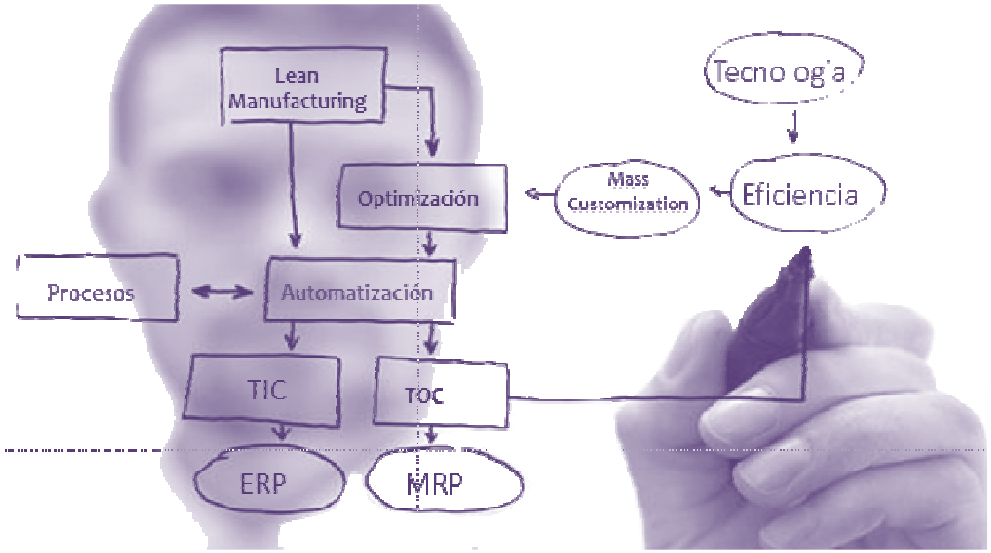


DEIOAC-UPV

Tema 5. Métodos de Programación Entera

Objetivos

Al finalizar el tema, deberás ser capaz de:

- Identificar la diferencia desde un punto de vista computacional entre distintas categorías de Programas Lineales.
- Conocer las principales características conceptuales y algebraicas de los algoritmos para la resolución de problemas de programación lineal entera.
- Aplicar estos algoritmos a problemas de dimensión reducida.

CONTENIDOS

5.1 Introducción

5.2. Técnicas de Programación Entera

- 5.2.1 Algoritmos Voraces
  - 5.2.1.1 El árbol de expansión mínima
- 5.2.2 Enumeración Exhaustiva
- 5.2.3 Enumeración Implícita: B&B

5.3 Bifurcación y Acotación

- 5.3.1 Solución Gráfica
- 5.3.2 Solución Algebraica
- 5.3.3 Estrategias de Bifurcación
- 5.3.4 Estrategias de Acotación
- 5.3.5 Estrategias de Poda

5.4 Desarrollos recientes en programación entera

5.1 Introducción

- Muchos de los problemas reales exigen soluciones con valores enteros, por lo tanto las variables de dicho problema deben ser definidas como variables enteras

Hipótesis	Programación Lineal	Programación Entera
H1: Divisibilidad	**SI**	**NO**
H2: No negatividad	**SI**	**SI**
H3: Linealidad	**SI**	**SI**
H4: Certidumbre	**SI**	**SI**

5.1 Introducción

- Este tipo de problemas se denominan en general Problemas de Programación Lineal Entera (PPLE). En concreto:

A. Si todas las variables del problema son enteras se habla de **PPLE Pura.**
B. Si sólo algunas son enteras y las restantes son continuas se habla de **PPLE Mixta.**
C. Si todas las variables enteras son binarias (0/1) el problema se denomina **PPLE Binaria.**

- En ingeniería los problemas más frecuentes son los PPLE Mixta. Estos problemas proporcionan un marco de modelado flexible y eficiente para formular y resolver muchos problemas de ingeniería.

5.1 Introducción

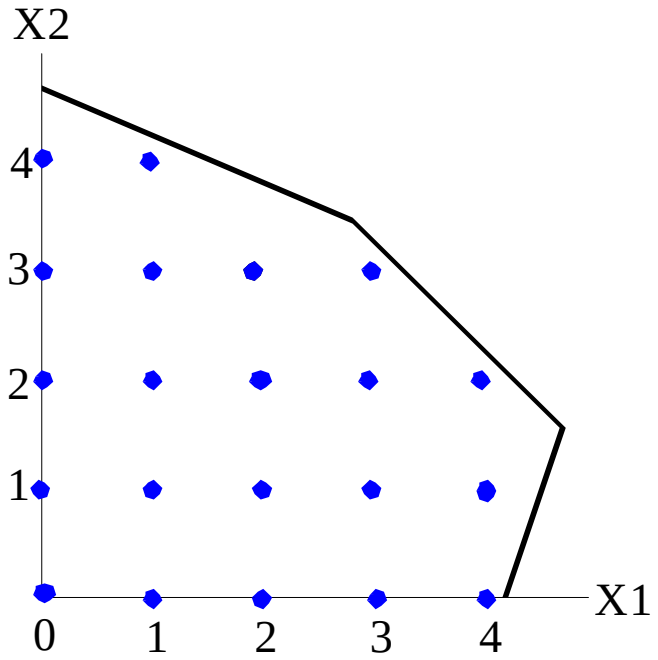
- **Paradójicamente** un modelo de **programación entera** pura o entera binaria tiene un número finito de soluciones mientras que un modelo de **programación lineal** tiene (en general) infinitas soluciones al modelo

¡¡¡ MIL MILLONES DE SOLUCIONES

ES UN NÚMERO FINITO !!!

5.1 Introducción

- **Un modelo de programación entera NO cumple la propiedad de convexidad:**



5.1 Introducción

- El **departamento de I+D** de una gran empresa dispone de 2.5 millones de € para adquirir un nuevo equipamiento informático y resulta imposible ampliar esta cantidad con fondos provenientes de otras fuentes. Diversos estudios realizados indican que sólo dos tipos de máquinas son adecuados y que cualquier número o combinación de ellas sería aceptable. Se han realizado unas pruebas para evaluar la capacidad de carga en unidades de "**trabajos promedio**" por hora para los dos tipos de máquinas

5.1 Introducción

Máquina	Coste (millones de €)	Capacidad (por hora)
1	1.4	28 trabajos
2	0.6	11 trabajos

- El **objetivo** del departamento de I+D es **maximizar la capacidad potencial de trabajo**.

  *Es evidente que **las máquinas sólo pueden adquirirse en unidades enteras***

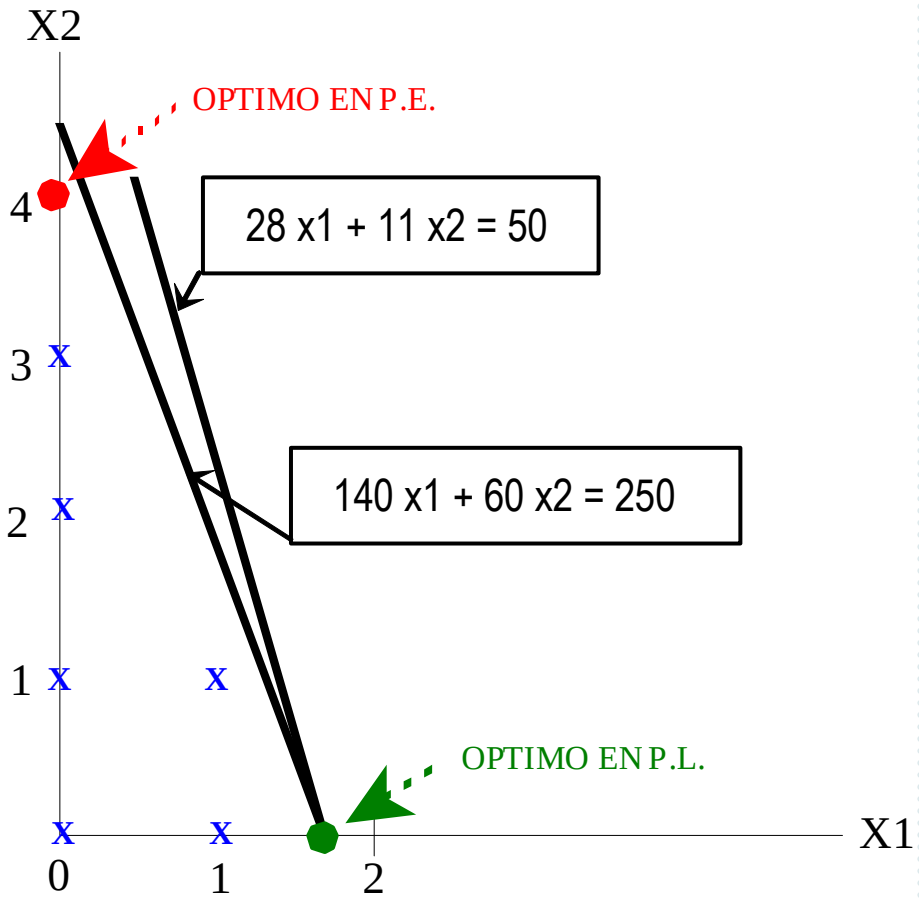
5.1 Introducción

- **VARIABLES:**
  - X_1 : número de máquinas de tipo 1
  - X_2 : número de máquinas de tipo 2
- **FUNCIÓN OBJETIVO:**
  - Maximizar la capacidad de trabajos por hora (en trabajos promedio)

    $\text{Max } Z = 28\ X_1 + 11\ X_2$
- **RESTRICCIONES:**
  - Recursos disponibles: $14\ X_1 + 6\ X_2 \leq 25$
  - Condiciones de no negatividad: $X_1, X_2 \geq 0$
  - Condición de enteros: **X_1, X_2 enteros**

6.2 Un problema entero sencillo

- REGIÓN FACTIBLE Y SOLUCIÓN ÓPTIMA:



- **Solución óptima continua**:

  X_1=1.78, X_2=0 y **Z=50**

- **Redondear al entero más cercano:**

  X_1=2, X_2=0 y Z=56

  (**Solución no factible**)

- **Solución entera factible más próxima:**

  X_1=1, X_2=0 y **Z=28**

- **Solución entera óptima:**

  X_1=0, X_2=4 y Z=44

5.1 Introducción

- **Un PPLE Mixta** general se formula en forma estándar como:

$$\begin{aligned} \textit{Minimizar} \quad & z = \sum_{j=1}^{n} c_j x_j \\ \textit{sujeto a} \quad & \sum_{j=1}^{n} a_{ij} x_j = b_i \quad ; \quad i = 1, 2, \ldots, m \\ & x_j \geq 0 \quad ; \quad j = 1, 2, \ldots, n \\ & x_j \in \mathbb{N} \quad ; \quad j = 1, 2, \ldots, I \quad ; \quad I \leq n \end{aligned}$$

- Los problemas de PPLE se clasifican en las siguientes clases:
  1. **Muy Fáciles**: Resolubles con algoritmos específicos.
  2. **Fáciles**: Resolubles mediante algoritmos voraces (greedy)
  3. **Muy Difíciles**: el número de pasos requeridos para encontrar la solución óptima en el peor caso crece exponencialmente con el tamaño del problema. (Enumeración Exhaustiva, B&B, Planos de Corte)

5.2 Técnicas de Programación Entera

CLASIFICACIÓN DE LOS PROBLEMAS DE PE:

- **Muy Fáciles**: Matriz Unimodular (toda submatriz cuadrada tiene determinante **1**, **0** ó -**1**) → directamente resolubles por PL. La mayoría de los modelos que presentan esta propiedad se ajustan a la definición de red pura. Su estructura especial ha llevado a desarrollar algoritmos específicos que se han demostrado mucho más rápidos que el método simplex general o las aproximaciones de punto interior. *Ejemplos:* Problema de Transporte, asignación, flujo con coste mínimo.

- **Fáciles:** Resolubles mediante algoritmos Greedy (voraces).

  Ejemplos: Árbol de expansión mínima, mochila con pesos iguales y numerosos problemas de secuenciación con una sola máquina.

- **Muy Difíciles:** Están entre los problemas de optimización más difíciles de resolver. Para ellos el número de pasos requeridos para encontrar la solución óptima en el peor caso crece exponencialmente con el tamaño del problema.

5.2 Técnicas de Programación Entera

- En este tema se proporciona una introducción a los principales algoritmos para obtener la solución a este tipo de problemas. En concreto, se estudian los métodos siguientes:

1. *Algoritmos greedy.*

2. *Método de Bifurcación y Acotación (B&B): La solución al PPLE original se obtiene resolviendo una secuencia ordenada de PPLs que se* obtienen relajando las restricciones de integralidad y añadiendo restricciones adicionales.

5.2.1 Algoritmos voraces (greedy)

- Los algoritmos voraces tienen las siguientes características:

1. Construyen una solución de forma iterativa:
   - En cada etapa se aumenta la solución parcial con el objetivo de maximizar la mejora inmediata.
2. La selección realizada en cada etapa no cambia en etapas posteriores (no hay backtracking)
3. El número de etapas es una función polinomial del tamaño del problema.

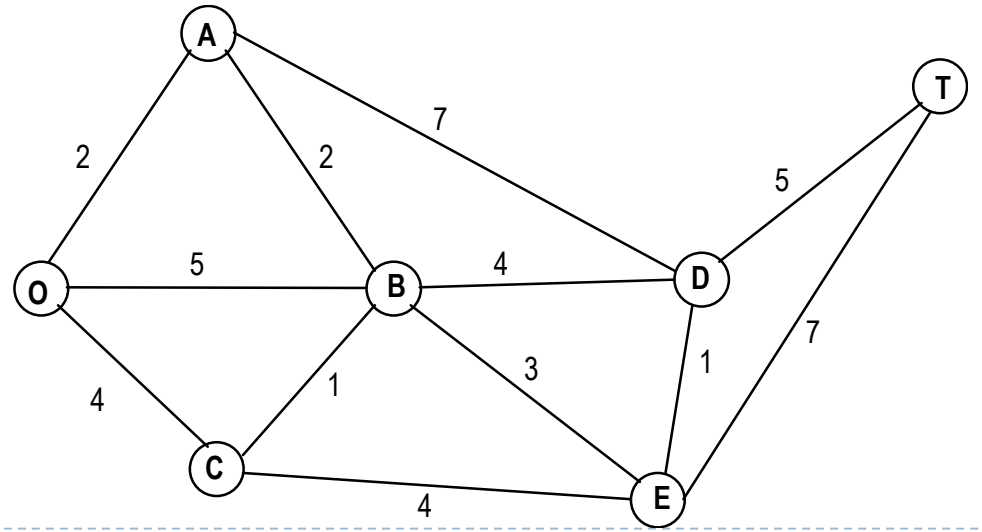
- Los algoritmos voraces son miopes en el sentido de que no miran hacia adelante. Como consecuencia de esto, habitualmente incurren en costes de oportunidad elevados por tomar buenas decisiones en las primeras etapas lo que al mismo tiempo en las etapas finales limita las posibilidades de elección a las mejores de un conjunto de malas alternativas. Ejemplo: Arbol de expansión mínima

5.2.1.1 El Arbol de Expansión Mínima

- Supongamos que se desea construir una red de transmisión de datos conectada a un ordenador central donde los nodos representan terminales de entrada y de transbordo y las aristas redes de comunicación bilaterales entre los diferentes nodos (terminales u ordenador central).

Si el coste de la red está determinado por el coste de las líneas de comunicación (aristas), se pretende determinar la mínima longitud total de la línea que alcance todos los terminales.

A este problema se le denomina **árbol de expansión mínima**.



5.2.1.1 El Arbol de Expansión Mínima

- Cuando las longitudes de los arcos son valores no negativos como los que se asumen en este caso, la selección de los arcos forma un árbol de expansión. Esta característica da lugar al nombre de árbol de expansión mínima.
- El problema puede resolverse con el siguiente algoritmo voraz(greedy):

PASO 1: (Inicialización) Sea *m* el número de nodos en el grafo y sean S1 y S2 dos conjuntos disjuntos. Seleccionar arbitrariamente un nodo i y añadirlo al conjunto S1. Añadir los m-**1** nodos restantes al conjunto S2.

PASO 2: (Selección) De entre todos los arcos con un extremo en S1 y el otro en S2, escoger el arco con menor longitud. Llamar a ese arco (i,j), donde $i \in$ S1 y $j \in$ S2.

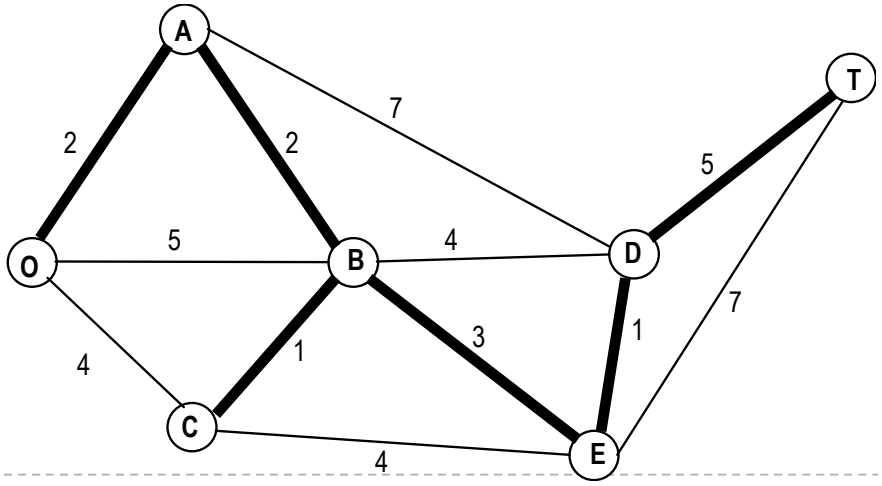
PASO 3: (Construcción) Añadir el arco (i,j) al árbol de expansión. Eliminar el nodo j de S2 y añadirlo al conjunto S1. Si S2=$\varnothing$, STOP, en otro caso, volver a PASO 2.

5.2.1.1 El Arbol de Expansión Mínima

S_I	Arco elegido	Longitud	TOTAL
{O}	(O,A)	2	-
{O,A}	(A,B)	2	2
{O,A,B}	(B,C)	1	4
{O,A,B,C}	(C,E)	3	5
{O,A,B,C,E}	(E,D)	1	8
{O,A,B,C,E,D}	(D,T)	5	9
{O,A,B,C,E,D,T}	-	-	**14**

El conjunto S2=∅, por tanto hemos encontrado la solución óptima.

La longitud total es 14.



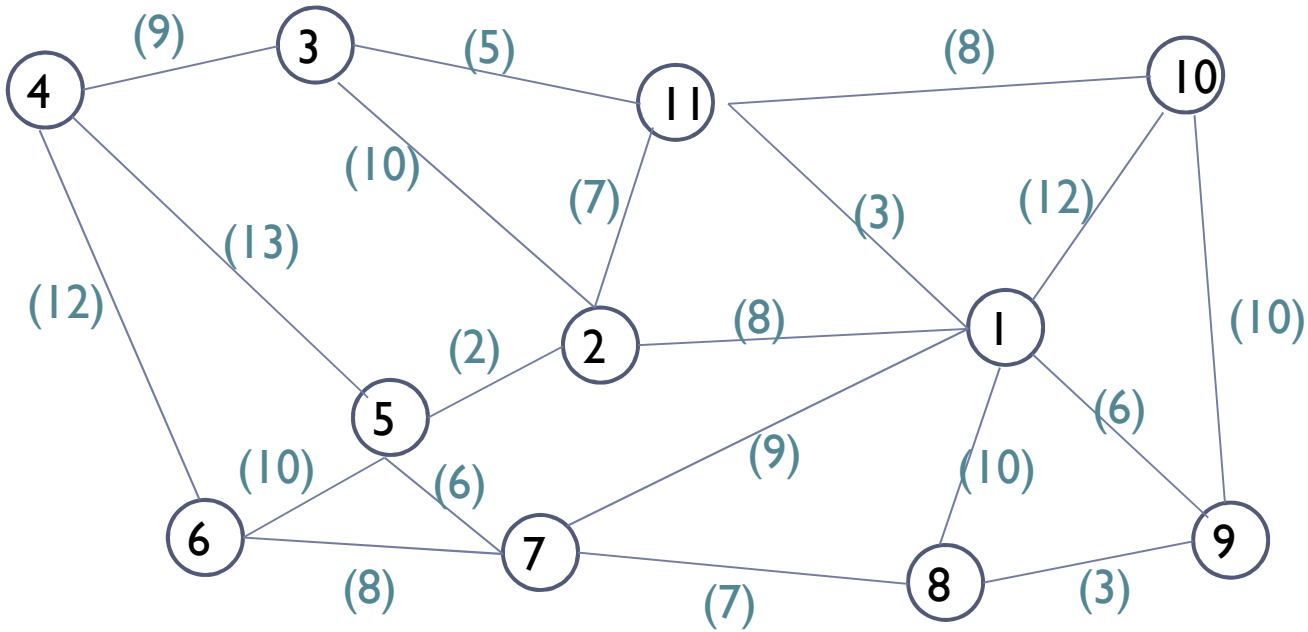
5.2.1.1 El Arbol de Expansión Mínima

- La selección greedy se hace en el PASO 2 al identificar el vértice más corto de todos los restantes candidatos y añadirlo al árbol.
- Esta es una política miope porque elige una componente de la solución sin tener en cuenta su efecto en pasos siguientes.
- Este algoritmo es uno de los más simples en teoría de optimización y su complejidad es $O(m^2)$. La solución óptima se encuentra después de ejecutar el paso de selección m-**1** veces, donde como mucho, m-**1** arcos han de chequearse en cada iteración.
- El problema del árbol de expansión mínima también se resuelve con una algoritmo polinomial en el caso de una red dirigida pero en ese caso la solución no se obtiene con un algoritmo greedy.

5.2.1.1 El Arbol de Expansión Mínima

- **EJERCICIO PROPUESTO:**

  Considerar la red no dirigida de la siguiente figura:



El grafo consta de 11 nodos y 20 arcos, donde cada nodo tiene asociada una longitud. El problema es escoger un conjunto de nodos tales que exista un camino entre cada par de nodos y la suma de las longitudes de los arcos sea mínima.

5.2.1.1 El Arbol de Expansión Mínima

SOLUCIÓN:

S_I	Arco elegido	Longitud	TOTAL
{2}	(2,5)	2	-
{2,5}	(5,7)	6	2
{2,5,7}	(7,8)	7	8
{2,5,7,8}	(8,9)	3	15
{2,5,7,8,9}	(9,1)	6	18
{2,5,7,8,9,1}	(1,11)	3	24
{2,5,7,8,9,1,11}	(11,3)	5	27
{2,5,7,8,9,1,11,3}	(7,6)	8	32
{2,5,7,8,9,1,11,3,6}	(11,10)	8	40
{2,5,7,8,9,1,11,3,6,10}	(3,4)	9	48
{2,5,7,8,9,1,11,3,6,10,4}	-	-	**57**

5.2.2 Enumeración Exhaustiva

- Si un problema NO pertenece a la clase de problemas de PE fáciles, su solución es mucho más difícil de obtener.

1. ENUMERACIÓN EXHAUSTIVA: enumerar y evaluar todas las soluciones posibles. Únicamente es aplicable cuando el problema incluye un número reducido de variables discretas.

2. Bifurcación y Acotación(Branch & Bound – B&B)

5.2.2 Enumeración Exhaustiva

- Un problema de **PE binaria** puede expresarse como:

$$\text{Minimizar} \quad z = \sum_{j=1}^{n} c_j x_j$$

$$\text{sujeto a} \quad \sum_{j=1}^{n} a_{ij} x_j = b_i \quad ; \quad i = 1, 2, \ldots, m$$

$$x_j \ \{0,1\} \quad ; \quad j = 1, 2, \ldots, n$$

$$x_j \in \mathbb{N} \quad ; \quad j = 1, 2, \ldots, I \quad ; \quad I \leq n$$

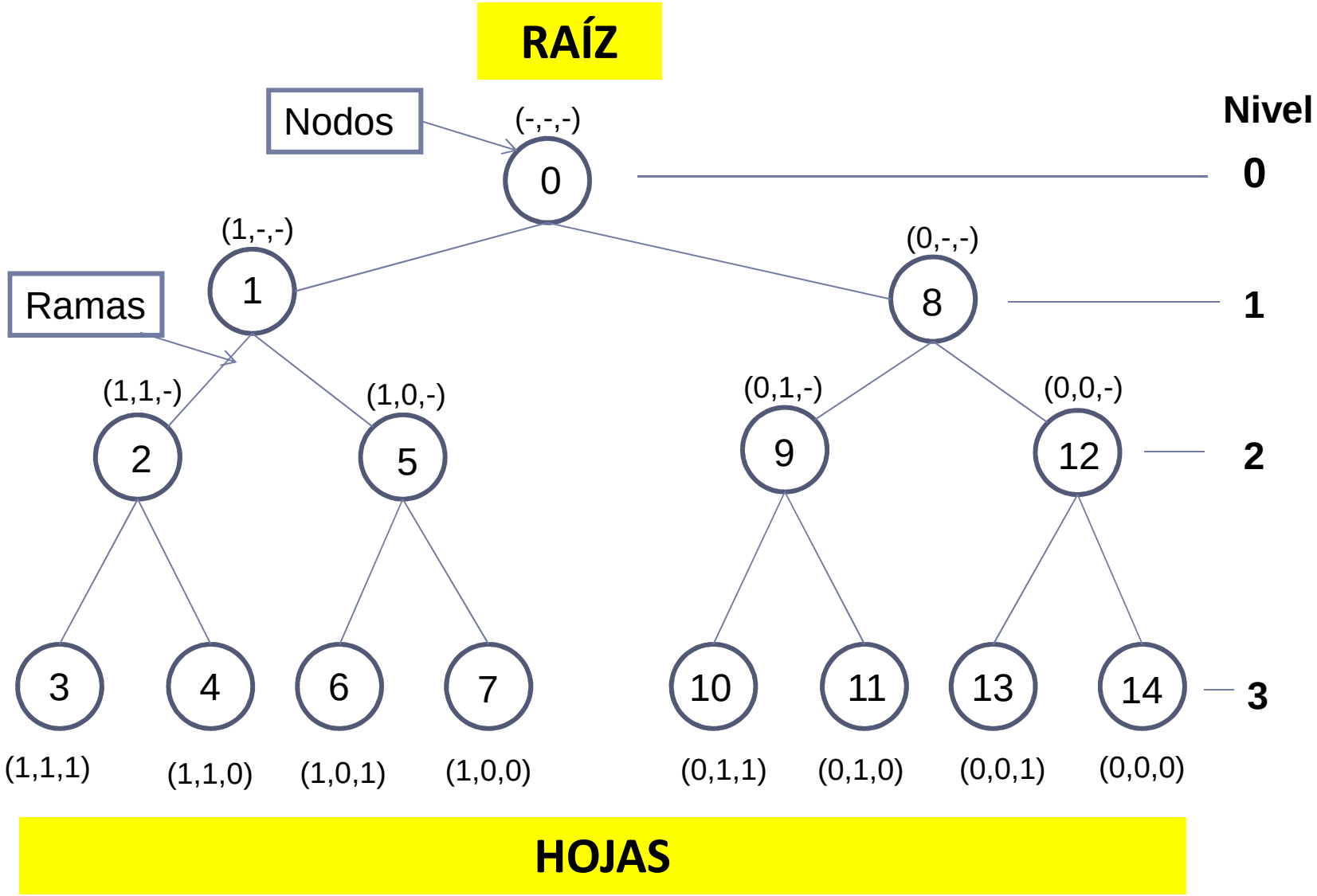
- Este problema puede resolverse enumerando todas las soluciones posibles.
- Esta aproximación puede utilizarse al menos en teoría porque el problema tiene un número finito de soluciones.
- Cada una se genera y, si es factible, su valor de la función objetivo puede compararse con la del resto de candidatos.
- La alternativa con el *menor* valor de *z* (si minimizamos) es la solución óptima.

5.2.2 Enumeración Exhaustiva

- Desafortunadamente el número de soluciones posibles es 2^n, siendo ***n*** el número de variables por lo que evaluar todas las soluciones posibles puede ser inabordable para los ordenadores actuales.

- En cualquier caso la Enumeración Exhaustiva requiere un modo formal de generar las soluciones: **Arbol de Busqueda o Arbol de Enumeración**.

5.2.2 Enumeración Exhaustiva



6.2.2 Enumeración Exhaustiva

- Para cada nodo k, hay un camino P_k que lleva a él desde el nodo 0, que corresponde a una asignación de valores binarios a un subconjunto de variables. Esta asignación se llama *Solución Parcial*.
- En cada nodo que no es hoja, alguna variable llamada *variable separación* se fija a uno de sus valores posibles en el siguiente nivel.
- El conjunto de soluciones en el nodo actual se divide en dos subconjuntos mutuamente excluyentes por esta operación.
- La elección de una variable separación y el paso al siguiente nivel se llama *Ramificación o bifurcación* (Branching) porque este es el modo en el que se forman las ramas del árbol.
- En el proceso enumerativo la estrategia de búsqueda más común es '*depth-first*'. Esto es, en primer lugar creamos un camino directo desde nodo raíz a alguna hoja del árbol y después se hace '*backtracking*' para explorar otros caminos divergentes desde este primero. El número de cada nodo en el árbol de la figura anterior corresponde al orden en el que se han generado las soluciones siguiendo esta estrategia.

5.2.2 Enumeración Exhaustiva

- Para implementar este proceso es necesaria una estructura de datos que informe del estado del árbol de búsqueda en cada punto. Usaremos un vector al que llamaremos P_k con este fin. Para un nodo k en el nivel l del árbol, P_k se define como sigue:

  - La longitud del vector es l y se escribe $P_k=(j_1, j_2,\ldots,j_l)$
  - La magnitud absoluta de j_i es la variable separación en el nivel i.
  - El valor de j_i indica que la variable se fija al valor 0 que se fija al valor 1.
  - La componente j_i puede estar marcada o no. Si está marcada, el nodo alternativo en el nivel i ya ha sido explorado. Si no está marcada, la alternativa todavía no ha sido explorada.
  - Las variables que no aparecen en P_k son las variables libres.

5.2.2 Enumeración Exhaustiva

- En la siguiente tabla se enumeran los P_k vectores para los nodos del árbol ejemplo en el orden en el que se han generado:

Nodo, k	Nivel, l	Pk		Nodo, k	Nivel, l	Pk
0	0	∅				
1	1	(1)		8	1	(0)
2	2	(1,1)		9	2	(0,1)
3	3	(1,1,1)		10	3	(0,1,1)
4	3	(1,1,0)		11	3	(0,1,0)
5	2	(1,0)		12	2	(0,0)
6	3	(1,0,1)		13	3	(0,0,1)
7	3	(1,0, 0)		14	3	(0,0, 1)

5.2.3 Enumeración Implícita: B&B

- Si un problema NO pertenece a la clase de problemas de PE fáciles, su solución es mucho más difícil de obtener.

1. Enumeración Exhaustiva: enumerar y evaluar todas las soluciones posibles. Únicamente es aplicable cuando el problema incluye un número reducido de variables discretas.

2. BIFURCACIÓN Y ACOTACIÓN(Branch & Bound – B&B)

5.3 Bifurcación y Acotación

- **Land** y **Doig** (1960): algoritmos de "**bifurcación y acotación**" o técnicas "**branch and bound**"-**B&B**
- Branch & Bound: Grupo de procedimientos que realizan la enumeración de forma inteligente de modo que no sea necesario examinar todas las combinaciones de valores de las variables.
- Dependiendo de la implementación se utilizan los términos: Enumeración implícita, Arbol de Búsqueda y Partición Estratégica.
- Independientemente del nombre, B&B tiene dos cualidades destacables:
  1. Puede aplicarse esencialmente del mismo modo a problemas de programación lineal entera -PPLE (PE Pura o PE Mixta)
  2. Típicamente obtiene una sucesión de soluciones factibles de modo que si es necesario interrumpir el proceso de búsqueda, la mejor solución hasta el momento puede ser aceptable como una solución aproximada.

5.3 Bifurcación y Acotación

- Comienzan resolviendo el problema original como un PL (i.e. relajando las condiciones de que las variables deben tomar valores discretos)
- Si no se obtiene solución entera, una de las variables enteras que todavía tiene valor continuo, denotada por x_s=a.b se selecciona y se crean dos descendientes del problema original, uno en el que $x_s \geq a+1$ y otro en el que $x_s \leq a$. Esta operación se denomina ***Ramificación (o Bifurcación)***. El proceso se repite seleccionando uno de los problemas como el actual problema de PE y tratándolo exactamente igual que el original. Esto a su vez genera dos nuevos problemas que reemplazan al anterior a menos que por ejemplo el problema en curso no tenga solución posible.
- Repitiendo iterativamente el proceso se alcanza una solución entera (si existe) para uno de los actuales problemas que se convierte en candidata para ser la solución óptima del problema original. La mejor de estas soluciones candidatas es una forma de eliminar descendientes del problema original que es inútil explorar porque no conducirían a la solución óptima. La mejor solución entera candidata se llama ***incumbente*** y el proceso de eliminación se denomina ***Poda***.

5.3.1 Resolución Gráfica

- Para exponer el algoritmo de Bifurcación y Acotación, utilizaremos el siguiente modelo lineal:

  (P_0):

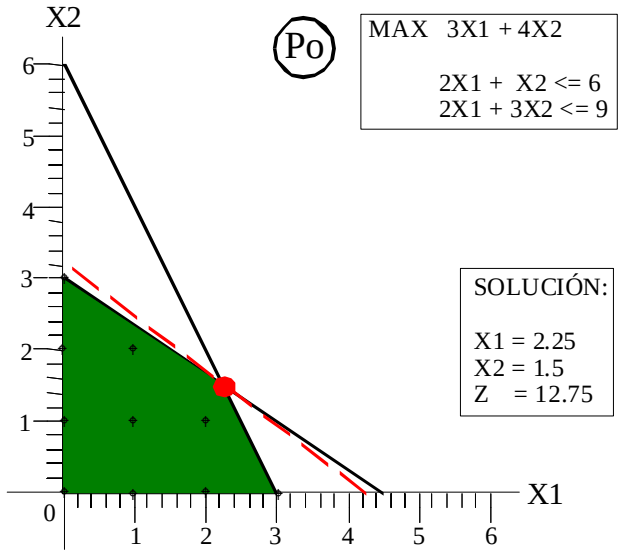
  Max 3 x1 + 4 x2

  s.a:

  $2\ x1 + x2 \leq 6$

  $2\ x1 + 3\ x2 \leq 9$

  $x1, x2 \geq 0$ **y** x1, x2 **enteras**



Resolución Gráfica

- Para exponer el algoritmo de Bifurcación y Acotación, utilizaremos el siguiente modelo lineal:

(P_0):

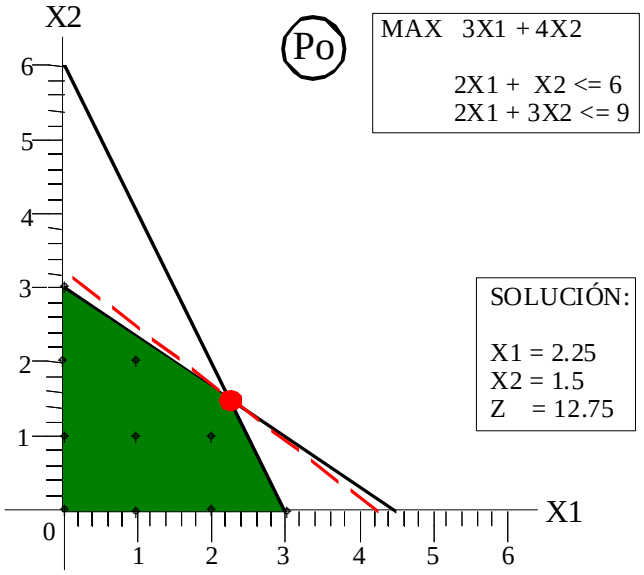
Max 3 x1 + 4 x2

s.a:

$2\,x1 + x2 \leq 6$

$2\,x1 + 3\,x2 \leq 9$

$x1, x2 \geq 0$ **y** x1, x2 **enteras**



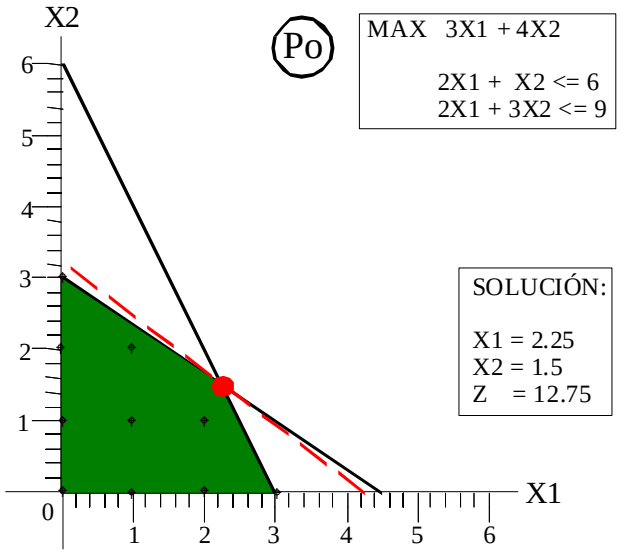
5.3.1 Resolución Gráfica

Cota inferior de la Función Objetivo en la Solución Óptima ENTERA → **$Z^* = -\infty$**

P0

X1=2.25
X2=1.5
Z_0=12.75
$Z^* = -\infty$

5.3.1 Resolución Gráfica



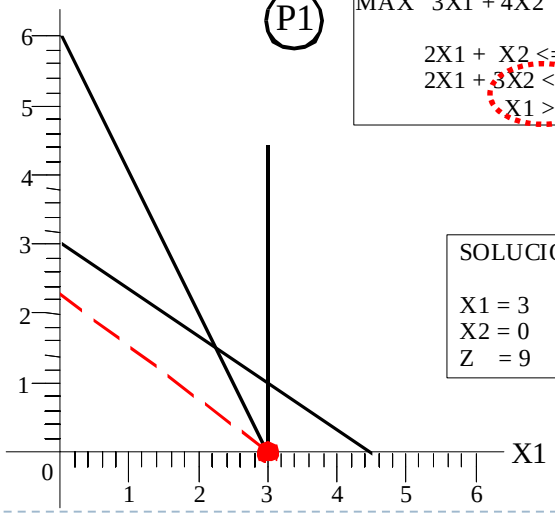
Po

MAX 3X1 + 4X2

2X1 + X2 <= 6
2X1 + 3X2 <= 9

SOLUCIÓN:

X1 = 2.25
X2 = 1.5
Z = 12.75



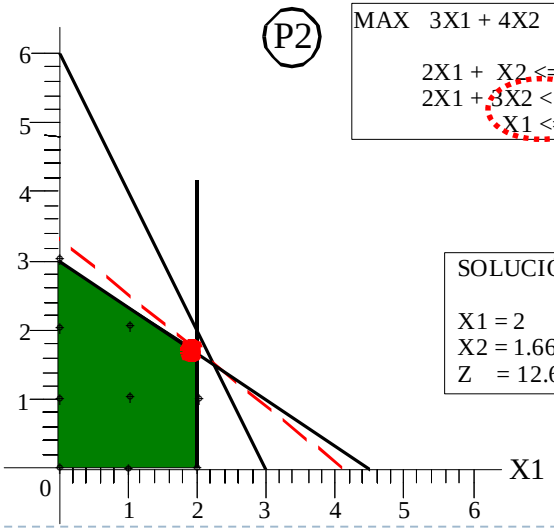
P1

MAX 3X1 + 4X2

2X1 + X2 <= 6
2X1 + 3X2 <= 9
X1 >= 3

SOLUCIÓN:

X1 = 3
X2 = 0
Z = 9



P2

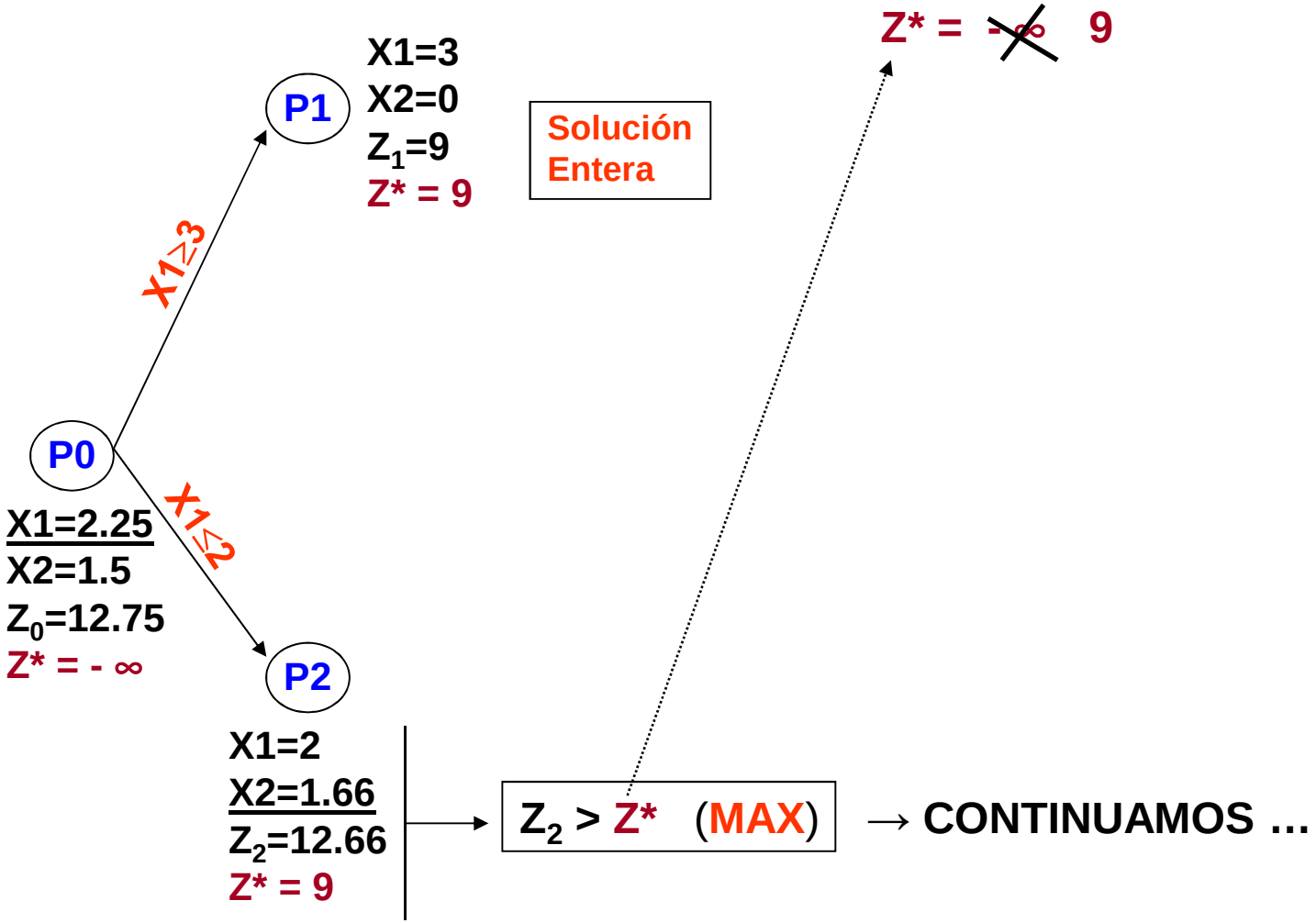
MAX 3X1 + 4X2

2X1 + X2 <= 6
2X1 + 3X2 <= 9
X1 <= 2

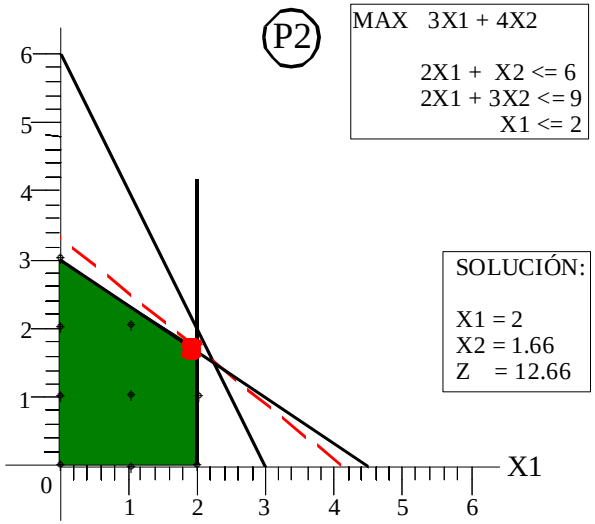
SOLUCIÓN:

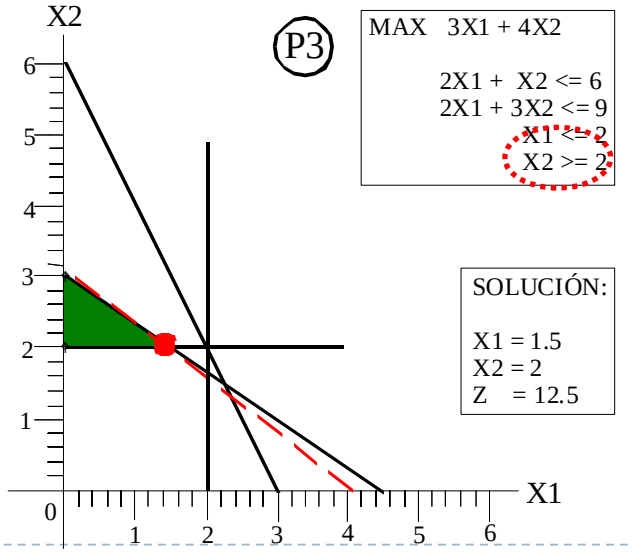
X1 = 2
X2 = 1.66
Z = 12.66

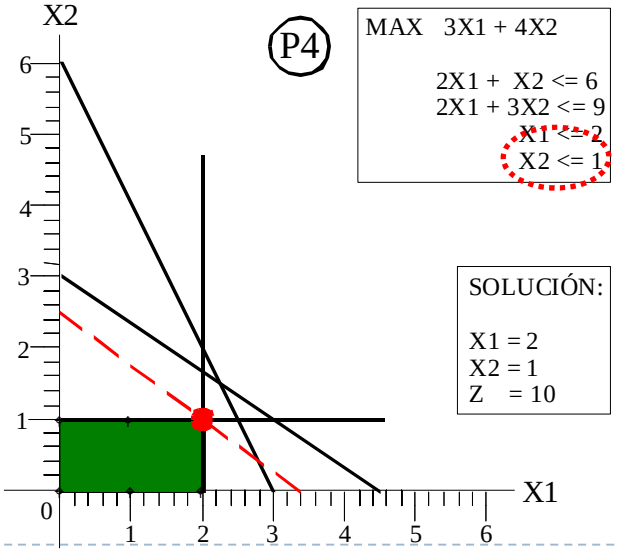
5.3.1 Resolución Gráfica



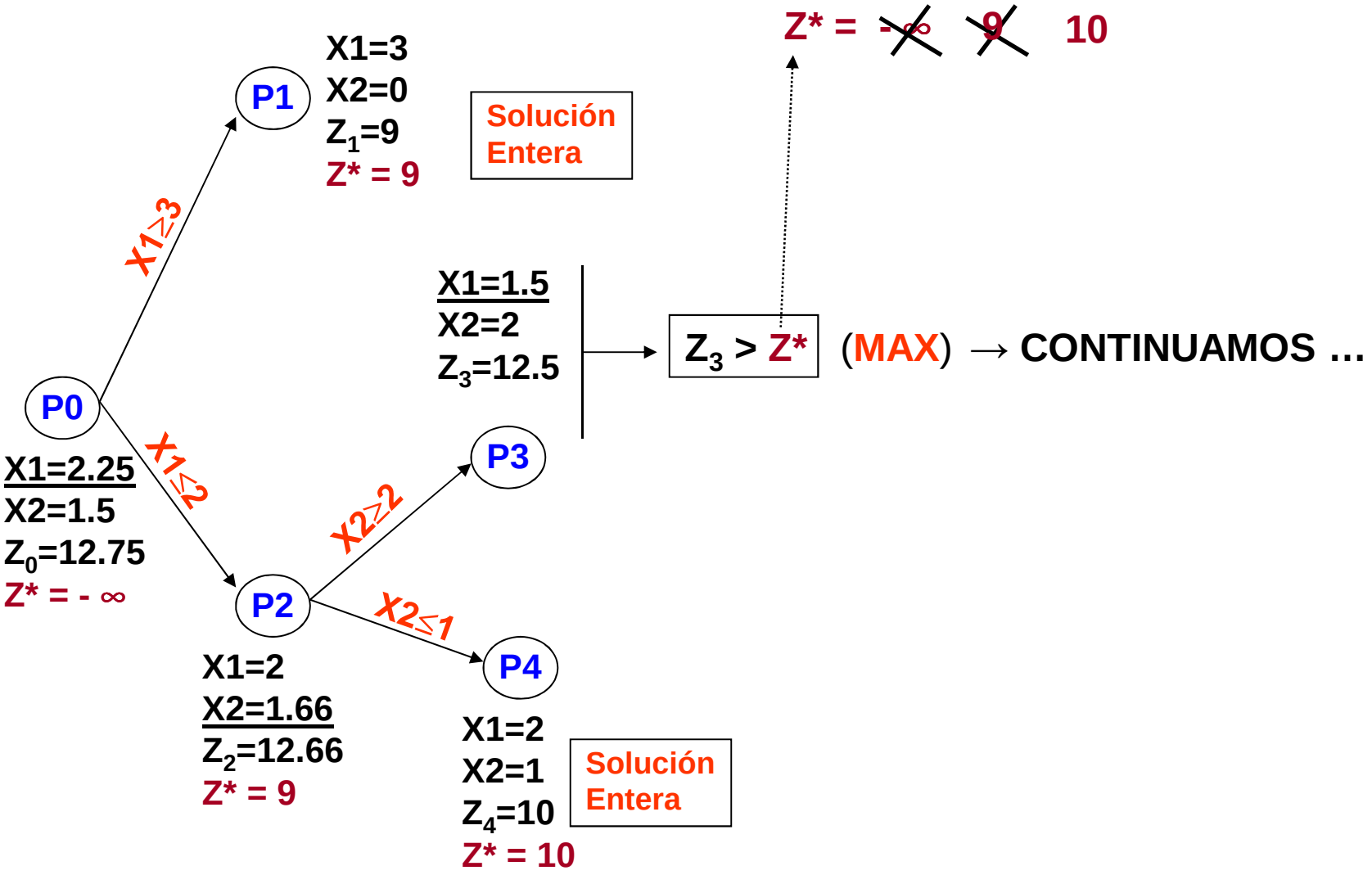
5.3.1 Resolución Gráfica



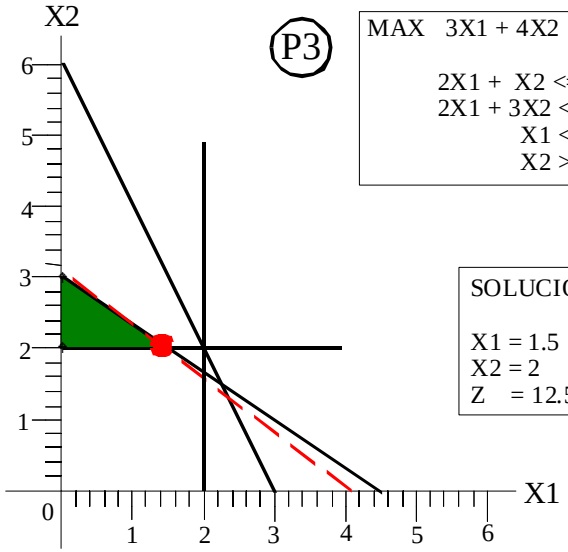




5.3.1 Resolución Gráfica



5.3.1 Resolución Gráfica



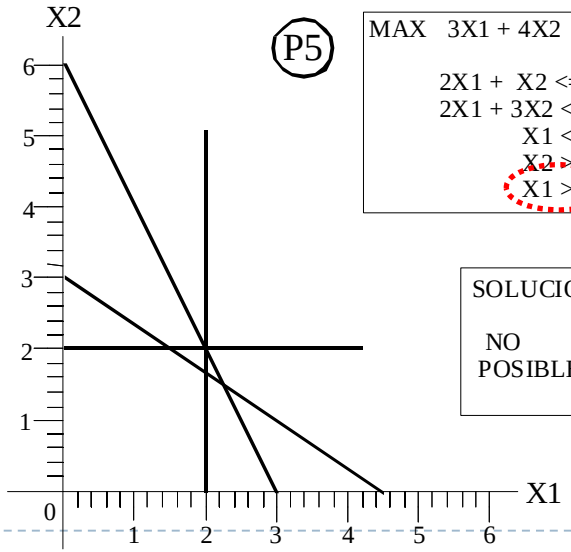
P3

MAX 3X1 + 4X2

2X1 + X2 <= 6
2X1 + 3X2 <= 9
X1 <= 2
X2 >= 2

SOLUCIÓN:

X1 = 1.5
X2 = 2
Z = 12.5



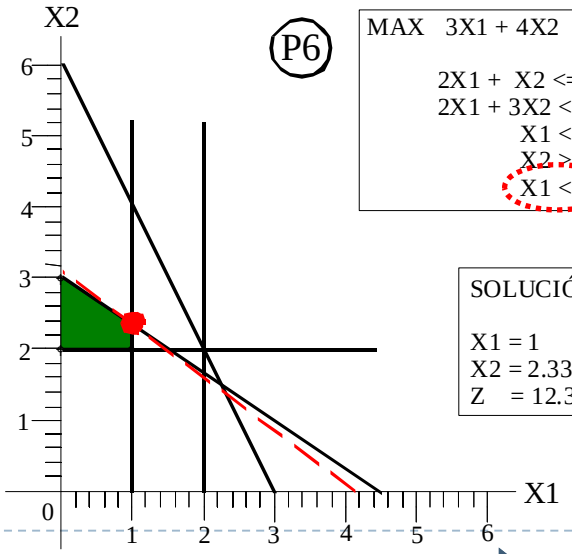
P5

MAX 3X1 + 4X2

2X1 + X2 <= 6
2X1 + 3X2 <= 9
X1 <= 2
X2 >= 2
X1 >= 2

SOLUCIÓN:

NO POSIBLE



P6

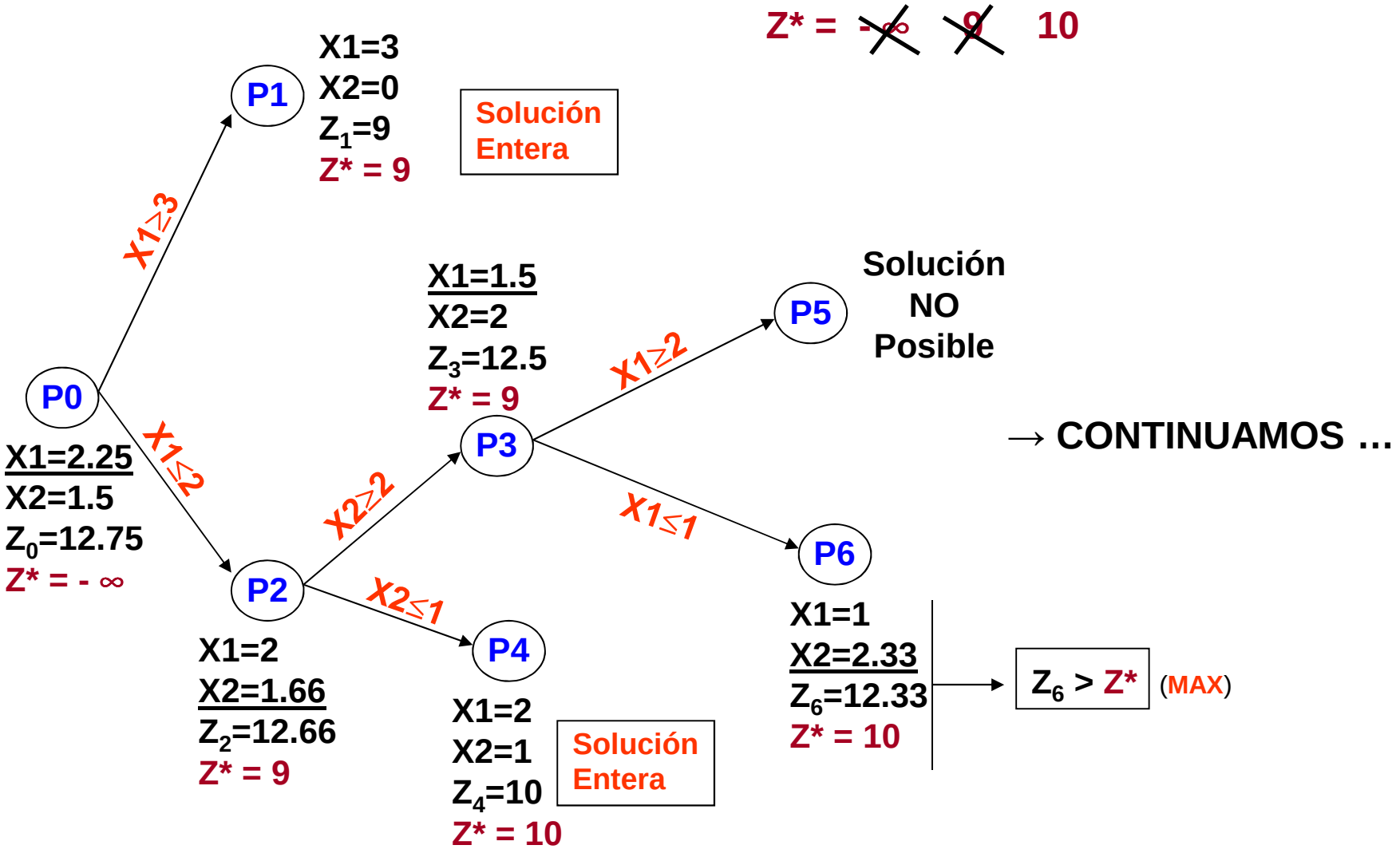
MAX 3X1 + 4X2

2X1 + X2 <= 6
2X1 + 3X2 <= 9
X1 <= 2
X2 >= 2
X1 <= 1

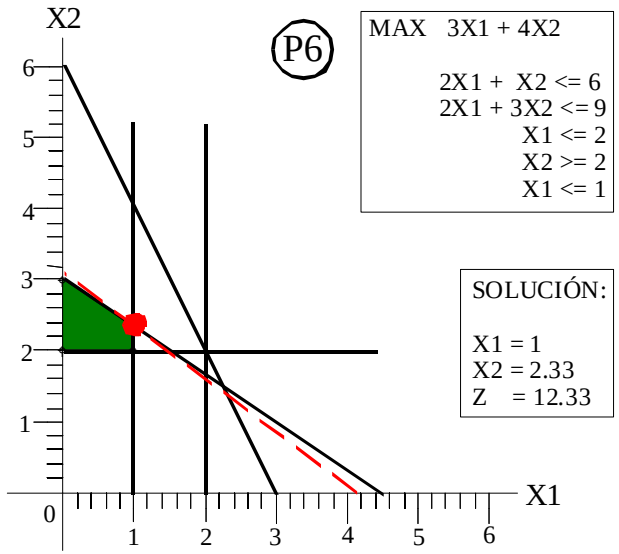
SOLUCIÓN:

X1 = 1
X2 = 2.33
Z = 12.33

5.3.1 Resolución Gráfica



5.3.1 Resolución Gráfica



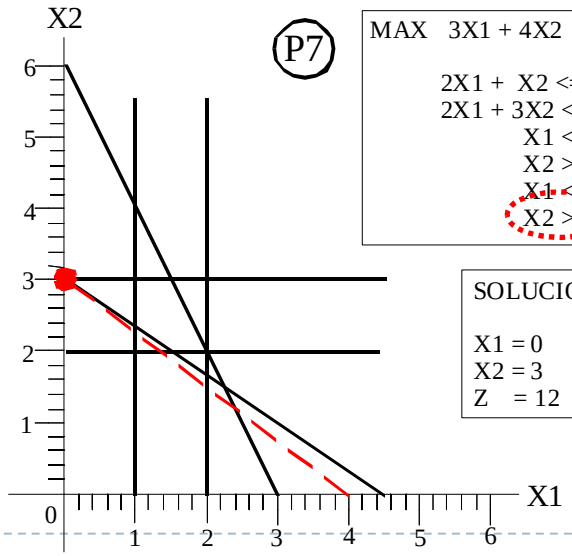
P6

MAX 3X1 + 4X2

2X1 + X2 <= 6
2X1 + 3X2 <= 9
X1 <= 2
X2 >= 2
X1 <= 1

SOLUCIÓN:

X1 = 1
X2 = 2.33
Z = 12.33



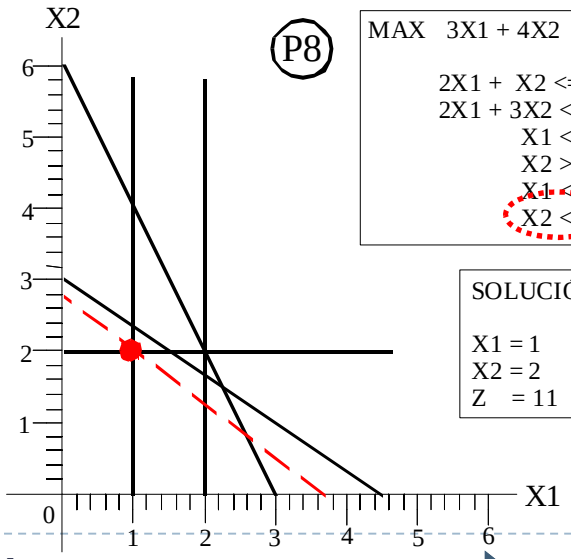
P7

MAX 3X1 + 4X2

2X1 + X2 <= 6
2X1 + 3X2 <= 9
X1 <= 2
X2 >= 2
X1 <= 1
X2 >= 3

SOLUCIÓN:

X1 = 0
X2 = 3
Z = 12



P8

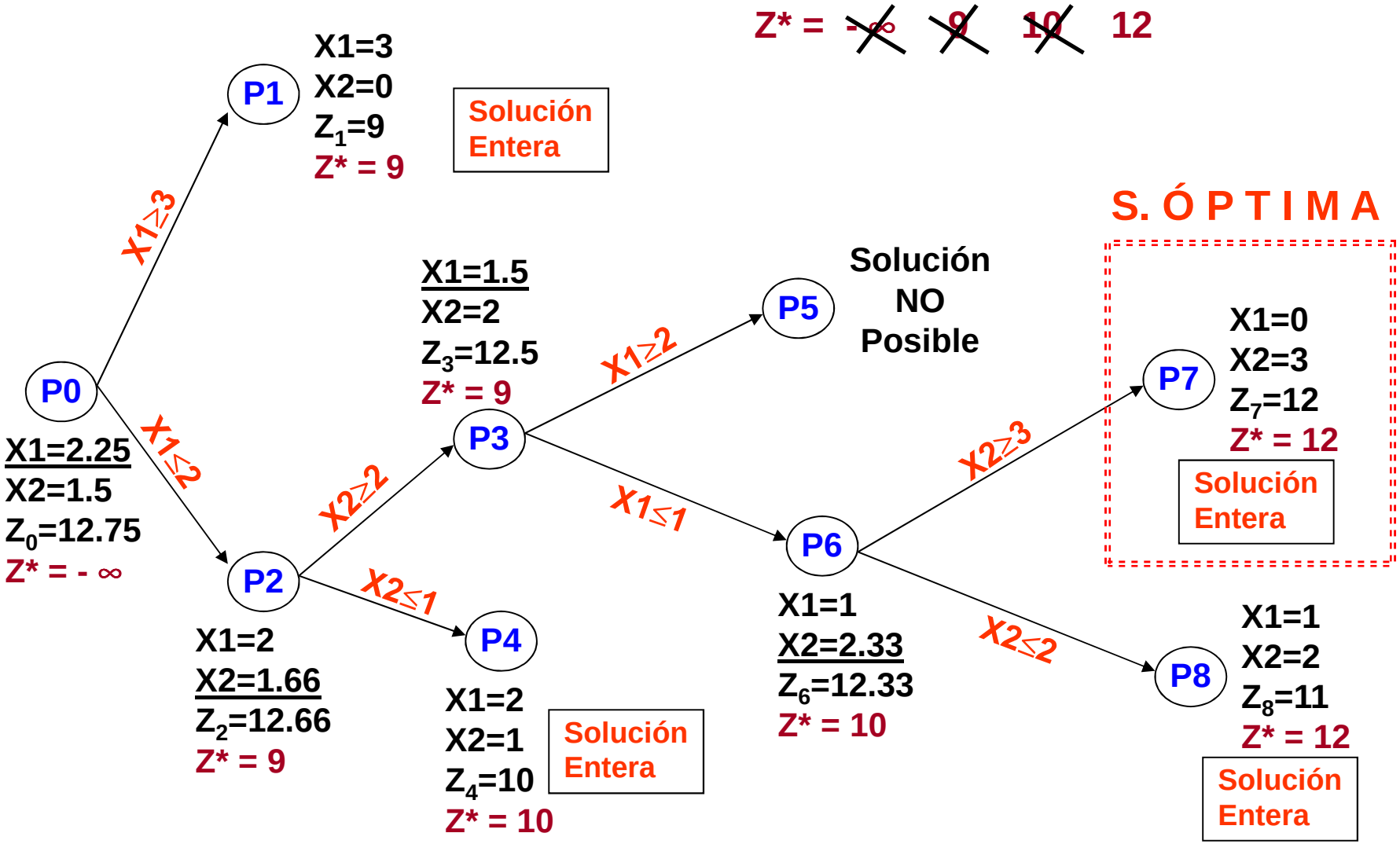
MAX 3X1 + 4X2

2X1 + X2 <= 6
2X1 + 3X2 <= 9
X1 <= 2
X2 >= 2
X1 <= 1
X2 <= 2

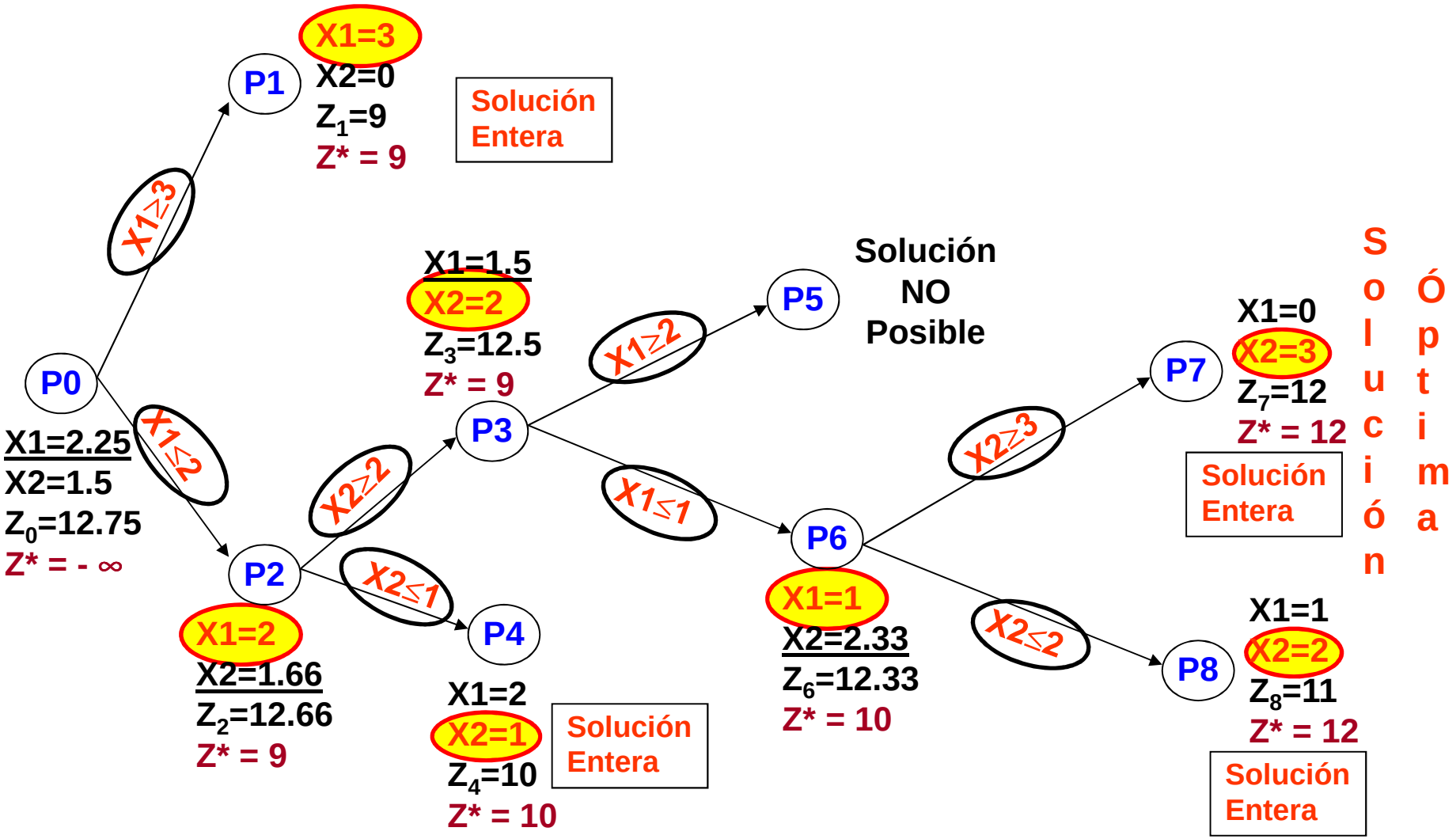
SOLUCIÓN:

X1 = 1
X2 = 2
Z = 11

5.3.1 Resolución Gráfica



5.3.1 Resolución Gráfica



5.3.2 Solución Algebraica

La acotación normalmente se lleva a cabo mediante la ***relajación lineal, consistente en la obtención de la cota a partir de*** la resolución del PPL obtenido relajando las restricciones de integralidad del PPLE original. Vamos a aplicar el modelo ejemplo este método de acotación detallando el proceso algebraico necesario.

(P_0): **Max 3 x1 + 4 x2**

s.a: **2 x1 + x2 ≤ 6**

2 x1 + 3 x2 ≤ 9

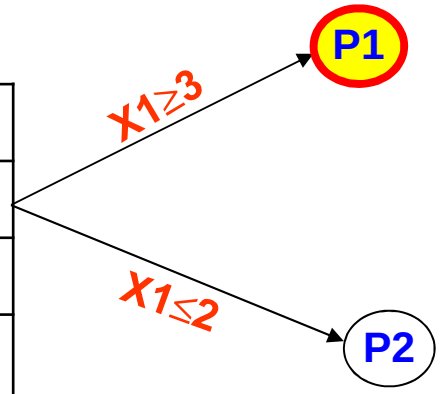
x1, x2 ≥ 0 y enteras

- A partir de la tabla de la solución óptima de **P_0**

Tabla simplex de la solución óptima de **P_0** :

Variables: **x1, x2, x3, x4** **(Z* > -∞)**

v.básicas	B^{-1}		x_B
x1	3/4	-1/4	9/4 = **2.25**
X2	-1/2	1/2	3/2 = 1.5
$c^t_B\ B^{-1}$	1/4	5/4	**Z = 51/4 = 12.75**



5.3.2 Solución Algebraica

P1 = P0(x1,x2,x3,x4) + x1 ≥ 3

- Técnica de la cota inferior **x1 = 3 + l1; l1≥ 0**:
  - **x1** en **P_1** tomará el valor **3**
  - Para que a partir de la solución óptima de P_0, x1 tome el valor 3, es necesario disponer de algún **$\alpha_{x1,j} < 0$** asociado a una **VNB** que permita **incrementar** el valor actual de x1
  - **VNB** en P_0: **x3** y **x4**

$$y_{x3} = \begin{pmatrix} 3/4 \\ -1/2 \end{pmatrix} \quad y_{x4} = \begin{pmatrix} -1/4 \\ 1/2 \end{pmatrix}$$

Pivote

JE

5.3.2 Solución Algebraica

- En P_1 : **VB**: x4 y x2;
  VNB:l1 y x3
- La matriz B^{-1} de P_1 será: $B^{-1}=\begin{pmatrix}-3 & 1\\ 1 & 0\end{pmatrix}$
- Modelo Equivalente: P_0 + (x1=3+l1):

Max 3 x1 + 4 x2	Max 9 + 3 **l1** + 4 x2
2 x1 + x2 ≤ 6	2 **l1** + x2 ≤ 0
2 x1 + 3 x2 ≤ 9	2 **l1** + 3 x2 ≤ 3
x1, x2 ≥ 0 y enteras	

- $x_B = \begin{pmatrix}-3 & 1\\ 1 & 0\end{pmatrix}\begin{pmatrix}0\\ 3\end{pmatrix} = \begin{pmatrix}3\\ 0\end{pmatrix}$
- **Z** = 9 + $(0, 4)\begin{pmatrix}3\\ 0\end{pmatrix}$ = **9**

5.3.2 Solución Algebraica

Tabla simplex de la **solución óptima de P_1**:

Variables: **l1, x2, x3, x4**

v.básicas	B^{-1}		x_B
x4	-3	1	3
x2	1	0	0
$c_B^t\ B^{-1}$	4	0	**Z = 9**

Solución entera $\rightarrow$ Actualizamos **Z*=9**

5.3.2 Solución Algebraica

(P_0):

Max 3 x1 + 4 x2

s.a:

2 x1 + x2 ≤ 6

2 x1 + 3 x2 ≤ 9

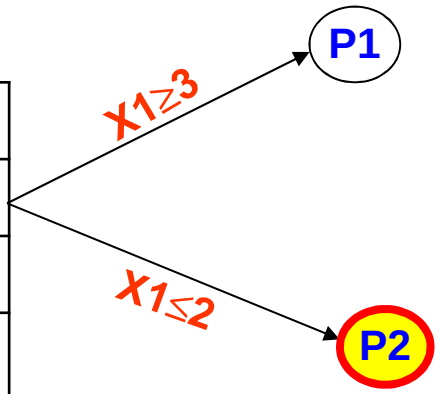
x1, x2 ≥ 0 y enteras

- A partir de la tabla de la solución óptima de **P_0**

Tabla simplex de la solución óptima de **P_0** :

Variables: **x1, x2, x3, x4**

v.básicas	B^{-1}		x_B
x1	3/4	-1/4	9/4 = **2.25**
X2	-1/2	1/2	3/2 = 1.5
$c^t_B\ B^{-1}$	1/4	5/4	**Z = 51/4 = 12.75**



5.3.2 Solución Algebraica

P2 = P0(x1,x2,x3,x4) + x1 ≤ 2

- Técnica de la cota superior **x1 = 2 –u1; 0 ≤ u1 ≤ 2**:
  - **x1** en **P_2** tomará el valor **2**
  - Para que a partir de la solución óptima de P_0, x1 tome el valor 2, es necesario disponer de algún $\alpha_{x1,j} > 0$ asociado a una **VNB** que permita **decrementar** el valor actual de x1
  - **VNB** en **P_0**: **x3** y **x4**

$$y_{x4} = \begin{pmatrix} -1/4 \\ 1/2 \end{pmatrix} \qquad y_{x3} = \begin{pmatrix} 3/4 \\ -1/2 \end{pmatrix}$$

Pivote

JE

5.3.2 Solución Algebraica

- En P_2 : **VB**: x3 y x2;

  VNB: u1 y x4

- La matriz B^{-1} de P_2 será: $B^{-1}=\begin{pmatrix}1 & -1/3\\ 0 & 1/3\end{pmatrix}$

- Modelo Equivalente: P_0 +(x1=2-u1):

  Max 3 x1 + 4 x2
  2 x1 + x2 ≤ 6
  2 x1 + 3 x2 ≤ 9
  x1, x2 ≥ 0 y enteras

  Max 6 - 3 u1 + 4 x2
  -2 u1 + x2 ≤ 2
  -2 u1 + 3 x2 ≤ 5

- $x_B = \begin{pmatrix}1 & -1/3\\ 0 & 1/3\end{pmatrix}\begin{pmatrix}2\\ 5\end{pmatrix} = \begin{pmatrix}1/3\\ 5/3\end{pmatrix}$

- $Z = 6 + (0, 4)\begin{pmatrix}1/3\\ 5/3\end{pmatrix} = 6 + 20/3 = 38/3 = \mathbf{12.66}$

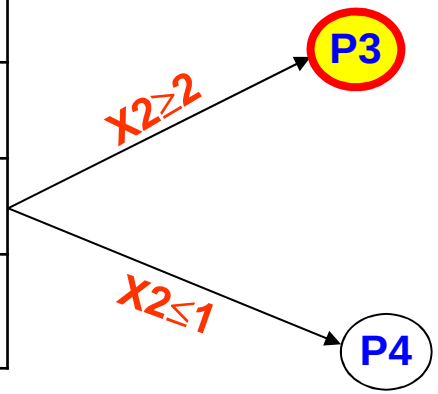
Solución no entera $\rightarrow$ **Z > Z***

5.3.2 Solución Algebraica

Tabla simplex de la **solución óptima de P_2**:

Variables: **u1, x2, x3, x4**

v.básicas	B^{-1}		x_B
x3	1	-1/3	1/3=0.33
x2	0	1/3	5/3=**1.667**
$c_B^t B^{-1}$	0	4/3	**Z = 38/3= 12.66**



5.3.2 Solución Algebraica

P3 = P2(u1,x2,x3,x4) + x2 ≥ 2

- Técnica de la cota inferior **x2 = 2 + l2; l2≥ 0** :
  - **x2** en **P_3** tomará el valor **2**
  - Para que a partir de la solución óptima de P_2, x2 tome el valor 2, es necesario disponer de algún $\alpha_{x2,j} < 0$ asociado a una **VNB** que permita **incrementar** el valor actual de x2
  - **VNB** en **P_2**: **u1** y **x4**

$$y_{x4} = \begin{pmatrix} -1/3 \\ 1/3 \end{pmatrix} \quad y_{u1} = \begin{pmatrix} -4/3 \\ -2/3 \end{pmatrix}$$

Pivote

JE

5.3.2 Solución Algebraica

- En P_3 : **VB**: x3 y u1

  VNB: l2 y x4

- La matriz B^{-1} de P_3 será: $B^{-1}=\begin{pmatrix}1 & -1\\ 0 & -1/2\end{pmatrix}$

- Modelo Equivalente: P_2+ (**x2 = 2 + l2**):

  Max 6 - 3 u1 + 4 x2
  -2 u1 + x2 ≤ 2
  -2 u1 + 3 x2 ≤ 5
  u1, x2 ≥ 0 y enteras

  Max 6 + 8 - 3 u1 + 4 **l2**
  -2 u1 + **l2** ≤ 0
  -2 u1 + 3 **l2** ≤ -1

- $x_B = \begin{pmatrix}1 & -1\\ 0 & -1/2\end{pmatrix}\begin{pmatrix}0\\ -1\end{pmatrix}=\begin{pmatrix}1\\ 1/2\end{pmatrix}$

- **Z** = (6 + 8) + (0, -3) $\begin{pmatrix}1\\ 1/2\end{pmatrix}$ = 28/2 – 3/2 = 25/2 = **12.5**

Solución no entera → **Z** > **Z***

5.3.2 Solución Algebraica

Tabla simplex de la **solución óptima de P_3**:

Variables: u1, l2, x3, x4

v.básicas	B^{-1}		x_B
x3	1	-1	1
u1	0	-1/2	1/2
$c^t_B\ B^{-1}$	0	+3/2	**Z = 25/2= 12.5**

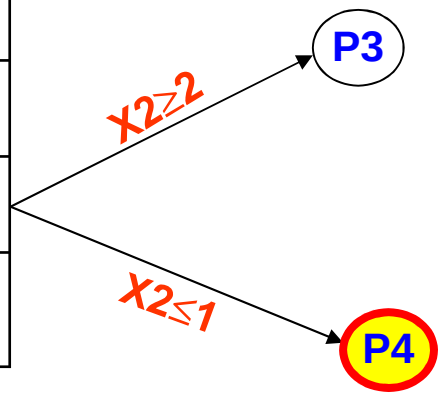
$P_5 = P_3 +$ x1 ≥ 2 y $P_6 = P_3 +$ x1 ≤ 1

5.3.2 Solución Algebraica

Tabla simplex de la **solución óptima de P_2**:

Variables: **u1, x2, x3, x4**

v.básicas	B^{-1}		x_B
x3	1	-1/3	1/3=0.33
x2	0	1/3	5/3=**1.667**
$c_B^t\ B^{-1}$	0	4/3	**Z = 38/3= 12.66**



5.3.2 Solución Algebraica

P4 = P2(u1,x2,x3,x4) + x2 ≤ 1

- Técnica de la cota superior **x2 = 1 –u2; 0 ≤ u2 ≤ 1**:
  - **x2** en **P_4** tomará el valor **1**
  - Para que a partir de la solución óptima de P_2, x2 tome el valor 1, es necesario disponer de algún $\alpha_{x2,j} > 0$ asociado a una **VNB** que permita **decrementar** el valor actual de x2
  - **VNB** en **P_2**: **u1** y **x4**

$$y_{u1} = \begin{pmatrix} -4/3 \\ -2/3 \end{pmatrix} \qquad y_{X4} = \begin{pmatrix} -1/3 \\ 1/3 \end{pmatrix} \leftarrow \text{Pivote}$$

JE

5.3.2 Solución Algebraica

- En P_4 : **VB**: x3 y x4 (con valor 2);
  VNB: u1 y u2
- La matriz B^{-1} de P_4 será: $B^{-1} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$
- Modelo Equivalente : P_2 + (**x2 = 1 –u2**):

Max 6 - 3 u1 + 4 x2	Max 6 + 4 - 3 u1 - 4**u2**
-2 u1 + x2 ≤ 2	-2 u1 – **u2** ≤ 1
-2 u1 + 3 x2 ≤ 5	-2 u1 – 3 **u2** ≤ 2
u1, x2 ≥ 0 y enteras	

- $x_B = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$
- **Z** = (6 + 4) + (0, 0) $\begin{pmatrix} 1 \\ 2 \end{pmatrix}$ = 10 + 0 = **10**

Solución entera y Z > **Z*** → **Z* =10**

5.3.2 Solución Algebraica

Tabla simplex de la **solución óptima de P_4**:

Variables: **u1, u2, x3, x4**

v.básicas	B^{-1}		x_B
x3	1	0	1
x4	0	1	2
$c_B^t\ B^{-1}$	0	0	**Z = 10**

5.3.2 Solución Algebraica

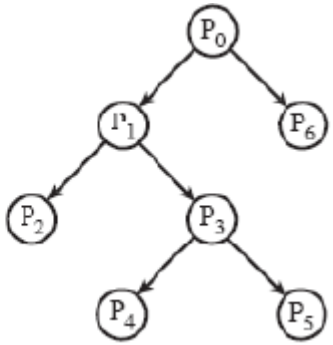
Ejercicio Propuesto:

- **Completar la resolución del problema con el que estamos trabajando y obtener la solución óptima entera**

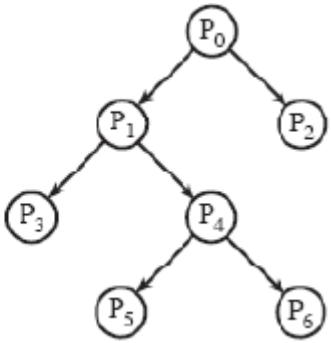
5.3.3 Estrategias de Bifurcación

- Cualquier variable que deba ser entera pero que no lo sea en la solución actual, es una *variable candidata* para bifurcación. Cuál escoger no es una cuestión trivial, y su respuesta ha de basarse en la estructura del problema.
- Los problemas almacenados para ser procesados pueden tratarse mediante estrategias en **profundidad**, en **anchura (mejor cota)** o mixtas.

Búsqueda en profundidad



Búsqueda en anchura



- Normalmente el conocimiento técnico del problema permite establecer el tipo de estrategia a utilizar.

5.3.3 Estrategias de Bifurcación

¿Qué nodo elegir para hacer la siguiente bifurcación?

1. **Técnica de la mejor cota (jumptracking)**

   Consiste en elegir el nodo con mejor valor óptimo continuo, pretendiendo con ello encontrar **buenas soluciones factibles**. Cuando se bifurca un nodo esta aproximación resuelve todos los subproblemas creados. Después el proceso continúa bifurcando de nuevo el nodo con mejor valor de la función objetivo. En problemas grandes, este criterio suele conducir a tener que almacenar muchos datos y presenta **problemas de memoria** de ordenador

5.3.3 Estrategias de Bifurcación

¿Qué nodo elegir para hacer la siguiente bifurcación?

2. **Técnica de la cota más reciente (backtracking)**

   Este criterio consiste en elegir el nodo de creación más reciente. De esta forma se avanza en profundidad en el árbol y, aunque haya que examinar más nodos que con el criterio anterior (no se tiene en cuenta la función objetivo), es más fácil poder eliminarlos durante el proceso. Requiere **poca memoria** de ordenador

5.3.4 Estrategias de Acotación

- Además de la Relajación Lineal utilizada en el apartado anterior, existen otras posibles relajaciones del PPLE original, como la r***elajación Lagrangiana en la que todo el conjunto de*** restricciones (***Ax ≤ b en notación matricial) es eliminado y la función objetivo del problema Maximixar*** $z=c^Tx$ ***es reemplazada por*** *Maximizar* $z_R=c^Tx - \lambda(Ax - b)$, ***donde*** $\lambda \geq 0$ ***es un vector fijo.***

- Si *x* es una solución óptima del problema original* $z \leq z_R$, *por lo que resolviendo la relajación Lagrangiana el valor óptimo de* z_R proporciona una cota válida para el problema original. Escogiendo adecuadamente el valor del vector λ dicha cota tiende a ser similar a la proporcionada por la solución de la relajación lineal, pero con la ventaja de que sin las restricciones del problema, la resolución de la relajación Lagrangiana puede llegar a ser mucho más rápida.

- En contrapartida, la poda llevada a cabo tras la acotación mediante la relajación Lagrangiana no suele ser tan potente como la llevada a cabo tras la relajación lineal. En general, dos son los factores deseables a la hora de escoger una u otra estrategia de acotación: (a) una rápida resolución del problema relajado; y (b) la obtención de una buena cota. En general, la ***relajación lineal suele ofrecer un buen*** compromiso entre ambos factores.

5.3.5 Estrategias de Poda

- La poda de la rama correspondiente tiene lugar por una de las tres razones siguientes:

1. La solución obtenida satisface las condiciones de integralidad (las variables que han de ser enteras, tienen valor entero).
2. El problema considerado NO tiene solución.
3. La solución del problema es continua y el valor de la función objetivo es peor que la mejor cota entera disponible.

(En los 3 casos anteriores, el nodo correspondiente será una **hoja** del árbol de soluciones y por tanto no se bifurcará de nuevo)

Ejercicio Propuesto:

- Dado el siguiente programa lineal:

  MIN 5 x1 + 2 x2

  s.a: 2 x1 + 2 x2 $\geq$ 9

  3 x1 + x2 $\geq$ 11

  x1, x2 $\geq$ 0 y enteras

  y teniendo en cuenta que la solución óptima continua es la que se incluye en la siguiente tabla simplex:

v.básicas	B^{-1}		x_B
x1	-1/4	1/2	13/4
x2	3/4	-1/2	5/4
$c^t_B\ B^{-1}$	1/4	3/2	**Z = 75/4**

Calcula la solución óptima entera mediante el algoritmo de bifurcación y acotación utilizando las siguientes técnicas de selección del nodo a bifurcar:

a) Técnica del nodo de creación más reciente

b) Técnica de la mejor cota

En ambos casos empezar acotando la variable x1 inferiormente (x1≥)

5.4 Desarrollos recientes en programación entera

- Los **modelos de programación entera** son **más difíciles de resolver que los de programación lineal continua** aunque a diferencia de estos últimos los modelos de programación entera tienen muchas menos soluciones que considerar.
  - Tener un número finito de soluciones no asegura que el problema se pueda resolver en un tiempo computacional razonable. Los números finitos pueden ser astronómicamente grandes
  - El hecho de que la solución óptima no se encuentre necesariamente en un punto extremo implica que no se puede utilizar el simplex aunque dada su eficiencia se utiliza tanto como sea posible

5.4 Desarrollos recientes en programación entera

- Existen **problemas** que por su **estructura especial** garantizan **solución entera** aunque no se especifiquen sus variables como tales:

Problema de flujo a coste mínimo
Problema de asignación
Problema de la ruta más corta
Problema de transporte
Problema de transbordo
Problema de flujo máximo

5.4 Desarrollos recientes en programación entera

- La **dificultad computacional** de los modelos de programación lineal entera depende del **número de variables enteras** y muy poco del número de restricciones de modo que es posible que añadir restricciones adicionales facilite la resolución al eliminar soluciones posibles.
- No existe ningún algoritmo para resolver de forma óptima modelos de programación entera cuya **eficiencia** computacional pueda compararse con la del método simplex para programación lineal. Por tanto el desarrollo de algoritmos de programación entera sigue siendo un tema de investigación. Uno de los algoritmos más populares de programación entera es la técnica de **bifurcación y acotación** que hemos visto en apartados anteriores.

5.4 Desarrollos recientes en programación entera

- A finales de la década de los 60 y principios de los 70 se produjo un gran cambio con el desarrollo y refinamiento de las técnicas de bifurcación y acotación. Se podían resolver de forma eficiente modelos relativamente pequeños (por debajo de 100 variables enteras)

- En 1983 y 1985 se presentaron técnicas mejoradas para resolver problemas de programación entera binaria. Al nuevo enfoque algorítmico se le ha dado el nombre de **Bifurcación y Corte** (**Branch and cut**)

5.4 Desarrollos recientes en programación entera

- Los algoritmos de Bifurcación y Corte utilizan una combinación de tres tipos de técnicas:

1. **Preproceso automático del problema**: inspección de la formulación del problema con el fin de detectar algunas reformulaciones que hacen que el problema se resuelva con más rapidez, sin eliminar ninguna solución factible. Estas reformulaciones se pueden clasificar en tres categorías:

   -Fijar variables

   -Eliminar restricciones redundantes

   -Reformular restricciones de modo que sean más restrictivas

5.4 Desarrollos recientes en programación entera

2. **Generación de planos de corte**. Los planos de corte son restricciones funcionales que reducen la región factible del modelo de programación lineal asociado al que estamos resolviendo, sin eliminar soluciones factibles para el problema original. Las técnicas de generación de planos de corte tenderán a acelerar la rapidez con la que el método de bifurcación y acotación puede encontrar una solución óptima para un problema de programación entera

3. **Bifurcación y acotación**

5.4 Desarrollos recientes en programación entera

- Durante los años 90 y siguientes se ha continuado e intensificado las actividades de investigación y desarrollo en programación entera. Se resuelven problemas cada vez más grandes. Por ejemplo al final de la década de los 90 **CPLEX** v6.5 aplicó con éxito un algoritmo de ramificación y corte para resolver un problema con más de 4000 restricciones funcionales y más de 12000 variables. También se está consiguiendo resolver problemas de programación entera mixta con miles de variables enteras generales junto con numerosas variables continuas y variables binarias. Por su parte los programas **LINDO** y **LINGO** incluyen optimizadores para resolver problemas de programación entera. En versiones recientes de LINGO se anunciaba la recodificación completa del módulo de programación entera y la incorporación de opciones adicionales a las versiones anteriores que permiten al usuario ajustar con numerosos parámetros el proceso de búsqueda de la solución.

EJERCICIOS ADICIONALES:

1. Dado el siguiente programa lineal:

MAX Z= 4X1 + 3X2

s.a. $3X1+ 4X2 \leq 12$

$4X1 + 2X2 \leq 9$

$X1, X2 \geq 0$ y **enteras**

Cuya solución óptima continua se muestra en la tabla siguiente:

v.básicas	B^{-1}		x_B
X2	2/5	-3/10	21/10
X1	-1/5	2/5	12/10
$c_B^t B^{-1}$	2/5	7/10	Z=111/10

Obtener la solución óptima entera. Aplicar el algoritmo de **Bifurcación y Acotación** bifurcando en la solución óptima continua la variable X2.Utilizar como **técnica** de **selección del nodo a explorar** la del **nodo de creación más reciente**. En cada nodo empezar acotando superiormente las variables ($\leq$).

SOLUCIÓN: **X1 = 1; X2 = 2; Z = 10**

Ejercicios Adicionales:

2. Dado el siguiente programa lineal:

MIN 2X1 + 3X2

s.a: [R1] X1 + X2 ≥3

[R2] X1 + 3X2 ≥6

X1, X2≥ 0 y enteras

Cuya solución óptima continua se incluye en la tabla siguiente:

v.básicas	B^{-1}		x_B
X1	3/2	-1/2	3/2
X2	-1/2	1/2	3/2
$c_B^t B^{-1}$	3/2	1/2	Z=15/2

Obtener la **solución óptima entera** aplicando el algoritmo de Bifurcación y Acotación. Genera el árbol de soluciones mediante la técnica de la mejor cota y comienza acotando inferiormente (≥) la variable X1.

SOLUCIÓN: **X1 = 1; X2 = 2; Z = 8**

Ejercicios Adicionales:

3. Dado el siguiente programa lineal:

MAX Z= 2X1 + X2

s.a. $X1 - X2 \leq 5$

$4X1 + 3X2 \leq 10$

$X1, X2 \geq 0$ y **enteras**

Cuya solución óptima continua se muestra en la tabla siguiente (X3 es la variable de holgura de la primera restricción):

v.básicas	B^{-1}		x_B
X3	1	-¼	5/2
X1	0	¼	5/2
$c_B^t B^{-1}$	0	½	Z=5

Aplicar el algoritmo de **Bifurcación y Acotación** hasta encontrar **una solución entera.** Utilizar como **técnica** de **bifurcación** la de la **mejor cota.** En cada nodo empezar acotando inferiormente las variables.

La solución obtenida en el apartado anterior, ¿es óptima? Justifica la respuesta.

SOLUCIÓN: **X1 = 2; X2 = 0; Z = 4**