UNIVERSIDAD
DE SANTIAGO
DE CHILE

Optimización I

Luis Rojo-González
luis.rojo.g@usach.cl
Departamento de ingeniería industrial,
Universidad de Santiago, Chile

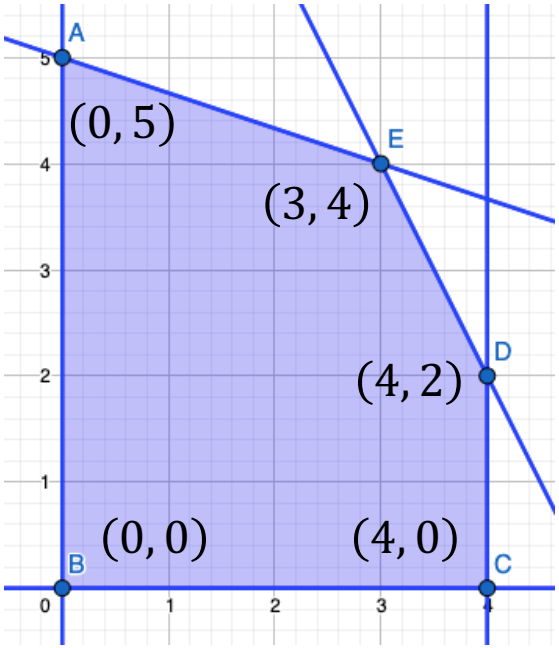
Ingeniería civil industrial

Introducción

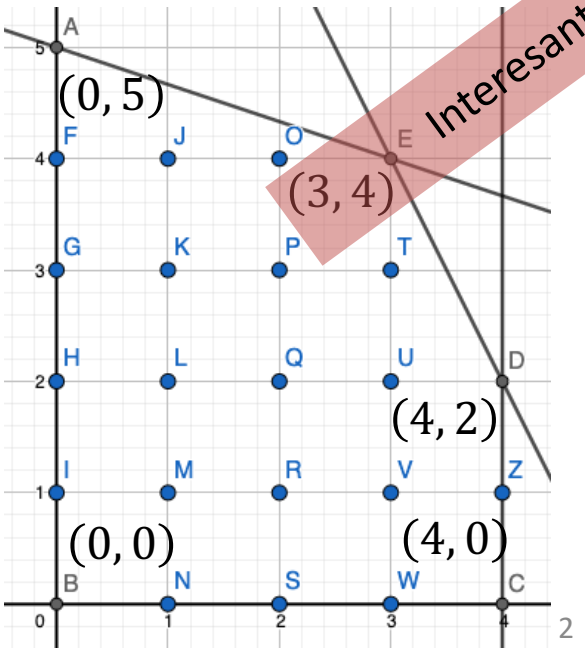
En general, los modelos matemáticos que *intentan* representar la realidad siempre considerarán variables que deban tener valores enteror (notar que una variable binaria también lo es, tan solo que se le da una connotación distinta). Como se ha visto –de manera informal– hasta ahora, una modelo de problemación entera se define como

$$\max c^T x$$
$$s.a. \quad Ax \leq b$$
$$x \in Z_+$$

Entonces, el politopo convexo es (problema mesas y sillas)



Si se impone integralidad en las varibales



UNIVERSIDAD DE SANTIAGO DE CHILE

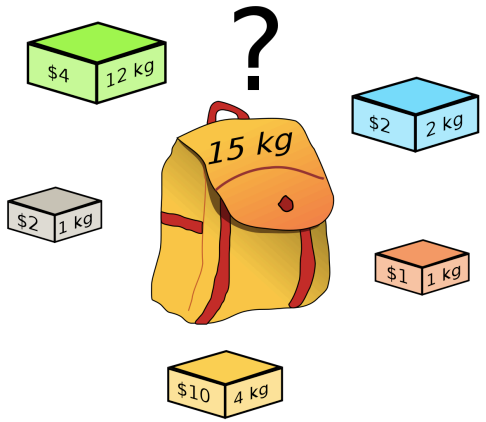
Problema de la mochila (Knapsack problem)

Este problema tiene relación directa con el nombre que recibe, es decir, se tiene un espacio de capacidad limitada (b) y elementos (x_i) con un cierto beneficio (w_i) y tamaño (p_i) que pueden (o no) ser incorporados a este espacio limitado (variable binaria).

$$\max \sum_{i \in I} w_i x_i$$

$$s.a. \quad \sum_{i \in I} p_i x_i \leq b$$

$$x \in \{0,1\}^{|I|}$$



En particular, este problema tiene distintas variaciones; y es claro ver que a medida que crece el número de elementos candidatos (cardinalidad de I, $|I|$), el tiempo de resolución de este problema crecerá.

Por otro lado, existen algoritmos optimales para resolver este tipo de problema como la regla de Smith, por ejemplo.

UNIVERSIDAD DE SANTIAGO DE CHILE

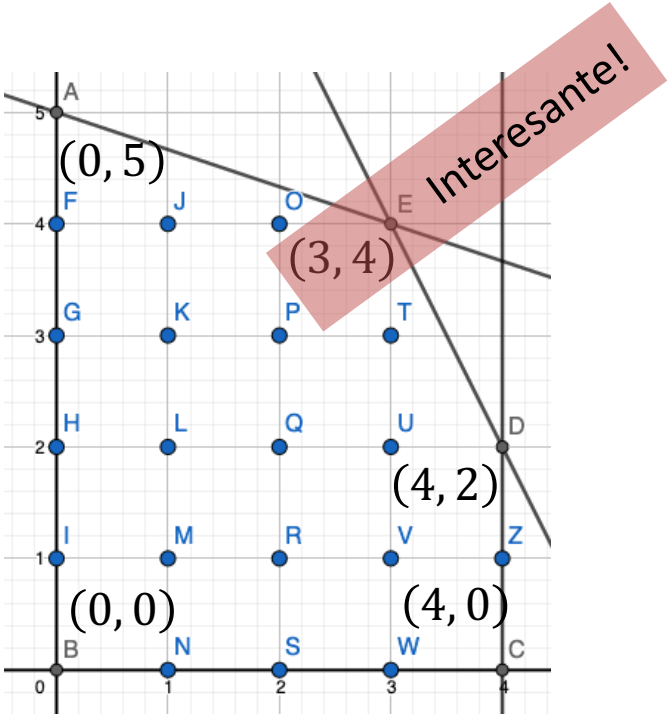
El problema del vendedor viajero (Travelling salesman problem, TSP)

Este problema ha sido ampliamente estudiado desde distintos puntos de vista, dado que existen distintas formulaciones que son mejores que otras dependiendo del algoritmo de resolución. Imagine que quiere planear sus vacaciones, el problema consiste en encontrar un camino de mínimo coste (distancia) para visitar cada una de las localidades de interés sin tener que visitar un lugar ya visitado, esto es conocido como un ciclo *Hamiltoniano*. Existen dos clases de este problema dependiendo si las distancias de ida-vuelta son simétricas.

UNIVERSIDAD DE SANTIAGO DE CHILE

Algoritmo de ramificación y poda (Branch-and-Bound, B&B)

El algoritmo Branch-and-bound (B&B) es un método que se como complemento al método simplex ya revisado en la primera parte del curso. Este algoritmo lo que hace, y en honor a su nombre, es encontrar la solución del problema relajado (restricción de integralidad) y, luego, agregar restricciones de acuerdo al valor actual de la variable en cuestión.



Notar que en este ejemplo, existe un conjunto de restricciones que se intersectan en una coordenada con componentes enteras.

Entonces, de esto se puede reconocer que no sería necesario tener un algoritmo que pruebe de manera *inteligente* la combinación de coordenadas, sino que es mejor idea encontrar un conjunto de restricciones que permitan obtener una solución como representa la figura, es decir, restricciones que se intersecten en puntos con coordenadas enteras.

Algoritmo de ramificación y poda (Branch-and-Bound, B&B)

En este sentido, el agoritmo B&B lo que hace es incluir restricciones basadas en el valor actual de la variable que debe ser actual de forma secuencial. Formalmente es resolver sub-problemas basados en análisis de sensibilidad (basados en incorporar nuevas restricciones) de la siguiente manera:

Sea $P_0 \coloneqq \max\{c^T x : x \in S\}$, tal que $S \coloneqq \{Ax \leq b, x \geq 0\}$ representa el politopo convexo del problema de programación lineal entera relajado. Además, considere que en el vector solución de P existe una variable $x_j = m$, tal que $m \notin Z_+$.

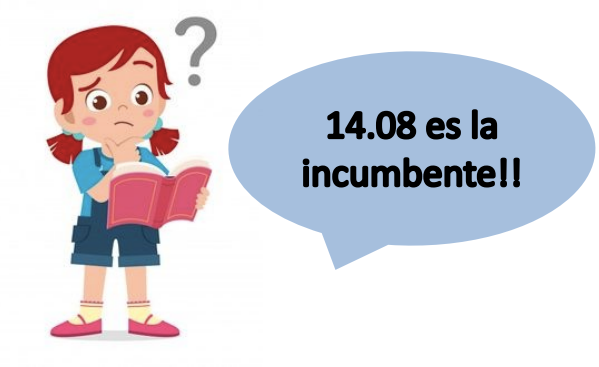
De esta manera se resuelven los subproblemas $P_1 \coloneqq \max\{c^T x : x \in (S \cap x_j \leq \lfloor m \rfloor)\}$ y $P_2 \coloneqq \max\{c^T x : x \in (S \cap x_j \geq \lfloor m \rfloor + 1)\}$.

Algoritmo de ramificación y poda
Ejemplo 1

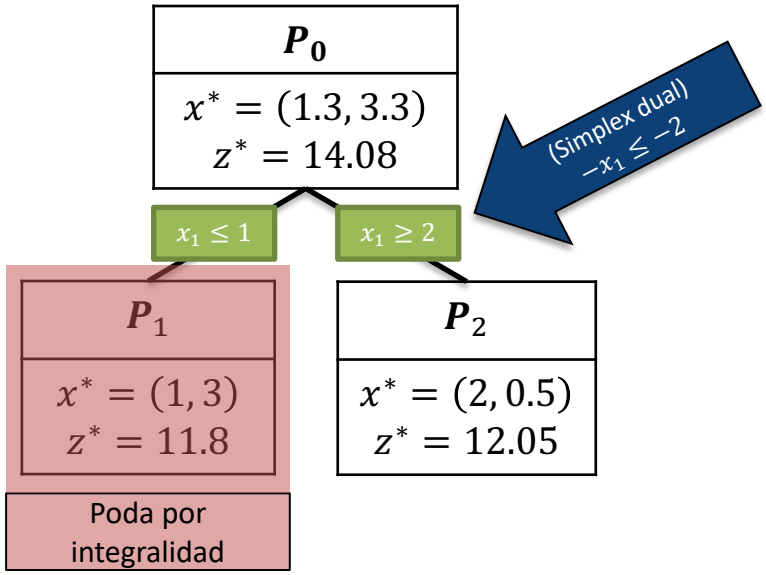
Considere el problema de programación entera dado por

$$\max z := 5.5x_1 + 2.1x_2$$
$$s.a. \quad -x_1 + x_2 \leq 2$$
$$8x_1 + 2x_2 \leq 17$$
$$x_1, x_2 \in Z_+$$

La solución del problema relajado está dado por el vector $(x_1^*, x_2^*) = (1.3, 3.3)$ con un valor objetivo $z^* = 14.08$.



Ahora, se debe elegir una variable para ramificar, sigamos x_1.



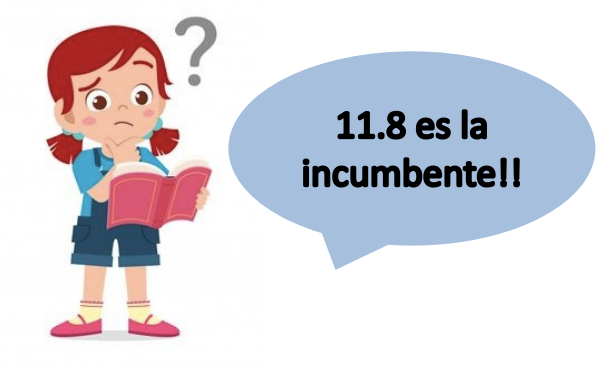
UNIVERSIDAD DE SANTIAGO DE CHILE

Algoritmo de ramificación y poda
Ejemplo 1

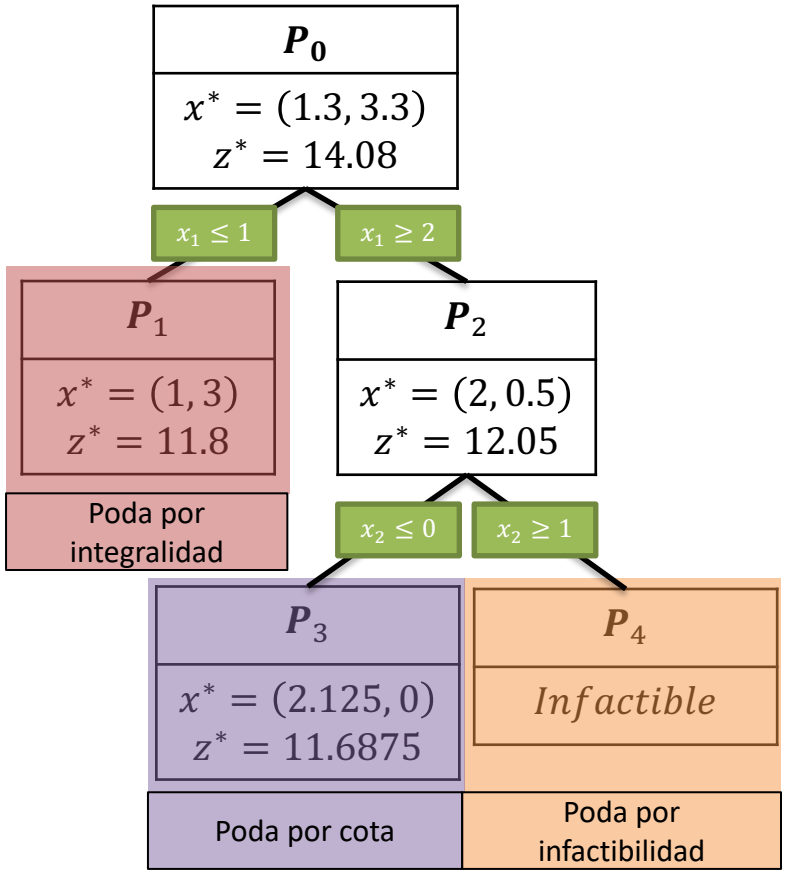
Considere el problema de programación entera dado por

$$\max z := 5.5x_1 + 2.1x_2$$
$$s.a. \quad -x_1 + x_2 \leq 2$$
$$8x_1 + 2x_2 \leq 17$$
$$x_1, x_2 \in Z_+$$

La solución del problema relajado está dado por el vector $(x_1^*, x_2^*) = (1.3, 3.3)$ con un valor objetivo $z^* = 14.08$.

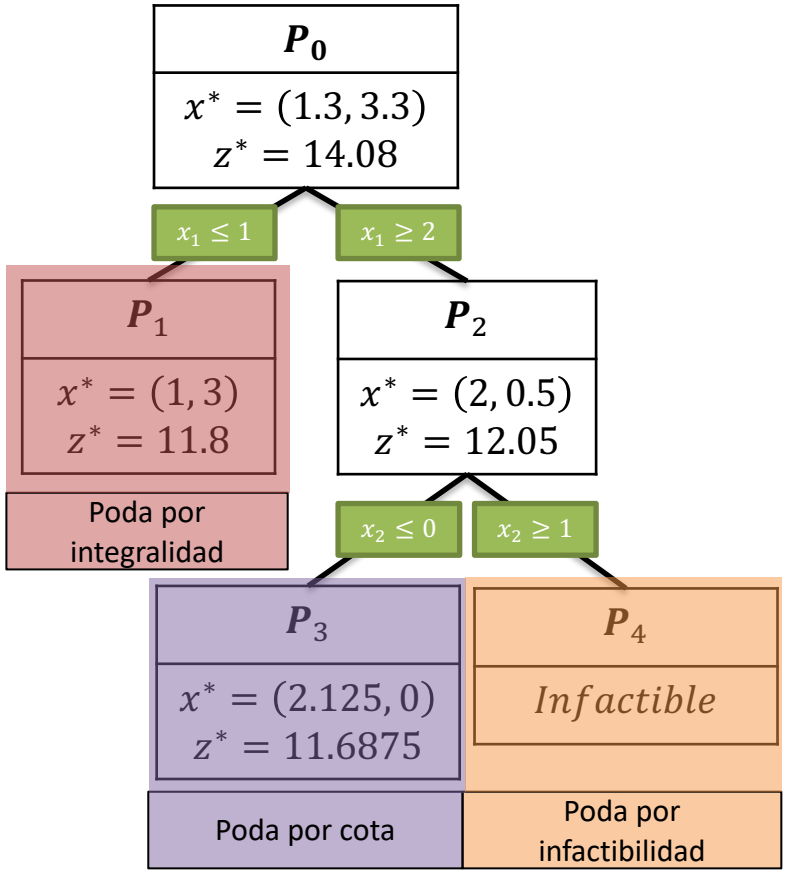


Ahora, se debe elegir una variable para ramificar, sigamos x_2.

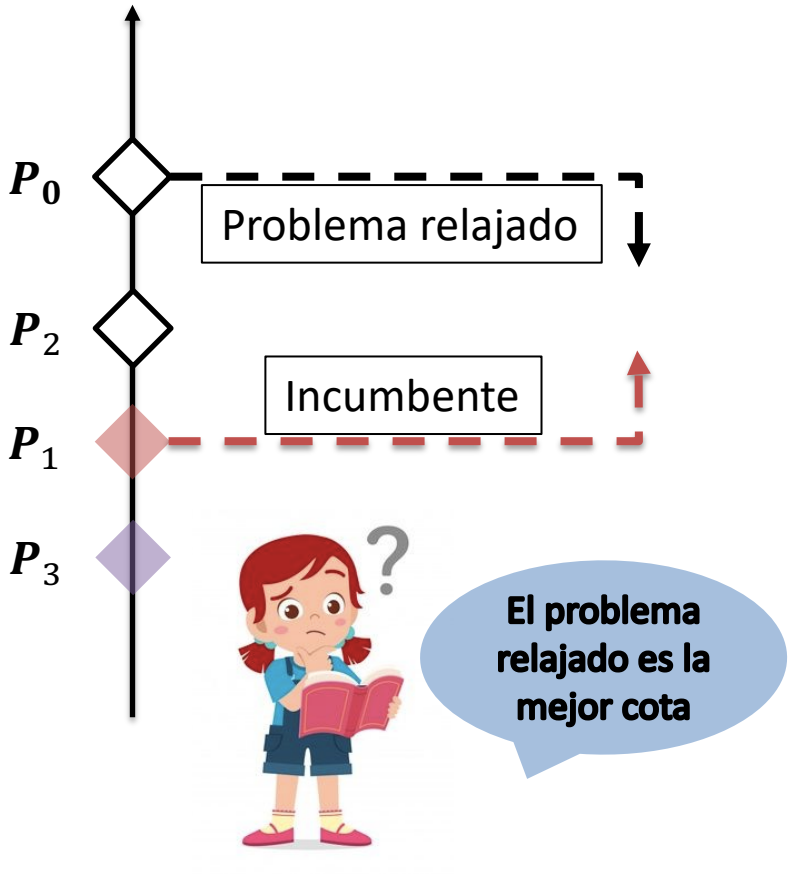


UNIVERSIDAD DE SANTIAGO DE CHILE

Algoritmo de ramificación y poda
Ejemplo 1



Recordando que es un problema de maximización, de forma esquemática se tiene:



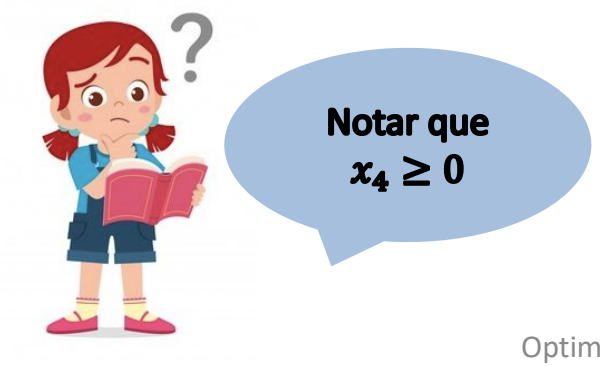
UNIVERSIDAD DE SANTIAGO DE CHILE

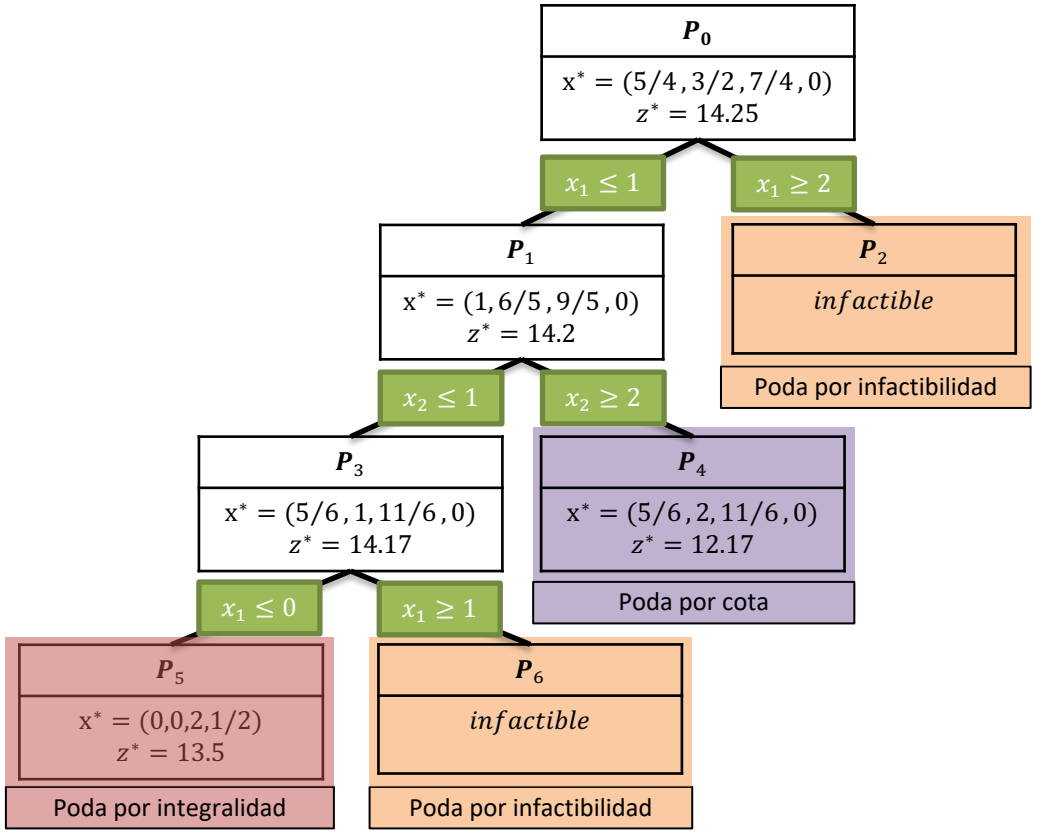
Algoritmo de ramificación y poda
Ejemplo 2

Considere el problema de programación entera dado por

$$\max z := 4x_1 - 2x_2 + 7x_3 - x_4$$
$$s.a. \quad x_1 + 5x_3 \leq 10$$
$$x_1 + x_2 - x_3 \leq 1$$
$$6x_1 - 5x_2 \leq 0$$
$$-x_1 + 2x_3 - 2x_4 \leq 3$$
$$x_1, x_2, x_3 \in Z_+, x_4 \geq 0$$

La solución del problema relajado está dado por el vector $x^* = (5/4, 3/2, 7/4, 0)$ con un valor objetivo $z^* = 14.25$.





Algoritmo de ramificación y poda
Ramificación eficiente

De acuerdo a lo anterior, se ha visto que la elección de la variable a ramificar es aleatoria, pero ¿existe alguna manera de elegir cuál se debe ramificar para encontrar una incumbente de antemano?

La respuesta es sí. El método que revisaremos corresponde al método de las penalizaciones y se define de la siguiente manera.

Considere que en el vector solución existe una variable $x_j = m$, tal que $m \notin Z_+$. Entonces, la ramificación será hacia el entero menor ó el mayor. De esta manera, se tiene que existe una diferencia hacia este entero menor y al mayor $\underline{f_j} = x_j - \lfloor m_j \rfloor$ y $\overline{f_j} = x_j - (\lfloor m_j \rfloor + 1) = x_j - \lceil m_j \rceil$, respectivamente.

Algoritmo de ramificación y poda
Ramificación eficiente

Entonces, es posible obtener los incrementos (en caso de minimizar) de la función objetivo cuando se agrega:

- $x_j \leq \lfloor m_j \rfloor$: $\quad D_i = \min_{j \in R} \left(\frac{\overline{f_j}\overline{c_j}}{(B^{-1}R)_{i,k \notin R}} : (B^{-1}R)_{i,k \notin R} > 0 \right)$.
- $x_j \geq \lfloor m_j \rfloor + 1$: $U_i = \min_{j \in R} \left(\frac{\overline{f_j}\overline{c_j}}{(B^{-1}R)_{i,k \notin R}} : (B^{-1}R)_{i,k \notin R} < 0 \right)$.

Luego, se tiene que existe una cota inferior de incremento dada por $P_i = \min_i \{D_i, U_i\}$.

Por otro lado, teniendo en cuenta las condiciones de integralidad; para tener una solución entera, alguna variable no-básica debe tomar valor positivo no menor a 1; con lo cual otra cota inferior está dada por $\hat{P} = \min_{j \in R} \overline{c_j}$. Entonces, la mejor cota inferior del incremento viene dadad por $P^* = \max\{P_i, \hat{P}\}$.

La incumbente encontrada corresponde a $z' = \lfloor z^* + P^* \rfloor$ ó $z' = \lfloor z^* - P^* \rfloor$ si minimizo o maximizo, respectivamente.

Finalmente, ramificar por $x_j \leq \lfloor m_j \rfloor$ si $\max\{D_i, U_i\} = U_i$ y por $x_j \geq \lfloor m_j \rfloor + 1$ si $\max\{D_i, U_i\} = D_i$.

UNIVERSIDAD DE SANTIAGO DE CHILE

Algoritmo de ramificación y poda
Ramificación eficiente - Ejemplo 1

Considere el problema de programación entera dado por

$$\min -z := -5.5x_1 - 2.1x_2$$
$$s.a. \quad -x_1 + x_2 \leq 2$$
$$8x_1 + 2x_2 \leq 17$$
$$x_1, x_2 \in Z_+ \quad \boxed{x_1, x_2 \geq 0}$$

La base solución está dada por $x_B = (x_2, x_1) = (3.3, 1.3)$ y $x_R = (h_2, h_1)$.

$$B^{-1}R = \begin{pmatrix} 0.1 & 0.8 \\ 0.1 & -0.2 \end{pmatrix}$$
$$c_R - c_B B^{-1} R = (0.76, 0.58)$$

Por lo tanto, si ramifico

$$D_1 = \min\left\{(1.3 - \lfloor 1.3 \rfloor) \times \frac{0.76}{0.8}, .\right\} = 0.285, D_2 = \min\left\{(3.3 - \lfloor 3.3 \rfloor) \times \frac{0.76}{0.1} = 2.28, (3.3 - \lfloor 3.3 \rfloor) \times \frac{0.58}{0.1} = 1.74\right\} = 1.74$$

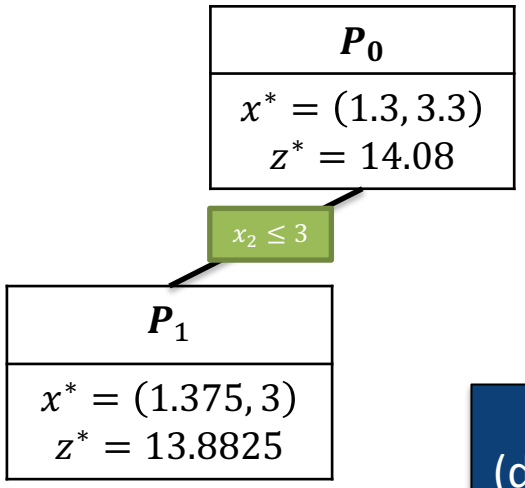
$$U_1 = \min\left\{., (1.3 - \lceil 1.3 \rceil) \times \frac{0.58}{-0.2}\right\} = 2.03, U_2 = \min\{.\} =. (= +\infty)$$

$$P_1 = \min\{D_1, U_1\} = \min\{0.285, 2.03\} = 0.285; P_2 = \min\{D_2, U_2\} = \min\{1.74, .\} = 1.74$$

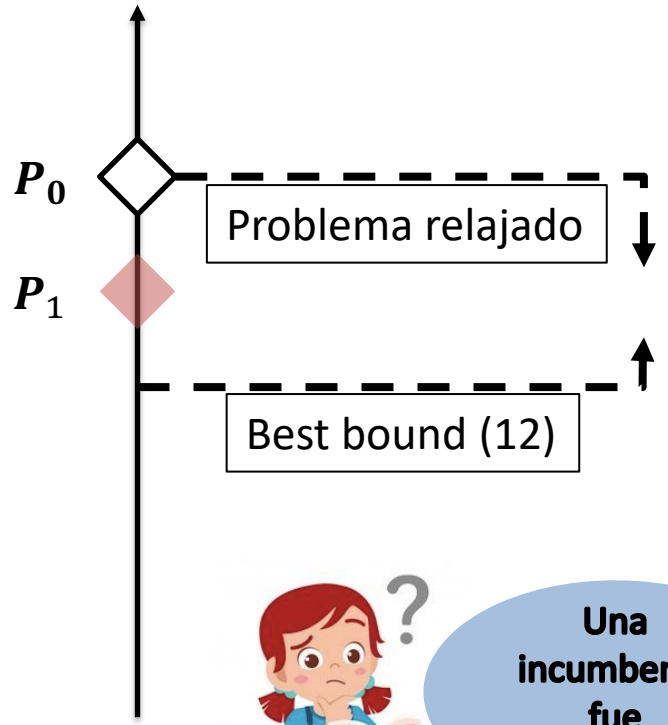
Luego, $\hat{P} = \min\{0.76, 0.58\} = 0.58$ y $P^* = \max\{P_1, P_2, \hat{P}\} = 1.74$, por lo que la incumbente encontrada es $z' = \lfloor 14.08 - 1.74 \rfloor = \lfloor 12.34 \rfloor = 12$.

Finalmente, elijo ramificar por $\max_i \{D_i, U_i\} = U_2 \rightarrow x_2 \leq \lfloor 3.3 \rfloor = 3$.

Algoritmo de ramificación y poda
Ramificación eficiente - Ejemplo 1



Recordando que es un problema de maximización, de forma esquemática se tiene:



GAP (de integralidad)

(13.8825-12)/12=0.16

Esto quiere decir que, al ramificar, si se encuentra alguna solución menor a esta incumbente, esta rama debe ser podada por el criterio de cota.