación dinámica, no debe ser una difícil decisión para los programadores, la implementación de un algoritmo de programación dinámica es superiormente efectiva en comparación a una implementación por fuerza bruta. Desde cursos anteriores como: Análisis de Algoritmos y Lenguajes de Programación, nos han inculcado como estudiantes en la búsqueda de mejorar los algoritmos presentes en los sistemas computacionales.

En el caso de este proyecto se refleja mucho esta diferencia de implementaciones, donde usar un problema con las mismas condiciones da menores tiempos a la hora de usar la PD, sin embargo, cabe recalcar que todas estas implementaciones se realizan con el uso de un único hilo, muy probablemente si se utiliza múltihilos para solucionar estos problemas el tiempo de ejecución disminuya, ya que las operaciones se dividen y se resuelven por separado.

En cuanto a las implementaciones realizadas, cabe recalcar algunas curiosidades:

- El caso de la mochila utilizando fuerza bruta, su tiempo es menor en los primeros casos.
- Una mochila con una cantidad de elementos de 25 y una de 26 difiere exponencialmente, en diferencias a sus anteriores.
- El problema de alineamiento de secuencias con un largo de N, M es superiormente más complejo que con hileras de largo N, M -1 como muestra de que un pequeño cambio en esta clase de problemas representa una gran diferencia.

Finalmente, mencionar que queda comprobado que la programación dinámica si representa una mejora de los problemas complejos, reduciendo el tiempo de ejecución, para lo cual se cumple con el principio de optimalidad, esto gracias a que el problema original es dividido en subproblemas.