


ALGORITMOS GENÉTICOS

“El problema de la Mochila”

Prof. Ing. Daniela E. Díaz






A pesar del desarrollo científico de los últimos años aún quedan muchos problemas reales de optimización con una gran importancia práctica que no han sido resueltos.

Dentro de este grupo están los que se caracterizan porque sus espacios solución están formados por subconjuntos de números naturales y por dicha razón se los llama problemas de optimización combinatoria.



La **optimización combinatoria** es una rama de la optimización en matemáticas aplicadas y en ciencias de la computación, relacionada a la investigación de operaciones, teoría de algoritmos y teoría de la complejidad computacional.

También está relacionada con otros campos, como la inteligencia artificial e ingeniería de software.



Los algoritmos de **optimización combinatoria** resuelven instancias de problemas que se creen ser difíciles en general, explorando el espacio de soluciones (usualmente grande) para estas instancias.

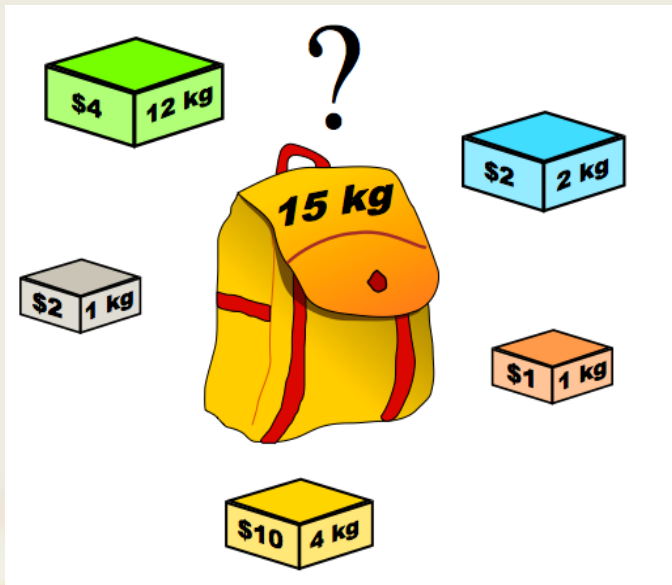
Los algoritmos de **optimización combinatoria** logran esto reduciendo el tamaño efectivo del espacio, y explorando el espacio de búsqueda eficientemente.

Los algoritmos de **optimización combinatoria** a menudo son implementados en lenguajes imperativos (tipo de lenguajes donde las instrucciones se ejecutan unas tras otras, de manera secuencial, salvo cuando se encuentran estructuras de control condicionales o bucles. Hay declaración de variables, tipos y procedimientos, aunque esto varía notablemente en función del lenguaje utilizado, pues hay cuales exigen las declaraciones mientras que otros permiten que esos elementos no sean declarados) como C y C++, en lenguajes de programación lógicos tales como Prolog.





Quizás los dos problemas de optimización combinatorios más famosos son el de la mochila y el del viajante (TSP Travelling salesman problem).





Problema de la mochila

Consiste en elegir, de entre un conjunto de n elementos, (cada uno con un valor $\$i$, y un volumen V_i), aquellos que puedan ser cargados en una mochila de volumen V de manera que el valor obtenido sea máximo.



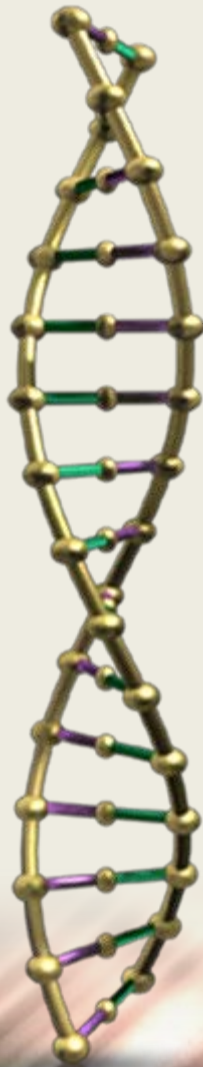
Utilizando una computadora, resolver el siguiente problema:

Cuáles son los elementos de la lista siguiente que cargaremos en una mochila de 4200 cm³ de manera que su valor en \$ sea máximo.



Objeto	Volumen (cm ³ .)	Valor (\$)
1	150	20
2	325	40
3	600	50
4	805	36
5	430	25
6	1200	64
7	770	54
8	60	18
9	930	46
10	353	28

Volumen máximo
soportado por la
mochila: 4200 cm³.



Para su solución, utilizar un procedimiento exhaustivo que consiste en evaluar para cada subconjunto de elementos el valor correspondiente y, posteriormente, clasificando los subconjuntos por su valor de mayor a menor, encontrar cuál es el subconjunto solución.

Planteando el problema de la mochila en la forma anterior resultaría de fácil ejecución y la **solución encontrada tendría la importancia de ser única.**


El número de elementos N está asociado a la cantidad de elementos que posee el espacio solución.

Si el conjunto es de N elementos, el conjunto solución tendrá la cantidad de elementos del conjunto de Partes, es decir, 2^N .

Por lo tanto, si una computadora pudiese, en un segundo, generar un millón de estos subconjuntos y evaluar su valor, necesitaríamos poco más de un segundo para hallar la solución óptima del problema para 20 elementos (se generan $2^{20}=1.048.580$ subconjuntos o soluciones).


Para $n=40$ ya nos llevaría 2 semanas, y para $n=60$ se necesitarían 365 siglos. Entonces, **el problema es sencillo si la cantidad de elementos N es reducida.** En cambio, a medida que aumenta el valor de N , el tiempo aumenta rápidamente y, para valores altos de N , no hay forma de resolverlo exhaustivamente.





Pero, cuando se presenta el problema es necesario resolverlo inmediatamente. Para esto, tenemos una manera de resolución bastante buena que no nos da, en la mayoría de los casos, la solución óptima sino una buena solución y que está basada en el sentido común. Este método se denomina ***heurístico***.

Existen, entonces, métodos heurísticos que proporcionan soluciones factibles (que satisfacen las restricciones del problema), que, aunque no optimicen la función objetivo, se acercan al valor óptimo en un tiempo de cálculo razonable.



Una clase de algoritmos heurísticos son los **métodos constructivos**, que consisten en ir agregando componentes individuales a la solución hasta que se obtiene una solución factible.

Un ejemplo de estos algoritmos heurísticos son los algoritmos **greedy** (golosos o devoradores). Estos algoritmos van construyendo paso a paso la solución, buscando el máximo beneficio en cada paso. En el problema de la mochila, debemos ir escogiendo los elementos que aporten el mayor valor en proporción a su peso ($\$i / V_i$). Una vez tabulado de mayor a menor comenzamos a cargar la mochila, siguiendo este ordenamiento, hasta que no entren más elementos. Si en ese momento, pudiese entrar uno posterior en lugar del siguiente, habría que agregarlo.

Ejercicios:

1. Resolver el problema de la Mochila utilizando una búsqueda exhaustiva.
2. Resolver el ejercicio anterior usando el algoritmo greedy y comentar su similitud o no con el exhaustivo.
3. En algunas ocasiones planteamos el problema de la mochila teniendo en cuenta los pesos en lugar del volumen. Luego, dados 3 elementos, cuyos pesos son: 1800 grs., 600 grs. Y 1200 grs. y cuyos valores son: \$72, \$36 y \$60 respectivamente, y con una mochila que puede soportar hasta 3000 grs. se pide:
 - A) Hallar una solución utilizando un algoritmo goloso y exhaustivo.
 - B) Analizar dicha solución respecto a su grado de optimización y elaborar las conclusiones que considere adecuadas.





¿Preguntas?

