**Universidad de los Andes**

**Departamento de Ingeniería de Sistemas y Computación**

**Diseño y Análisis de Algoritmos**

**2015-19**

**Jorge de los Ríos (201011164)  
Laura Camila Mojica (201212513)**

**PROBLEMA A**

El problema consiste en encontrar un subconjunto del conjunto de canciones d1…dN con el cual se pueda lograr el máximo tiempo de grabación de un disco compacto que posee capacidad S. Para la implementación de la solución, se nos otorgó una solución ingenua y se proponía a mejorar su complejidad mediante dos métodos: Memoización y Programación dinámica. Estos dos métodos fueron implementados con complejidades temporales y espaciales menores.

En el momento de analizar la complejidad de la solución ingenua otorgada con el problema, se puede observas que lo que se pretende es realizar un árbol de recurrencias, donde en se van a tener N niveles. Esto nos da una complejidad temporal de la solución ingenua de orden O(2n). Para los métodos de solución planteados, se desarrollaron soluciones las cuales tuvieran como propósito mejorar la complejidad de la solución ingenua. Para esto, se implementó un algoritmo de memoización, en el cual se va calculando un máximo a la vez y se agrega en una matriz, en la cual van a quedar estos datos para ser consultados en el futuro, así como ir almacenando un máximo general. Estas operaciones van a tener una complejidad temporal de O(N\*S). Para la solución con programación dinámica, lo que se realiza es inicializar una matriz y rellenarla de ceros para posteriormente ir asignando a cala posición de la misma un valor máximo hasta el momento. Al final del algoritmo, el máximo general se encontrará en la posición N, S. Esta solución tiene una complejidad temporal y espacial de O(N\*S).

Como se puede observar en la siguiente gráfica, y si consideramos que la complejidad O(N\*S) se puede comportar como O(n2), en la gráfica se muestra que los algoritmos implementados son mejores para n > 4. Como no siempre va a pasar que S = N, el límite para que estos sean mejor aumentará, pero como la complejidad de la solución ingenia es exponencial, siempre llegará a ser peor que una complejidad pseudopolinomial.

Para finalizar, se puede decir que el problema se comporta de igual manera que un problema de mochila(knapsack problema), el cual es un problema que se resuelve en tiempo pseudopolinomial con complejidad O(NS) y que es un problema NP-Completo.

Con programación dinámica y con memoización, se decide usar la matriz como estructura de almacenamiento de los resultados parciales del avance hacia la solución del problema. El algoritmo desarrollado con programación dinámica tiene como fuerte que tiene una complejidad espacial menor al algoritmo memoizado, lo cual hace que su desempeño sea mejor cuando se corre en consola. Los puntos débiles de los algoritmos es que van a depender del tamaño de la matriz, siendo este N\*S. Igual, siempre y cuando se alcance el límite, este va a ser un mejor algoritmo al propuesto por la solución ingenua.