Análisis y Diseño de Algoritmos II

Juan Francisco Díaz, Ph.D Robinson Duque, Ph.D
Jesús Aranda Ph.D

Universidad del Valle

juanfco.diaz@correounivalle.edu.co
robinson.duque@correounivalle.edu.co
jesus.aranda@correounivalle.edu.co

Programa de Ingeniería de Sistemas
Escuela de Ingeniería de Sistemas y Computación

Overview

Introducción- Conceptos Generales

Optimizar es el proceso de maximizar o minimizar una función objetivo deseada mientras se satisface un conjunto de restricciones predominantes [Belegundu and Chandrupatla, 2019].

La naturaleza tiene abundantes ejemplos donde se busca un sistema óptimo:

- Una gota de agua en gravedad cero es una esfera perfecta, la cual es una figura geométrica de menor área de superficie para un volumen dado
- La estructura de los panales de abejas es uno de los arreglos más compactos. Los alvéolos tienen forma hexagonal, forma estable y compacta de utilización del espacio
- La mutación genética para sobrevivir es otro ejemplo de proceso de optimización natural

Introducción- Conceptos Generales

- Definir un programa de producción que maximice los ingresos totales
- Determinar la secuencia de tipos de productos a fabricar tal que se minimice el tiempo total de la producción
- Determinar la ubicación de sucursales que optimice las ventas
- Determinar la ubicación y dosificación de rayos X a aplicar para maximizar el efecto de un tratamiento de cáncer con el mínimo daño en tejidos sanos

Pasos para resolver un problema de optimización

1. Formulación del problema
2. Construcción del modelo matemático que representa el sistema o problema bajo estudio
3. Solución a partir del modelo (algoritmos - implementación del modelo bajo un lenguaje de modelado para problemas de optimización)
4. Verificación del modelo y la solución
5. Derivación de la solución del problema (decisión a tomar)

El problema de programación no-lineal (NLP)

La mayoría de problemas de optimización pueden ser expresados como problemas de mínimos (o máximos) de una función sujeta a restricciones de igualdades y desigualdades, lo cual se conoce usualmente como el problema de programación[1] no-lineal (NLP).

[1] La palabra " programación" significa " planificación"

El problema de programación no-lineal (NLP)

Modelo: descripción esquemática de un sistema, teoría, o fenómeno a partir de propiedades conocidas o inferidas y que puede ser utilizado para estudiar sus características.

Modelo matemático: modelos abstractos que describen la relación matemática entre elementos en un sistema.

El problema de programación no-lineal (NLP)

Notación de un NLP:

$$
\begin{array}{ll}
\texttt{minimize} & f(x) \\
\texttt{subject to} & g_i(x) \leq 0 \quad \text{i} = 1, \ldots, \text{m} \\
\texttt{and} & h_j(x) = 0 \quad \text{j} = 1, \ldots, \text{l} \\
\texttt{and} & x^L \leq x \leq x^U
\end{array}
$$

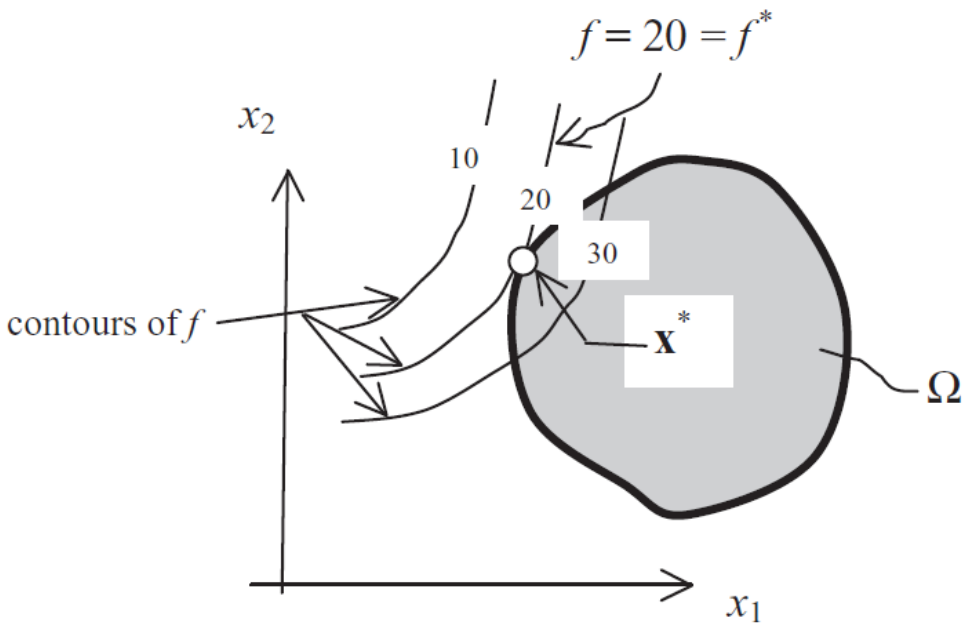
donde $x = (x_1, x_2, ..., x_n)^T$ es un vector columna de n variables de decisión. $f(x)$ es la función objetivo o función de costo; las $g's$ son restricciones de desigualdad, y las $h's$ son restricciones de igualdad. Los vectores x^L y x^U representan cotas o límites inferiores y superiores de las variables x.

Región factible

Otra representación de un NLP:

$$\begin{array}{ll} \texttt{minimize} & f(x) \\ \texttt{subject to} & x \in \Omega \end{array}$$

donde $\Omega = \{x : g(x) \leq 0, h(x) = 0, x^L \leq x \leq x^U\}$. Ω, es un subconjunto de R^n y se denomina región factible.

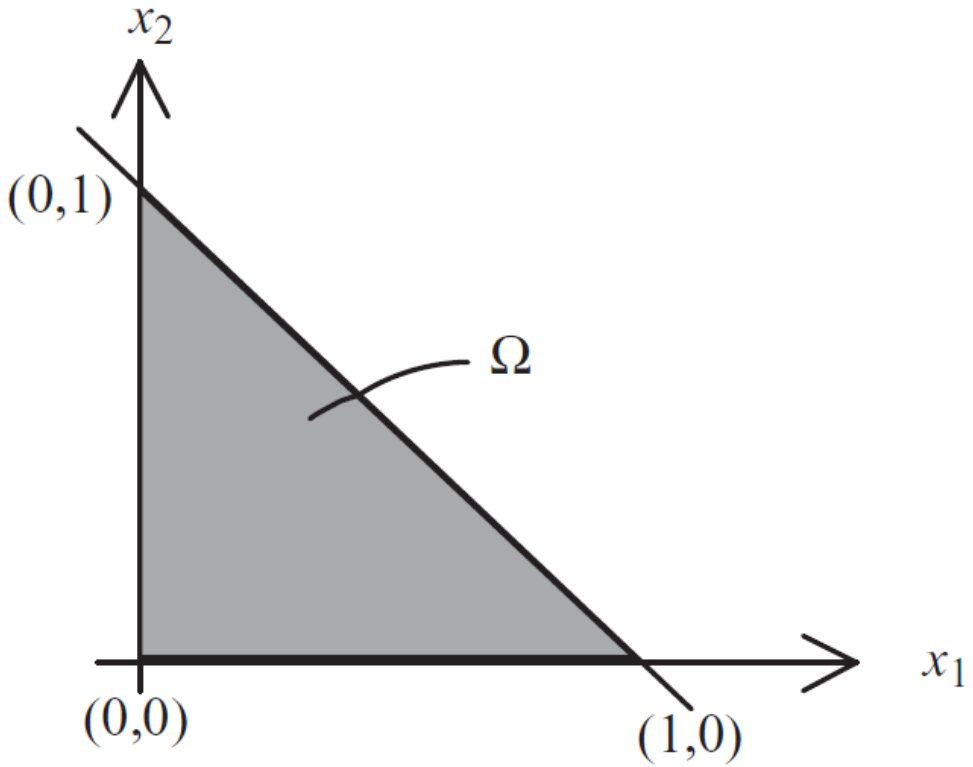


Región factible

Considere las siguientes restricciones:

$$\{g_1 \equiv x_1 \geq 0, g_2 \equiv x_2 \geq 0, g_3 \equiv x_1 + x_2 \leq 1\}$$

La región factible Ω se muestra en la siguiente figura:



Existencia de mínimos y máximos (Teorema de Wierstraas)

Theorem

Si f es una función continua, definida sobre un conjunto cerrado y acotado $\Omega \subseteq D(f)$. Entonces existen puntos x^* y x^{**} donde f toma sus valores máximo y mínimo[a].

[a] Aunque es cierto que puede existir una solución cuando no se cumplen las condiciones, esto es muy poco probable en situaciones prácticas.

Esto quiere decir que $f(x^*)$ y $f(x^{**})$ representan respectivamente los valores máximo y mínimo de f.

$D(f)$ representa el dominio de la función.

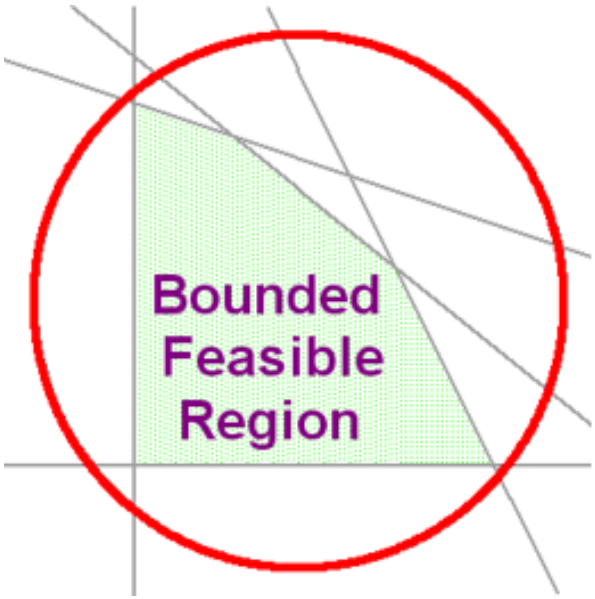
Existencia de mínimos y máximos (Teorema de Wierstraas)

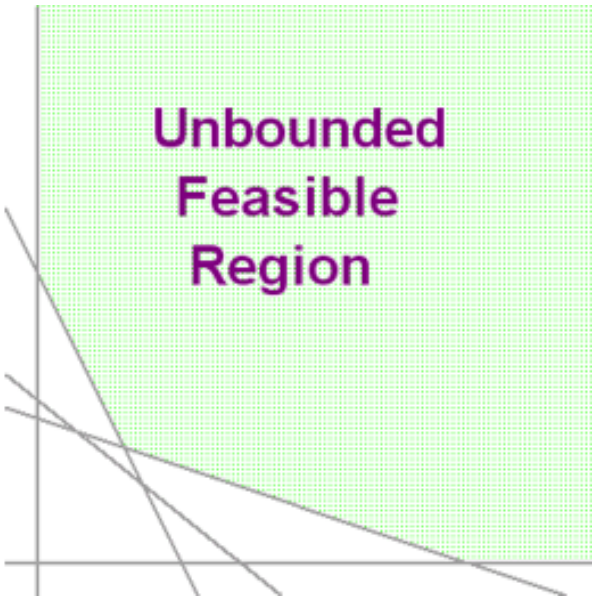
Una función f es continua en un punto a, si dada una secuencia $\{x_k\}$ en $D(f)$ que converge en a, entonces $f(x_k)$ debe converger en $f(a)$ (i.e., límite por la izquierda = límite por la derecha = $f(a)$).

Un conjunto es cerrado si contiene puntos limites. El conjunto $\{x : |x| \leq 1\}$ es un ejemplo de conjunto cerrado; $\{x : |x| < 1\}$ es un ejemplo de conjunto abierto.

Existencia de mínimos y máximos (Teorema de Wierstraas)

Una región Ω es acotada (i.e., bounded) si puede ser contenida dentro de una esfera de radio finito. Por ejemplo, el conjunto de todos los enteros positivos $\{1,2,..\}$, es no acotado.





Existencia de mínimos y máximos (Teorema de Wierstraas)

Ejemplo, considere el siguiente problema:

$$\begin{aligned} \texttt{minimize} \quad & f(x) = x \\ \texttt{subject to} \quad & 0 < x \leq 1 \end{aligned}$$

Evidentemente no tiene solución, podemos observar que el conjunto de restricciones Ω no es cerrado. Reescribiendo la restricción como $0 \leq x \leq 1$ resulta en $x = 0$ como solución.

```
% Implementacion en MiniZinc:
var 0.0..1.0: x; % Variable flotante (continua)
constraint x<=1;
constraint x>0; % No cerrado, no hay solucion
solve minimize x;
output ["x=", show(x)]

% Ejecutar con (Gecode Gist) o con (Gcode)
```

Existencia de mínimos y máximos (Teorema de Wierstraas)

Ejemplo, considere el siguiente problema:

$$\begin{aligned} \texttt{minimize} \quad & f(x) = x \\ \texttt{subject to} \quad & 0 < x \leq 1 \end{aligned}$$

Evidentemente no tiene solución, podemos observar que el conjunto de restricciones Ω no es cerrado. Reescribiendo la restricción como $0 \leq x \leq 1$ resulta en $x = 0$ como solución.

```
% Implementacion en MiniZinc:
var 0.0..1.0: x; % Variable flotante (continua)
constraint x<=1;
constraint x>0; % No cerrado, no hay solucion
solve minimize x;
output ["x=", show(x)]

% Ejecutar con (Gecode Gist) o con (Gcode)
```

Existencia de mínimos y máximos (Teorema de Wierstraas)

Ejemplo, considere el siguiente problema:

$$\begin{aligned} \texttt{minimize} \quad & f(x) = x \\ \texttt{subject to} \quad & 0 < x \leq 1 \end{aligned}$$

Evidentemente no tiene solución, podemos observar que el conjunto de restricciones Ω no es cerrado. Reescribiendo la restricción como $0 \leq x \leq 1$ resulta en $x = 0$ como solución.

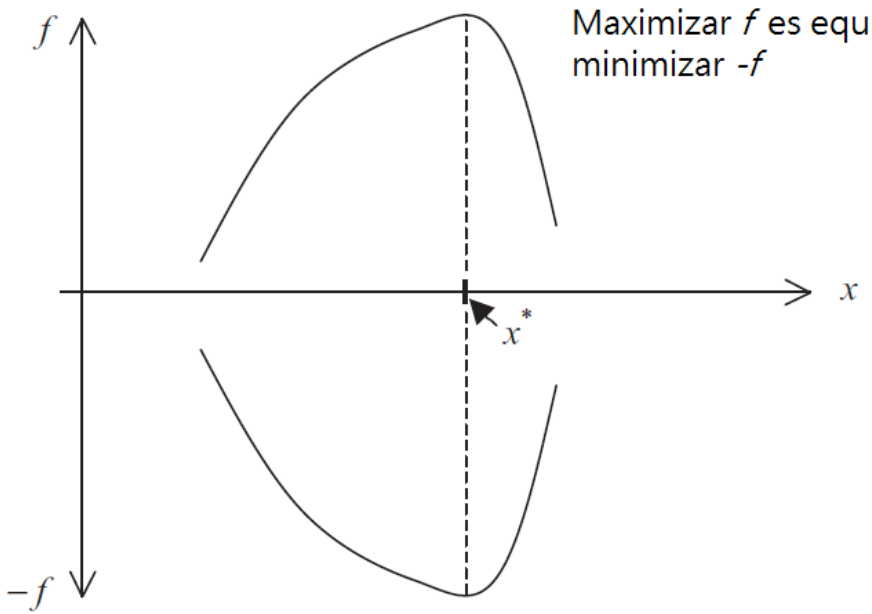
```
% Implementacion en MiniZinc:
var 0.0..1.0: x; % Variable flotante (continua)
constraint x<=1;
constraint x>0; % No cerrado, no hay solucion
solve minimize x;
output ["x=", show(x)]

% Ejecutar con (Gecode Gist) o con (Gcode)
```

MiniZinc

MiniZinc es un lenguaje de modelado para problemas de restricciones (satisfacción y optimización) que ha tomado fuerza en los últimos años. Es la herramienta que utilizaremos para el curso, más adelante introduciremos la herramienta en detalle.

- Principalmente desarrollado por las Universidades Melbourne y Monash, Australia
- Introducido en el 2007
- Página: `http://www.minizinc.org`
- Cursos disponibles en Coursera (`https://www.coursera.org/`)

Maximizar vs. Minimizar



Note en la figura que maximizar la función objetivo f es equivalente a minimizar la función $-f$.

Ejemplo: la función (maximizar: $3x_1 + 4x_2 - x_3$) es equivalente a (minimizar: $-3x_1 - 4x_2 + x_3$)

Maximizar vs. Minimizar

Ejemplo: Distancia más corta entre un punto $p = (x, y)$ y una línea $a_0 + a_1 x_1 + a_2 y_1 = 0$. Se pueden plantear los siguientes modelos equivalentes:

$$\begin{array}{ll} \texttt{minimize} & f = (x_1 - x)^2 + (y_1 - y)^2 \\ \texttt{subject to} & h(x) \equiv a_0 + a_1 x_1 + a_2 x_2 = 0 \end{array}$$

$$\begin{array}{ll} \texttt{maximize} & f = -(x_1 - x)^2 - (y_1 - y)^2 \\ \texttt{subject to} & h(x) \equiv a_0 + a_1 x_1 + a_2 x_2 = 0 \end{array}$$

Maximizar vs. Minimizar

```
% Implementación en MiniZinc:
int: x = 2;  % Valor constante
int:  y = 6; % Valor constante

var 0..100: x_1; % Variable entera
var 0..100: y_1; % Variable entera

constraint y_1 - 2*x_1 - 5 = 0;    % (f(x) = 2x+5)

solve minimize pow((x_1 - x),2) + pow((y_1 - y),2);
%solve maximize -pow((x_1 - x),2) - pow((y_1 - y),2);

output ["x_1=", show(x_1), "\n y_1=", show(y_1), ""];
```

Tipos de variables y problemas de optimización

Tipo de variables:

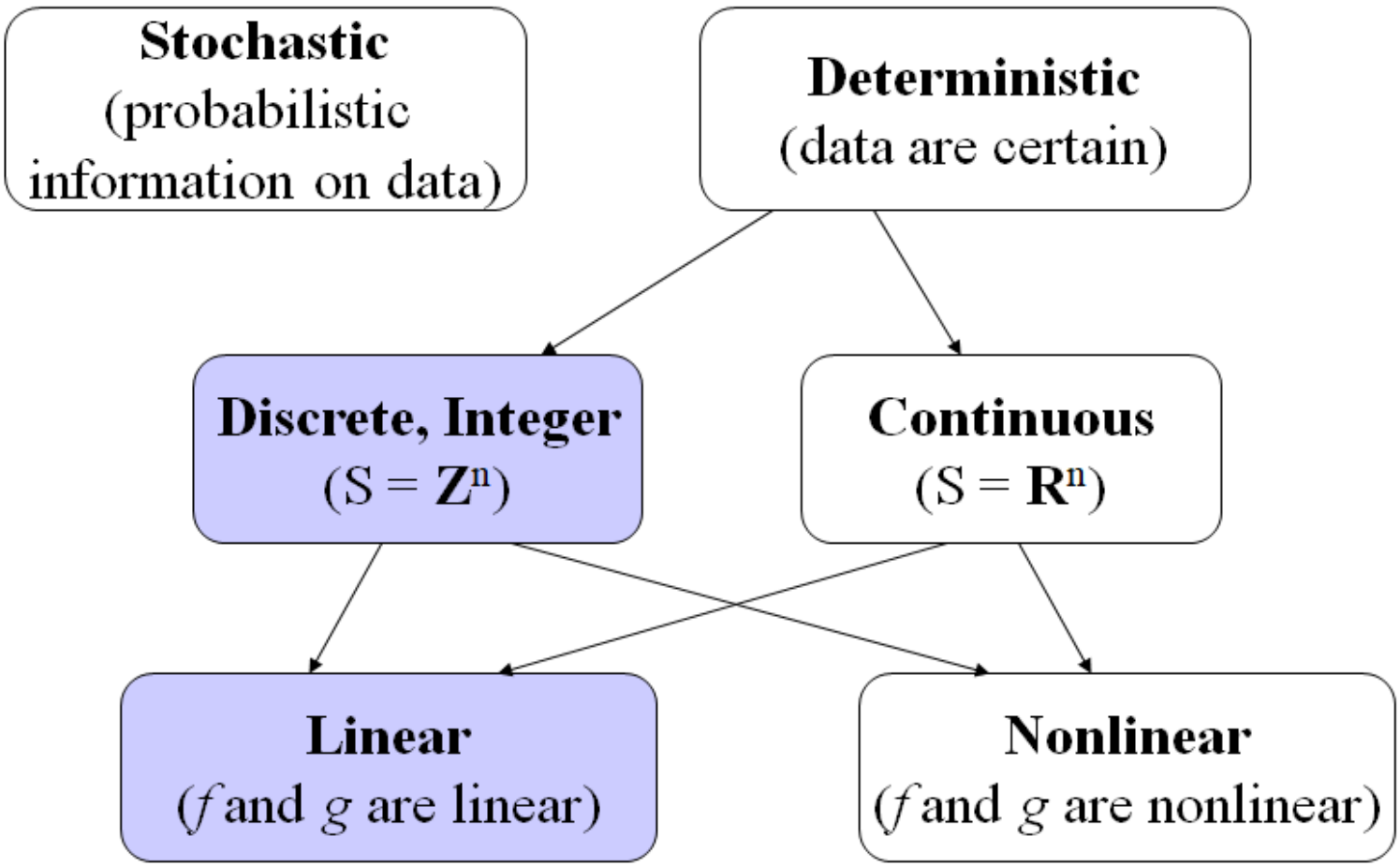
- x_j puede ser continua
- x_j puede ser binaria (0 o 1)
- x_j puede ser entera (1, 2, 3,... ,N)
- x_j puede ser discreta (e.g., tomar valores 10mm, 20mm, 30mm)

Tipos de variables y problemas de optimización

Diferentes nombres especializados se le dan al problema NLP:

- **Programación Lineal (LP):** cuando todas las funciones (objetivo y restricciones) son lineales
- **Programación Entera (IP):** un LP con la restricción que todas las variables deben ser enteras
- **Programación binaria:** caso especial de IP donde las variables son 0 o 1
- **Programación Entera Mixta (MIP):** un IP donde algunas de las variables son enteras y otras continuas
- **MINLP:** un MIP con funciones no lineales
- **Programación Cuadrática (QP):** cuando una función objetivo es una función cuadrática en x y todas las restricciones son lineales

Tipos de variables y problemas de optimización



Programación Lineal- Introducción

Programación lineal (LP) es el término utilizado para definir una amplia gama de problemas de optimización:

- la función objetivo que se debe minimizar o maximizar es lineal en las variables desconocidas
- las restricciones son una combinación de igualdades y desigualdades lineales

Programación Lineal- Introducción

- los problemas de LP ocurren en muchas situaciones de la vida real donde los beneficios deben maximizarse o los costos deben minimizarse con límites de restricción de recursos
- estudiaremos el método símplex para solucionar problemas de LP, sin embargo, *nos enfocaremos principalmente en el modelado de problemas* debido a que se vuelve necesario el uso de computadoras incluso para un pequeño número de variables
- comúnmente utilizado en problemas que involucran decisiones de dieta, transporte, producción y manufactura, combinación de productos, análisis de límites de ingeniería en diseño, programación de aerolíneas, etc.

Programación Lineal- Forma general

La forma general de un problema de programación lineal tiene una función objetivo y un conjunto de restricciones:

$$\texttt{maximize} \quad c_1x_1 + c_2x_2 + c_3x_3 + ... + c_nx_n$$
$$\texttt{subject to} :$$

$$\% \text{ Restricciones LE } (i = 1...l)$$
$$a_{i1}x_1 + a_{i2}x_2 + a_{i3}x_3 + ... + a_{in}x_n \leq b_i$$

$$\% \text{ Restricciones GE } (j = l + 1...l + r)$$
$$a_{j1}x_1 + a_{j2}x_2 + a_{j3}x_3 + ... + a_{jn}x_n \geq b_j$$

$$\% \text{ Restricciones EQ } (k = l + r + 1...l + r + q)$$
$$a_{k1}x_1 + a_{k2}x_2 + a_{k3}x_3 + ... + a_{kn}x_n = b_k$$

$$x_1 \geq 0, x_2 \geq 0, ..., x_n \geq 0$$

Programación Lineal- Forma general

- El número de restricciones es $m = l + r + q$
- c_j y a_{ij} son coeficientes constantes
- b_j son constantes reales fijas, los cuales están ajustados a valores no negativos
- x_j son variables de decisión
- los límites de i y j son: $i = 1...m$ equivalente al número de restricciones; $j = 1...n$ equivalente al número de variables

Los problemas LP son problemas **convexos**, lo que implica que un máximo local es de hecho un máximo global.

Programación Lineal- Ejemplo Introductorio

Una empresa que fabrica computadores de mesa y notebooks desea saber cuántos computadores debe producir para maximizar sus ganancias:

- Cada computador (de mesa o notebook) requiere de un chip de procesamiento. La empresa cuenta con 10.000 chips.
- Cada computador requiere memoria. La memoria viene en chips de 16GB, un notebook requiere 1 chip (16GB), mientras un computador de mesa requiere de 2 chips (32GB). Se cuenta con un inventario de 15.000 chips.
- Cada computador requiere tiempo de ensamblaje, un notebook toma 4 minutos y uno de mesa toma 3 minutos. Se tienen 25.000 minutos de ensamblaje disponibles.
- Cada notebook genera \$750 de ganancia y uno de mesa genera \$1000.

Programación Lineal- Ejemplo Introductorio

Algunas preguntas:

- ¿Cuántos computadores de cada tipo se deben producir para maximizar las ganancias?
- ¿Cuál es la ganancia máxima que se puede obtener?

Programación Lineal- Ejemplo Introductorio

Modelamiento: escribir el problema en lenguaje de programación lineal. Definir las variables de decisión, el objetivo y las restricciones:

- **Variables de decisión:** a diferencia de valores del problema que son dados o que pueden ser calculados simplemente de lo que nos proveen, las variables representan valores desconocidos.

  En este caso las variables son el número de notebooks y el número de computadores de mesa y las representaremos con x_1 y x_2 respectivamente.

Programación Lineal- Ejemplo Introductorio

Modelamiento: escribir el problema en lenguaje de programación lineal. Definir las variables de decisión, el objetivo y las restricciones:

- **Variables de decisión:** a diferencia de valores del problema que son dados o que pueden ser calculados simplemente de lo que nos proveen, las variables representan valores desconocidos.

  En este caso las variables son el número de notebooks y el número de computadores de mesa y las representaremos con x_1 y x_2 respectivamente.

Programación Lineal- Ejemplo Introductorio

Modelamiento: escribir el problema en lenguaje de programación lineal. Definir las variables de decisión, el objetivo y las restricciones:

- **Función objetivo:** cada LP tiene una función objetivo a maximizar o minimizar. El objetivo debe ser lineal respecto a las variables de decisión, lo cual significa que debe ser una suma de constantes que multiplican las variables de decisión (e.g., $3x_1 + 2x_2$); $(x_1 x_2)$ no es lineal.

En este caso nuestro objetivo es maximizar las ganancias y sabemos que cada notebook genera \$750 de ganancia y uno de mesa genera \$1000. Por lo tanto tendremos que:

$$Maximizar : 750x_1 + 1000x_2$$

Programación Lineal- Ejemplo Introductorio

Modelamiento: escribir el problema en lenguaje de programación lineal. Definir las variables de decisión, el objetivo y las restricciones:

- **Función objetivo:** cada LP tiene una función objetivo a maximizar o minimizar. El objetivo debe ser lineal respecto a las variables de decisión, lo cual significa que debe ser una suma de constantes que multiplican las variables de decisión (e.g., $3x_1 + 2x_2$); $(x_1 x_2)$ no es lineal.

  En este caso nuestro objetivo es maximizar las ganancias y sabemos que cada notebook genera \$750 de ganancia y uno de mesa genera \$1000. Por lo tanto tendremos que:

$$Maximizar : 750x_1 + 1000x_2$$

Programación Lineal- Ejemplo Introductorio

Modelamiento: escribir el problema en lenguaje de programación lineal. Definir las variables de decisión, el objetivo y las restricciones:

- **Restricciones:** en este problema tenemos 4 tipos de restricciones: chips de procesamiento, memoria, tiempo de ensamblaje, no negatividad. Las restricciones deben ser lineales (e.g., $3x_1 + 2x_2 \geq 5$ es una restricción lineal); $(x_1 x_2 \leq 3$ o $x_1^2 \geq 7)$ no son lineales.

  Chips disponibles: $x_1 + x_2 \leq 10000$
  Memoria disponible: $x_1 + 2x_2 \leq 15000$
  Ensamblaje: $4x_1 + 3x_2 \leq 25000$
  No negatividad: $x_1 \geq 0$ y $x_2 \geq 0$

Programación Lineal- Ejemplo Introductorio

Modelamiento: escribir el problema en lenguaje de programación lineal. Definir las variables de decisión, el objetivo y las restricciones:

- **Restricciones:** en este problema tenemos 4 tipos de restricciones: chips de procesamiento, memoria, tiempo de ensamblaje, no negatividad. Las restricciones deben ser lineales (e.g., $3x_1 + 2x_2 \geq 5$ es una restricción lineal); $(x_1 x_2 \leq 3$ o $x_1^2 \geq 7)$ no son lineales.

  Chips disponibles: $x_1 + x_2 \leq 10000$
  Memoria disponible: $x_1 + 2x_2 \leq 15000$
  Ensamblaje: $4x_1 + 3x_2 \leq 25000$
  No negatividad: $x_1 \geq 0$ y $x_2 \geq 0$

Programación Lineal- Ejemplo Introductorio

Modelo final:

$$
\begin{aligned}
\texttt{maximize} \quad & f = 750x_1 + 1000x_2 \\
\texttt{subject to} \quad & x_1 + x_2 \leq 10000 \\
& x_1 + 2x_2 \leq 15000 \\
& 4x_1 + 3x_2 \leq 25000 \\
& x_1 \geq 0 \\
& x_2 \geq 0
\end{aligned}
$$

Programación Lineal- Ejemplo Introductorio

Implementación en MiniZinc

```
var int: x_1;  % Variable entera sin cota superior
var int: x_2;  % Variable entera sin cota superior

constraint x_1 + x_2 <= 10000;
constraint x_1 + 2*x_2 <= 15000;
constraint 4*x_1 + 3*x_2 <= 25000;
constraint x_1 >= 0;
constraint x_2 >= 0;

solve maximize 750*x_1 + 1000*x_2;

output ["x_1=", show(x_1),"\n x_2=",show(x_2)];
```

Solución: $x_1 = 1000$ y $x_2 = 7000$

Fin de la Presentación

¿Preguntas?

References I

Belegundu, A. D. and Chandrupatla, T. R. (2019).

Optimization concepts and applications in engineering.

Cambridge University Press.