Introducción a la Optimización Combinatoria

Jose Antonio Lozano

Intelligent Systems Group
Departamento de Ciencias de la Computación e Inteligencia Artificial
Universidad del País Vasco–Euskal Herriko Unibertsitatea

UPV - EHU

Organización del tema

1. Introducción
2. Problemas de optimización combinatoria
3. Complejidad de problemas de optimización
4. Métodos Clásicos vs Métodos Heurísticos

UPV - EHU

Organización del tema

1. Introducción
2. Problemas de optimización combinatoria
3. Complejidad de problemas de optimización
4. Métodos Clásicos vs Métodos Heurísticos

El problema a resolver

El caso general

$$\begin{array}{lll} \textit{min(max)} & f(\boldsymbol{x}) & \boldsymbol{x} \in \Omega \\ \textit{sujeto a} & & \\ & g_i(\boldsymbol{x}) \geq b_i & i = 1, 2, \ldots, m \\ & h_j(\boldsymbol{x}) = c_j & j = 1, 2, \ldots, s \end{array}$$

- En muchas ocasiones se quiere conocer el punto $\boldsymbol{x}$ donde se alcanza el óptimo (arg max $f(\boldsymbol{x})$)

Casos conocidos

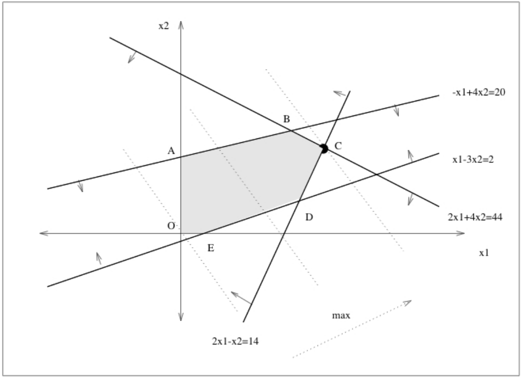
Programación lineal (Investigación Operativa)

$$
\begin{array}{rll}
\textit{min(max)} & \sum_{i=1}^{n} a_i.x_i & \\
\textit{sujeto a} & & \\
& \sum_{i=1}^{n} c_{ji}.x_i \geq b_j & j = 1, 2, \ldots, m \\
& x_i \in \mathbb{R} & i = 1, 2, \ldots, n
\end{array}
$$

- Método de resolución: Algoritmo Simplex

UPV - EHU

Resolución geométrica



UPV - EHU

Casos conocidos

Programación 0-1 (I.O.)

$$
\begin{array}{rll}
min(max) & \sum_{i=1}^{n} a_i.x_i & \\
sujeto\ a & & \\
& \sum_{i=1}^{n} c_{ji}.x_i \geq b_j & j = 1, 2, \ldots, m \\
& x_i = 0, 1 & i = 1, 2, \ldots, n
\end{array}
$$

- Método de resolución: Ramificar y acotar

UPV - EHU

Casos conocidos

Programación entera

$$
\begin{array}{rll}
min(max) & \sum_{i=1}^{n} a_i.x_i & \\
sujeto\ a & & \\
& \sum_{i=1}^{n} c_{ji}.x_i \geq b_j & j = 1, 2, \ldots, m \\
& x_i = 1, \ldots, k_i & i = 1, 2, \ldots, n
\end{array}
$$

- Método de resolución: Ramificar y acotar

UPV - EHU

Casos conocidos (I.O.)

Programación Cuadrática

$$\begin{array}{rl} min(max) & \sum_{i=1}^{n} \sum_{k=1}^{n} a_{ik}.x_i.x_k \\ sujeto\ a & \\ & \sum_{i=1}^{n} c_{ji}.x_i \geq b_j \qquad j = 1, 2, \ldots, m \\ & x_i \in I\!R \qquad i = 1, 2, \ldots, n \end{array}$$

- Método de resolución: Algoritmos de programación cuadrática

UPV - EHU

Casos conocidos (I.O.)

Programación No lineal

$$
\begin{array}{rl}
\textit{min(max)} & f(\boldsymbol{x}) \\
\textit{sujeto a} & \\
& g_i(\boldsymbol{x}) \geq b_i \quad i = 1, 2, \ldots, m \\
& x_i \in \mathbb{R} \qquad i = 1, 2, \ldots, n
\end{array}
$$

- Método de resolución: Algoritmos numéricos (Fletcher-Powell)

UPV - EHU

Tipos de problemas de optimización (clasificación alternativa)

Optimización combinatoria

- El cojunto de soluciones Ω es finito o numerable

Optimización en el mundo continuo

- El espacio de búsqueda está incluido en $\mathbb{R}^n$ y no es ni finito ni numerable

UPV - EHU

Tipos de problemas de optimización (clasificación alternativa)

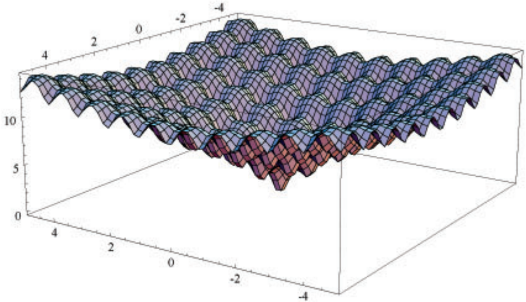
Optimización combinatoria

- El cojunto de soluciones Ω es finito o numerable

Optimización en el mundo continuo

- El espacio de búsqueda está incluido en $\mathbb{R}^n$ y no es ni finito ni numerable

Ejemplo de función continua



$$h(\boldsymbol{x}) = -c_1 \cdot \exp\left(-c_2 \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(c_3 \cdot x_i)\right) + c_1 + e$$

UPV - EHU

Organización del tema

1 Introducción
2 Problemas de optimización combinatoria
3 Complejidad de problemas de optimización
4 Métodos Clásicos vs Métodos Heurísticos

UPV - EHU

El problema del agente viajero (TSP)

Enunciado

- Planteamiento: se trata de establecer una ruta que pasando por n ciudades una sola vez y terminando en la ciudad inicial recorra la mínima distancia
- Datos: número de ciudades n, matriz de distancias entre las ciudades $D = (d_{ij})_{i,j=1,\ldots,n}$
- Formalización:
  - ¿Cómo podemos representar una solución?
  - ¿Cómo podemos representar el espacio de búsqueda? ¿qué tamaño tiene dicho espacio?
  - ¿Cómo podemos representar la función objetivo?

El problema del agente viajero (TSP)

Enunciado

- Planteamiento: se trata de establecer una ruta que pasando por n ciudades una sola vez y terminando en la ciudad inicial recorra la mínima distancia
- Datos: número de ciudades n, matriz de distancias entre las ciudades $D = (d_{ij})_{i,j=1,\ldots,n}$
- Formalización:
  - ¿Cómo podemos representar una solución?
  - ¿Cómo podemos representar el espacio de búsqueda? ¿qué tamaño tiene dicho espacio?
  - ¿Cómo podemos representar la función objetivo?

El problema del agente viajero (TSP)

Enunciado

- Planteamiento: se trata de establecer una ruta que pasando por n ciudades una sola vez y terminando en la ciudad inicial recorra la mínima distancia
- Datos: número de ciudades n, matriz de distancias entre las ciudades $D = (d_{ij})_{i,j=1,\ldots,n}$
- Formalización:
  - ¿Cómo podemos representar una solución?
  - ¿Cómo podemos representar el espacio de búsqueda? ¿qué tamaño tiene dicho espacio?
  - ¿Cómo podemos representar la función objetivo?

El problema del agente viajero (TSP)

Enunciado

- Planteamiento: se trata de establecer una ruta que pasando por n ciudades una sola vez y terminando en la ciudad inicial recorra la mínima distancia
- Datos: número de ciudades n, matriz de distancias entre las ciudades $D = (d_{ij})_{i,j=1,\ldots,n}$
- Formalización:
  - ¿Cómo podemos representar una solución?
  - ¿Cómo podemos representar el espacio de búsqueda? ¿qué tamaño tiene dicho espacio?
  - ¿Cómo podemos representar la función objetivo?

UPV - EHU

El problema del agente viajero (TSP)

Enunciado

- Planteamiento: se trata de establecer una ruta que pasando por n ciudades una sola vez y terminando en la ciudad inicial recorra la mínima distancia
- Datos: número de ciudades n, matriz de distancias entre las ciudades $D = (d_{ij})_{i,j=1,\ldots,n}$
- Formalización:
  - ¿Cómo podemos representar una solución?
  - ¿Cómo podemos representar el espacio de búsqueda? ¿qué tamaño tiene dicho espacio?
  - ¿Cómo podemos representar la función objetivo?

El problema del agente viajero (TSP)

Enunciado

- Planteamiento: se trata de establecer una ruta que pasando por n ciudades una sola vez y terminando en la ciudad inicial recorra la mínima distancia
- Datos: número de ciudades n, matriz de distancias entre las ciudades $D = (d_{ij})_{i,j=1,\ldots,n}$
- Formalización:
  - ¿Cómo podemos representar una solución?
  - ¿Cómo podemos representar el espacio de búsqueda? ¿qué tamaño tiene dicho espacio?
  - ¿Cómo podemos representar la función objetivo?

UPV - EHU

Formalización del TSP

Representación de una solución

Formalización del TSP

Representación de una solución

- Solución: permutación de las ciudades

Formalización del TSP

Representación de una solución

- Solución: permutación de las ciudades

Espacio de búsqueda

Formalización del TSP

Representación de una solución

- Solución: permutación de las ciudades

Espacio de búsqueda

$$\Omega = \{(\pi_1\ \pi_2 \ldots \pi_n) \mid \pi_i \in \{1, 2, \ldots, n\} \text{ y } \pi_i \neq \pi_j\ \forall i \neq j\}$$

Formalización del TSP

Representación de una solución

- Solución: permutación de las ciudades

Espacio de búsqueda

$$\Omega = \{(\pi_1\ \pi_2 \ldots \pi_n) \mid \pi_i \in \{1, 2, \ldots, n\}\ \text{ y }\ \pi_i \neq \pi_j\ \forall i \neq j\}$$

- ¿Tamaño del espacio de búsqueda?
- $(1\ 2\ 3\ 4\ 5) = (2\ 3\ 4\ 5\ 1) = (5\ 4\ 3\ 2\ 1)$

UPV - EHU

Formalización del TSP

Representación de una solución

- Solución: permutación de las ciudades

Espacio de búsqueda

$$\Omega = \{(\pi_1\ \pi_2 \ldots \pi_n) \mid \pi_i \in \{1, 2, \ldots, n\} \text{ y } \pi_i \neq \pi_j\ \forall i \neq j\}$$

- ¿Tamaño del espacio de búsqueda?
- $(1\ 2\ 3\ 4\ 5) = (2\ 3\ 4\ 5\ 1) = (5\ 4\ 3\ 2\ 1)$
- $\frac{(n-1)!}{2}$

UPV - EHU

Formalización del TSP

Función objetivo

Formalización del TSP

Función objetivo

$$F(\pi_1\pi_2\dots\pi_n) = \sum_{i=2}^{n} d_{\pi_{i-1}\pi_i} + d_{\pi_n\pi_1}$$

Problema de la mochila

Planteamiento

- Se dispone de un conjunto de objetos cada uno de ellos con un peso y un valor asociado. Se trata de introducir un subconjunto de objetos en una mochila, de forma que: (i) se maximiza la suma de los valores de los objetos y (ii) la suma de los pesos no sobrepasa una cantidad máxima

Problema de la mochila

Planteamiento

- Se dispone de un conjunto de objetos cada uno de ellos con un peso y un valor asociado. Se trata de introducir un subconjunto de objetos en una mochila, de forma que: (i) se maximiza la suma de los valores de los objetos y (ii) la suma de los pesos no sobrepasa una cantidad máxima

Datos

Problema de la mochila

Planteamiento

- Se dispone de un conjunto de objetos cada uno de ellos con un peso y un valor asociado. Se trata de introducir un subconjunto de objetos en una mochila, de forma que: (i) se maximiza la suma de los valores de los objetos y (ii) la suma de los pesos no sobrepasa una cantidad máxima

Datos

- Número de objetos: n

UPV - EHU

Problema de la mochila

Planteamiento

- Se dispone de un conjunto de objetos cada uno de ellos con un peso y un valor asociado. Se trata de introducir un subconjunto de objetos en una mochila, de forma que: (i) se maximiza la suma de los valores de los objetos y (ii) la suma de los pesos no sobrepasa una cantidad máxima

Datos

- Número de objetos: n
- Pesos para los objetos: $\{p_1, \ldots, p_n\}$

UPV - EHU

Problema de la mochila

Planteamiento

- Se dispone de un conjunto de objetos cada uno de ellos con un peso y un valor asociado. Se trata de introducir un subconjunto de objetos en una mochila, de forma que: (i) se maximiza la suma de los valores de los objetos y (ii) la suma de los pesos no sobrepasa una cantidad máxima

Datos

- Número de objetos: n
- Pesos para los objetos: $\{p_1, \ldots, p_n\}$
- Valores de los objetos: $\{v_1, \ldots, v_n\}$

Problema de la mochila

Planteamiento

- Se dispone de un conjunto de objetos cada uno de ellos con un peso y un valor asociado. Se trata de introducir un subconjunto de objetos en una mochila, de forma que: (i) se maximiza la suma de los valores de los objetos y (ii) la suma de los pesos no sobrepasa una cantidad máxima

Datos

- Número de objetos: n
- Pesos para los objetos: $\{p_1, \ldots, p_n\}$
- Valores de los objetos: $\{v_1, \ldots, v_n\}$
- Capacidad de la mochila: C

Formalización del problema de la mochila

Representación de una solución

Formalización del problema de la mochila

Representación de una solución

- Solución: vector binario $(x_1, \ldots, x_n)$ de tamaño n tal que:

$$x_i = \begin{cases} 1 & \text{si el objeto } i \text{ se introduce en la mochila} \\ 0 & \text{en caso contrario} \end{cases}$$

Formalización del problema de la mochila

Representación de una solución

- Solución: vector binario $(x_1, \ldots, x_n)$ de tamaño n tal que:

$$x_i = \begin{cases} 1 & \text{si el objeto } i \text{ se introduce en la mochila} \\ 0 & \text{en caso contrario} \end{cases}$$

Espacio de búqueda

UPV - EHU

Formalización del problema de la mochila

Representación de una solución

- Solución: vector binario $(x_1, \ldots, x_n)$ de tamaño n tal que:

$$x_i = \begin{cases} 1 & \text{si el objeto } i \text{ se introduce en la mochila} \\ 0 & \text{en caso contrario} \end{cases}$$

Espacio de búqueda

$$\Omega = \{(x_1, \ldots, x_n) | x_i \in \{0, 1\} \text{ y } \sum_{i=1}^{n} p_i \cdot x_i \leq C\}$$

UPV - EHU

Formalización del problema de la mochila

Función objetivo

$$F((x_1,\ldots,x_n)) = \sum_{i=1}^{n} v_i \cdot x_i \quad \text{con} \quad (x_1,\ldots,x_n) \in \Omega$$

Problema de la partición de un grafo

Enunciado

Consideremos un grafo no dirigido en el que cada arista tiene asociado un peso y el número de vértices es par. Se trata de dividir el conjunto de vértices en dos subconjuntos iguales, de forma que se minimice la suma de los pesos asociados a aristas que unen vértices de diferentes conjuntos.

Problema de planificación (scheduling)

Enunciado

Supongamos que debemos realizar un conjunto de trabajos y que cada trabajo requiere de la realización de un conjunto ordenado de m operaciones. Cada una de estas operaciones se debe realizar en una de las m máquinas de las que se dispone y posee un tiempo de proceso concreto.

Problema de planificación (scheduling)

Restricciones

- Una operación debe realizarse de forma ininterrumpida en una máquina
- En cada instante de tiempo una máquina únicamente puede procesar una operación
- En cada trabajo una operación no puede comenzar a procesarse hasta que las operaciones previas hayan terminado

Problema de planificación (scheduling)

Función a optimizar

- La función a optimizar más común consiste en procesar todos los trabajos lo más rápido posible, es decir, minimizar el tiempo de terminación del último trabajo.

Problema de la ruta de vehículos

Enunciado

Un conjunto de vehículos debe dar servicio a un conjunto de clientes. Cada cliente realiza una demanda concreta y cada vehículo dispone de una capacidad limitada. Se trata de satisfacer la demanda de manera que los camiones recorran la cantidad mínima de distancia. Se supone que todos los vehículos parten del mismo lugar y que la distancia que pueden recorrer está limitada

Organización del tema

1. Introducción
2. Problemas de optimización combinatoria
3. Complejidad de problemas de optimización
4. Métodos Clásicos vs Métodos Heurísticos

UPV - EHU

¿Cuál es la complejidad de un problema de optimización combinatorial?

Nociones de complejidad computacional

- ¿Cómo medir la complejidad?
- Instancias de un problema
- El modelo de la máquina de Turing
- El algoritmo más sencillo: enumeración
- El problema de decisión y el problema de optimización

UPV - EHU

Problemas P y problemas NP

Complejidad exponencial

- Problemas Polinomiales
- Problemas No Polinomiales
- Ejemplos: TSP, Max-SAT, Ruta de vehículos, ...
- Problemas NP-completos
- Transición de fase

UPV - EHU

Organización del tema

1. Introducción
2. Problemas de optimización combinatoria
3. Complejidad de problemas de optimización
4. Métodos Clásicos vs Métodos Heurísticos

Ventajas de los clásicos frente a los heurísticos

- Alcanzan la solución óptima
- Muy desarrollados y estudiados matemáticamente
- Experimentalmente muy probados

Desventajas de los clásicos frente a los heurísticos

- Imposible acometer problemas de alta dimensionalidad
- Necesidad de una formulación matemática del problema
- Implementación complicada

Ventajas de los heurísticos frente a los clásicos

- Tiempos de cómputo razonables
- Aplicables de manera general
- Fáciles de implementar y entender

Desventajas de los heurísticos frente a los clásicos

- Son algoritmos de optimización aproximados
- Están poco estudiados matemáticamente
- Difícil realizar comparaciones entre diferentes métodos

A modo de conclusión

Aspectos importantes

- Búsqueda de la excelencia en optimización: diseño de un método específico para cada problema
- Algoritmos metaheurísticos: son algoritmos de propósito general, se pueden aplicar a cualquier problema
- No existe un método de optimización que sea mejor que otro: No free lunch