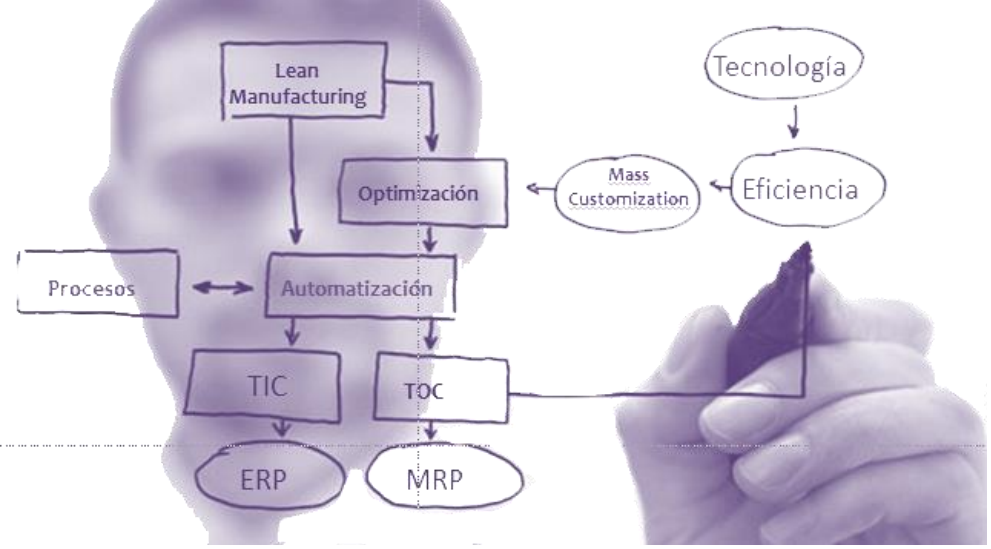


DEIOAC-UPV

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 de los algoritmos para la resolución de problemas de programación lineal entera.
- Interpretar los árboles que se generan al aplicar estos algoritmos así como deducir el criterio aplicado para su obtención

CONTENIDOS

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

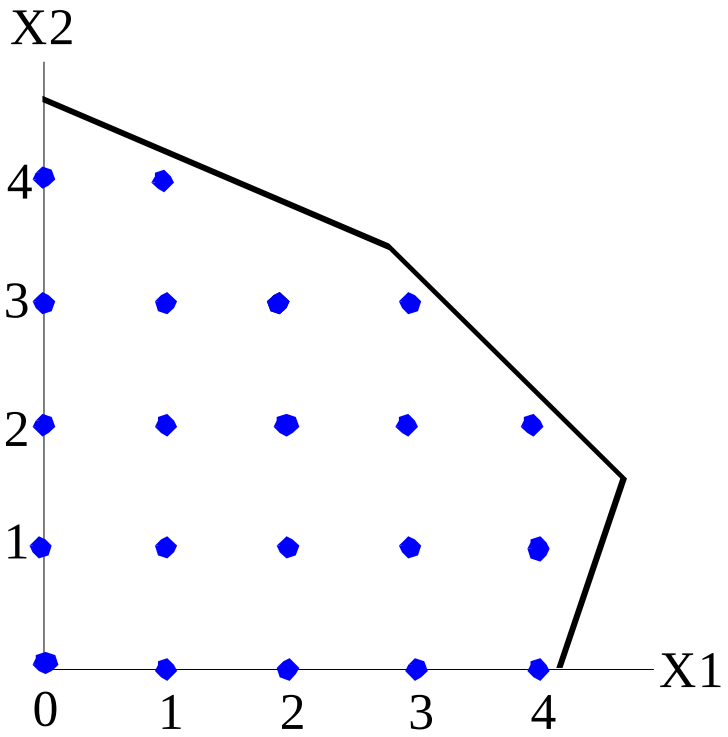
- **Paradójicamente** un modelo de **programación entera** pura o entera binaria tiene un número finito de soluciones mientras que un modelo de **programación lineal** tiene (en general) infinitas soluciones al modelo

¡¡¡ 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

Un problema entero sencillo para desconfiar de redondeos

- 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***

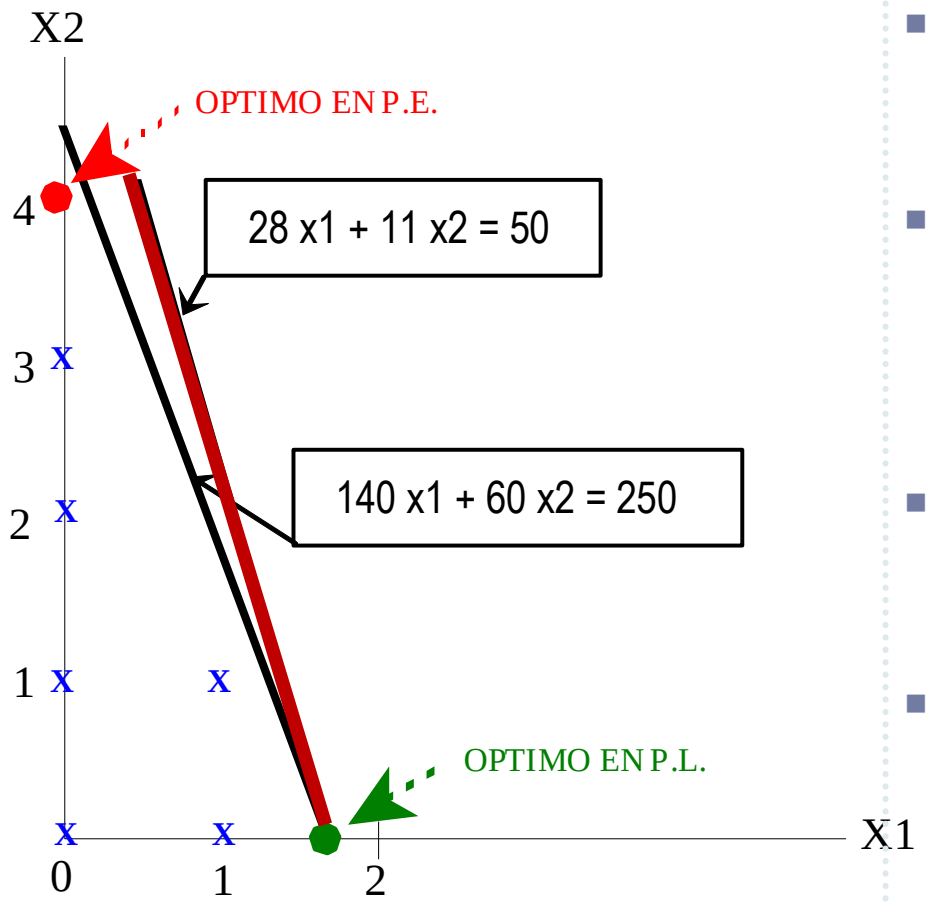
5.1 Introducción

- **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**

5.1 Introducción

- 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

5.2 Técnicas de programación entera: algoritmo de bifurcación y acotación

- Ralph **Gomory** fue el creador del primer algoritmo para resolver modelos de programación entera (1958):

Algoritmos de planos de corte

- El **algoritmo de Gomory** consiste en resolver el problema sin considerar las restricciones de carácter entero de las variables. Si la solución no es entera añade restricciones que reducen el conjunto de soluciones del problema lineal continuo asociado sin excluir ninguna solución entera.

5.2 Técnicas de programación entera: algoritmo de bifurcación y acotación

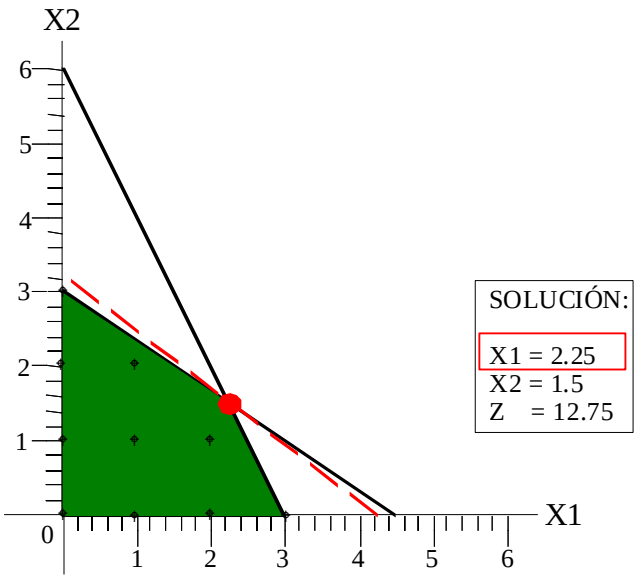
- **Land** y **Doig** (1960): algoritmos de "**bifurcación y acotación**" o técnicas "**branch and bound**"-**B&B**
- Branch & Bound: Grupo de procedimientos que realizan la enumeración de forma inteligente de modo que no sea necesario examinar todas las combinaciones de valores de las variables.
- B&B 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.

5.2 Técnicas de programación entera: algoritmo de bifurcación y acotación

- Las técnicas B&B 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***.

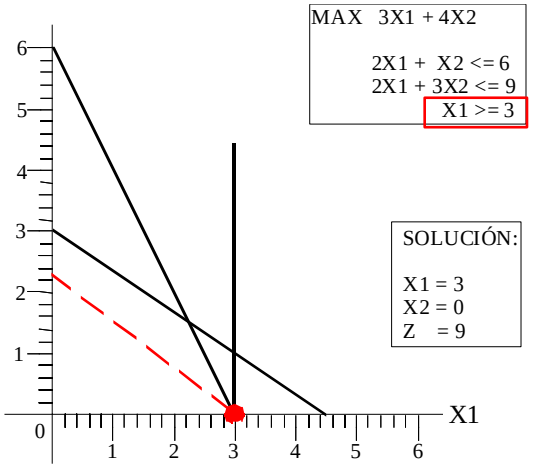
5.2 Técnicas de programación entera: algoritmo de bifurcación y acotación

En la figura se muestra la región factible de un problema de Programación Lineal en el que x1 y x2 han de ser enteras. La solución óptima se ha obtenido sin tener en cuenta dicha condición:

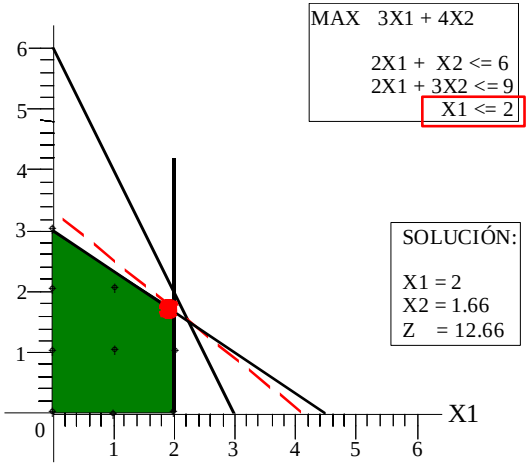


Dado que la Solución Optima requiere que tanto x1 como x2 sean enteras y todavía no lo son, el problema se divide en dos nuevos subproblemas en los que x1 toma valor entero:

❶



❷



El proceso se repite hasta que se alcanza una *hoja* del árbol de soluciones.

5.2 Técnicas de programación entera: algoritmo de bifurcación y acotación

Resolución Gráfica

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

(P_0):

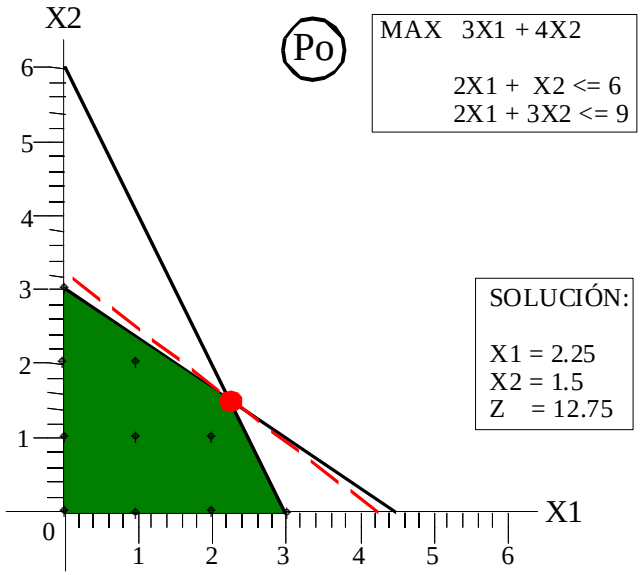
Max 3 x1 + 4 x2

s.a:

$2\ x1 + x2 \leq 6$

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

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



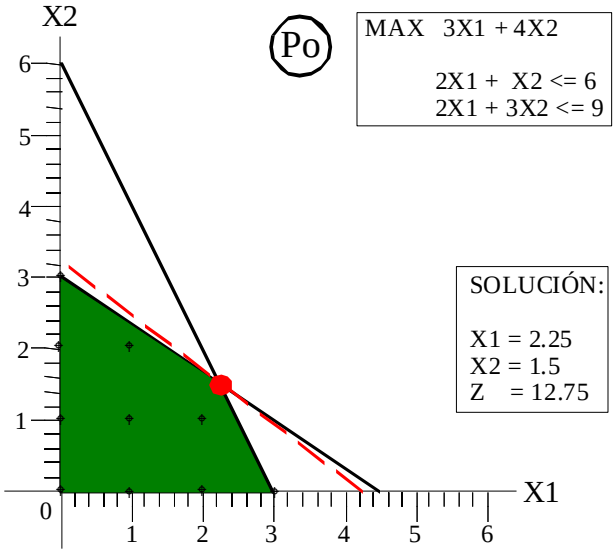
5.2 Técnicas de programación entera: algoritmo de bifurcación y acotación

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

P0

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

5.2 Técnicas de programación entera: algoritmo de bifurcación y acotación



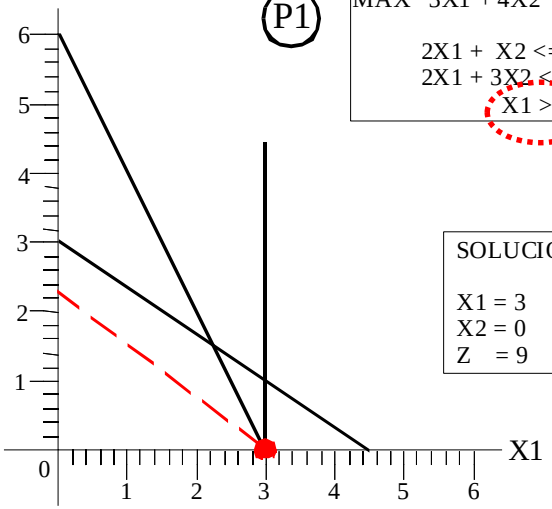
Po

MAX 3X1 + 4X2

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

SOLUCIÓN:

X1 = 2.25
X2 = 1.5
Z = 12.75



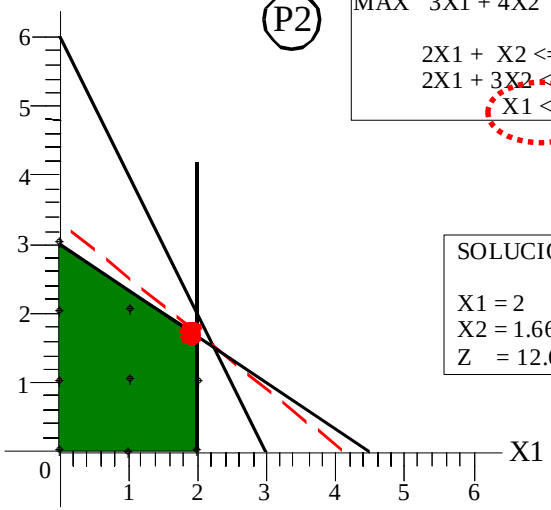
P1

MAX 3X1 + 4X2

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

SOLUCIÓN:

X1 = 3
X2 = 0
Z = 9



P2

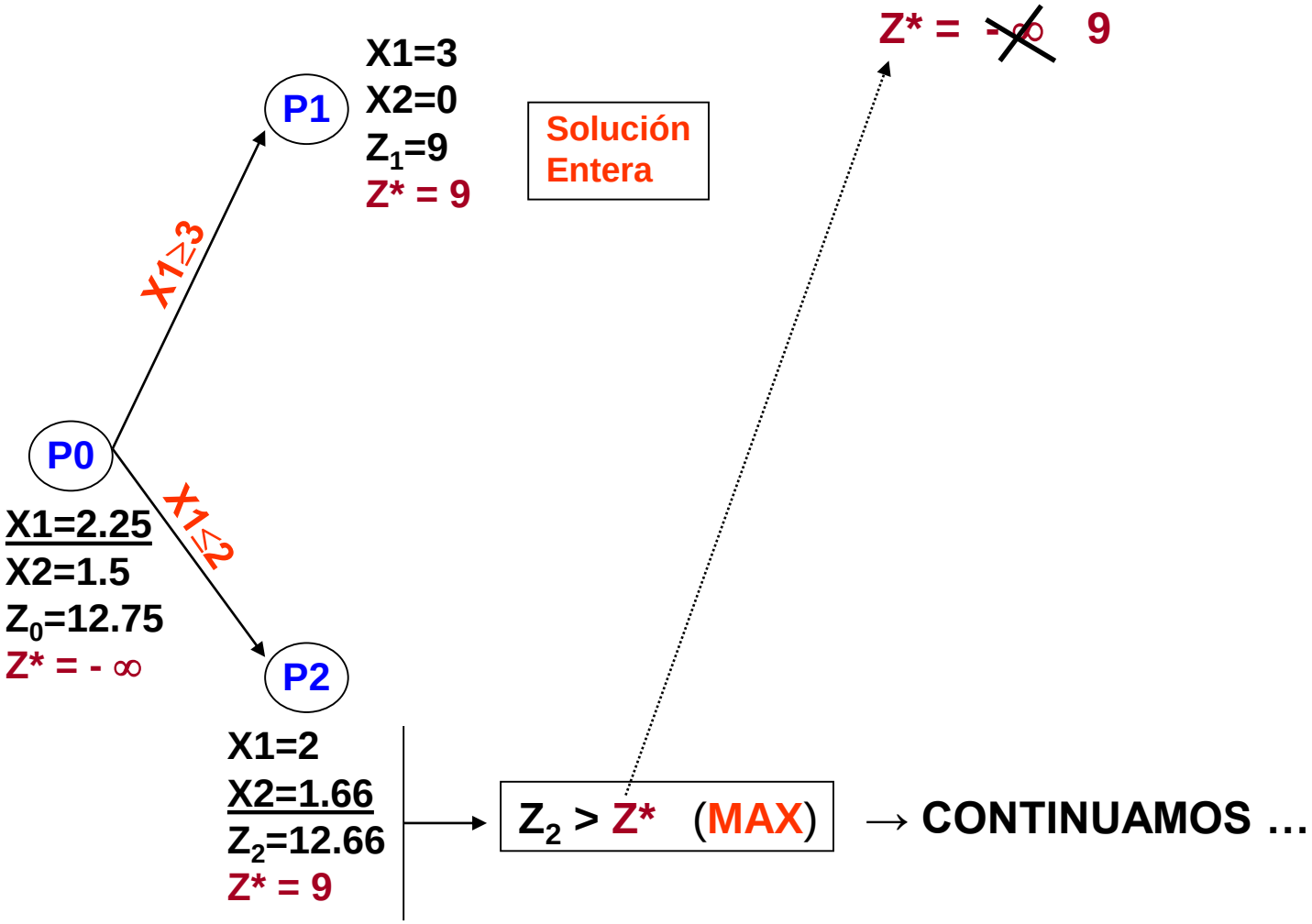
MAX 3X1 + 4X2

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

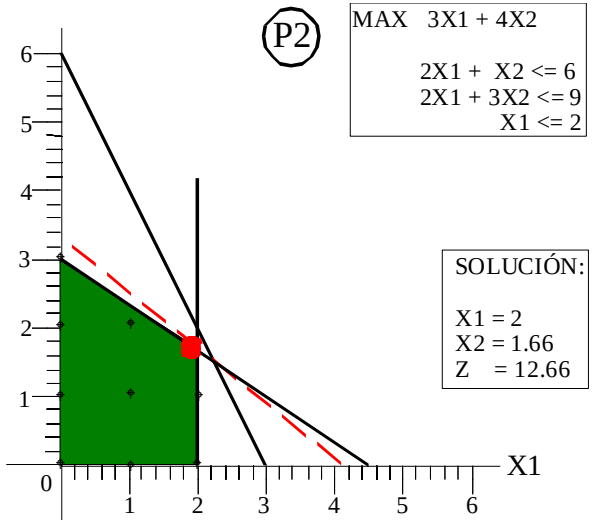
SOLUCIÓN:

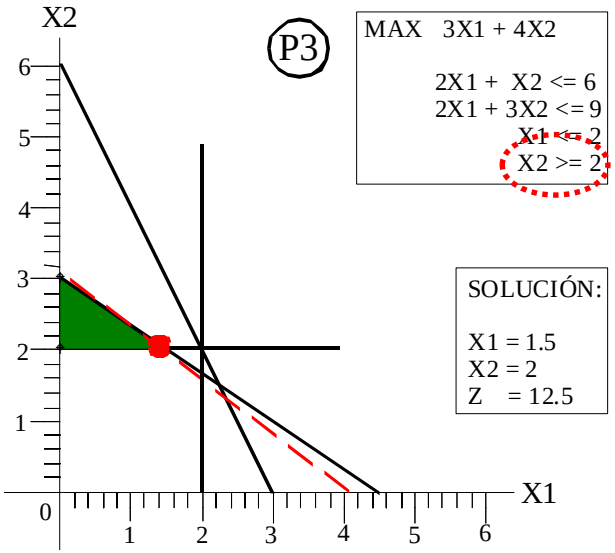
X1 = 2
X2 = 1.66
Z = 12.66

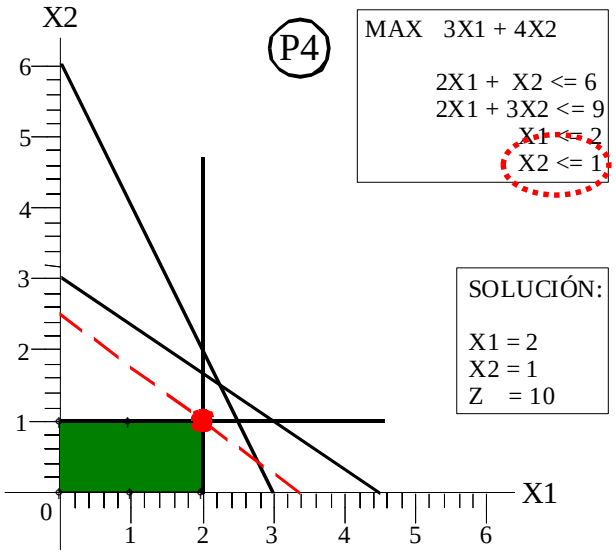
5.2 Técnicas de programación entera: algoritmo de bifurcación y acotación



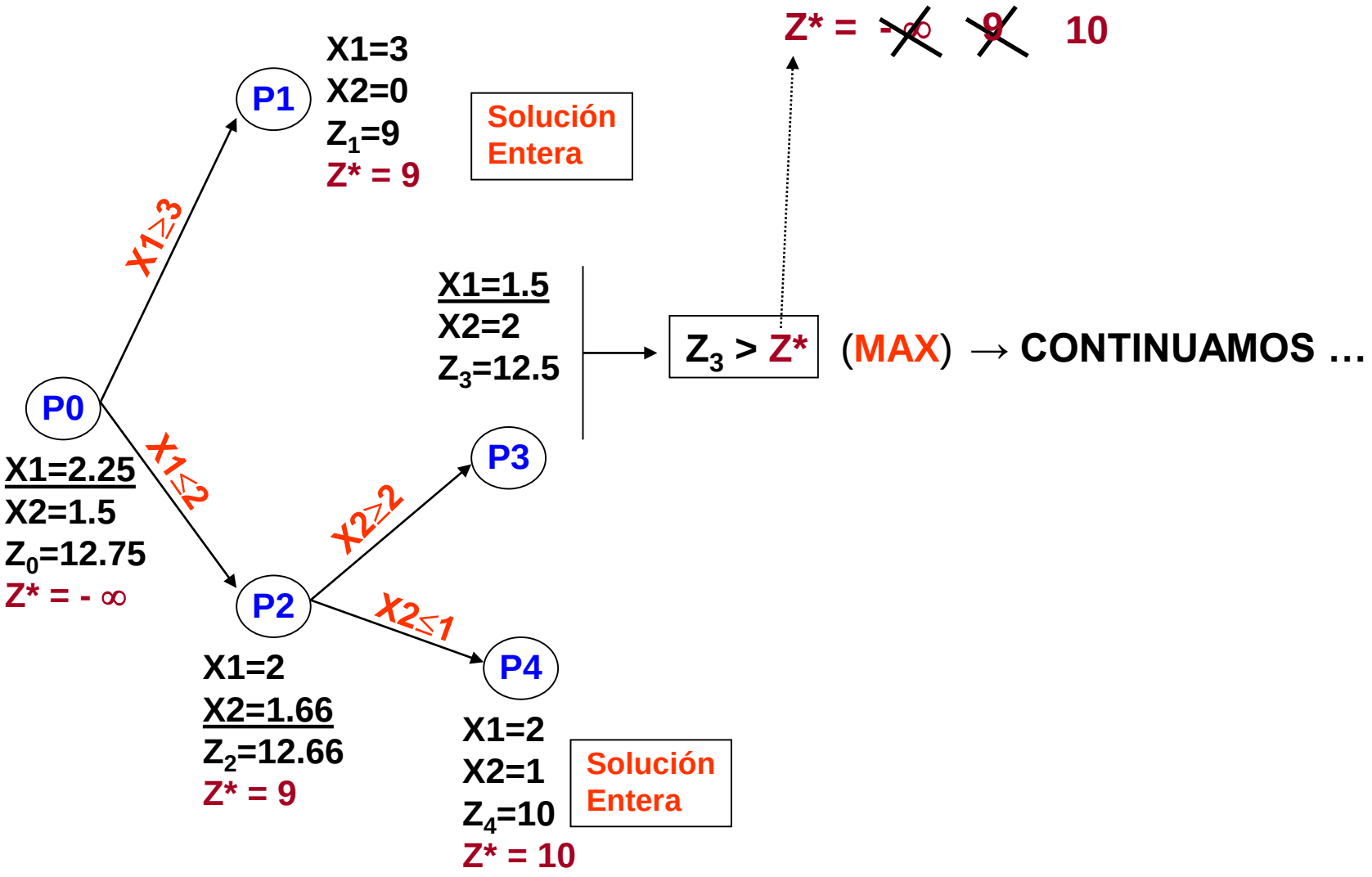
5.2 Técnicas de programación entera: algoritmo de bifurcación y acotación



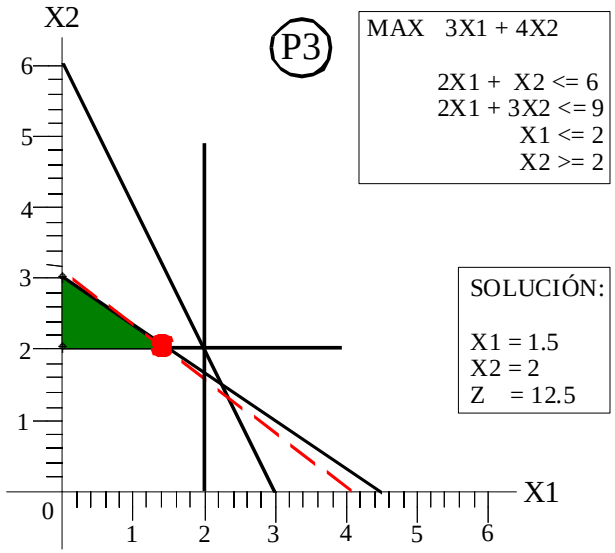


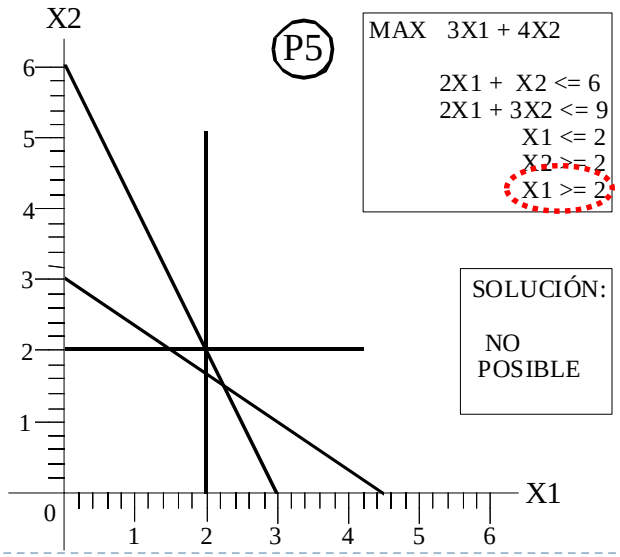


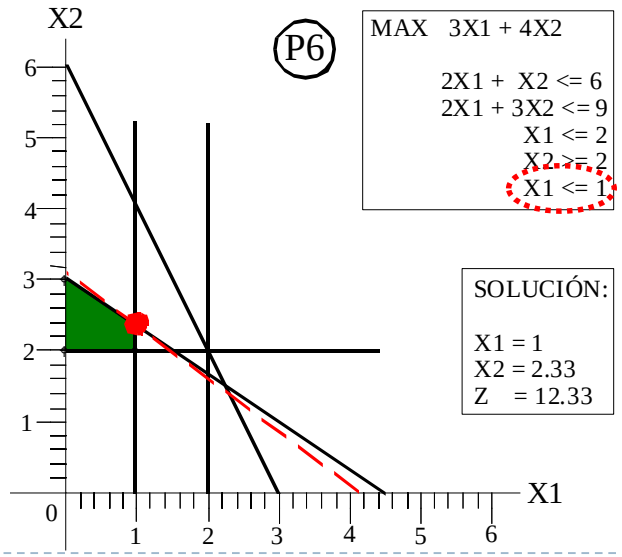
5.2 Técnicas de programación entera: algoritmo de bifurcación y acotación



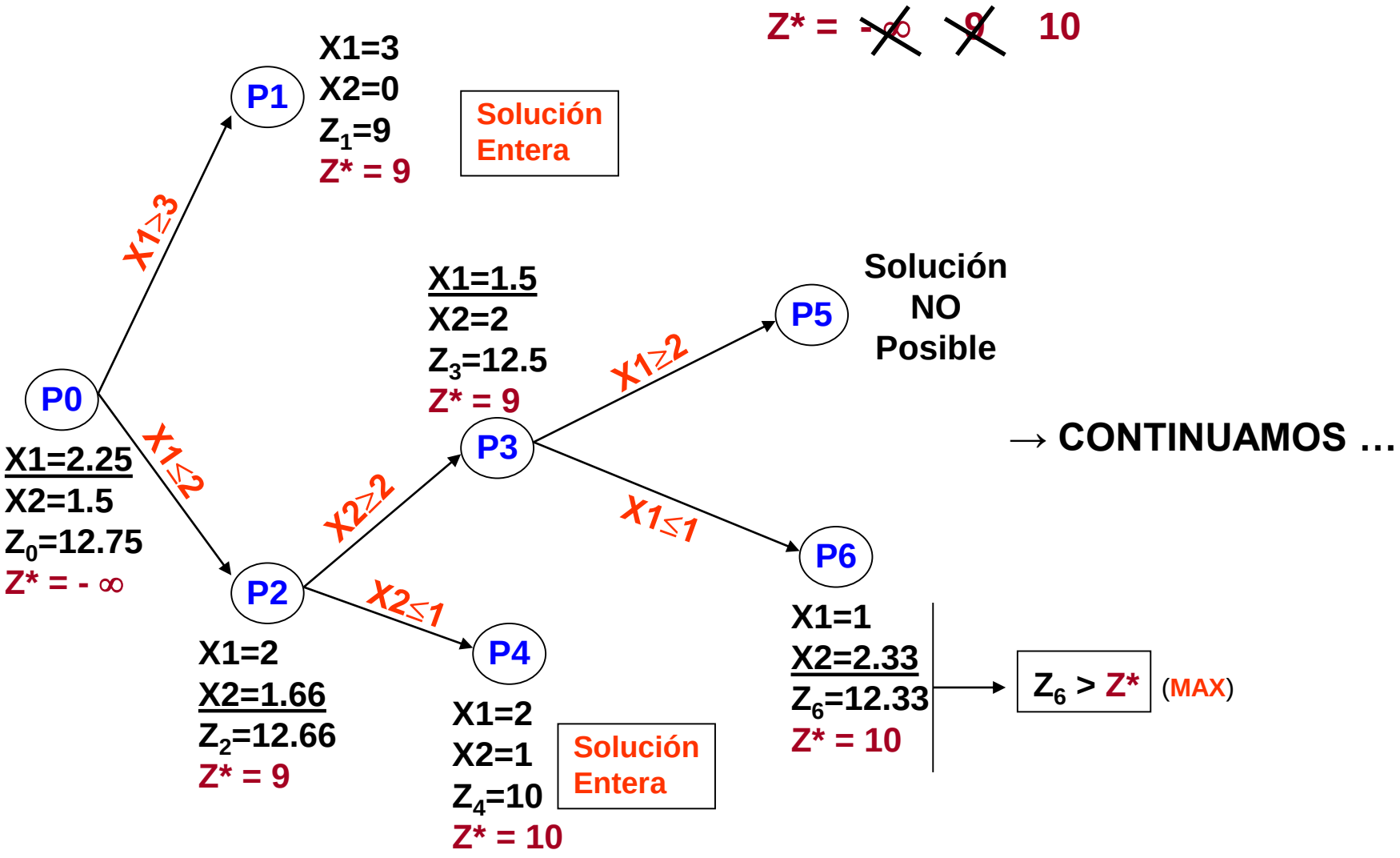
5.2 Técnicas de programación entera: algoritmo de bifurcación y acotación



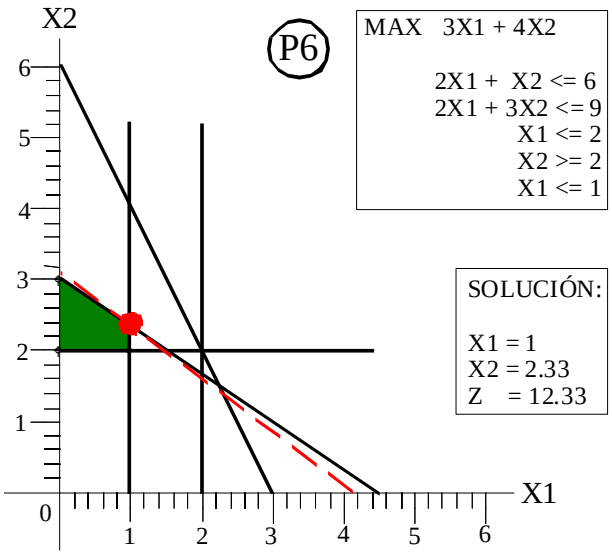




5.2 Técnicas de programación entera: algoritmo de bifurcación y acotación



5.2 Técnicas de programación entera: algoritmo de bifurcación y acotación



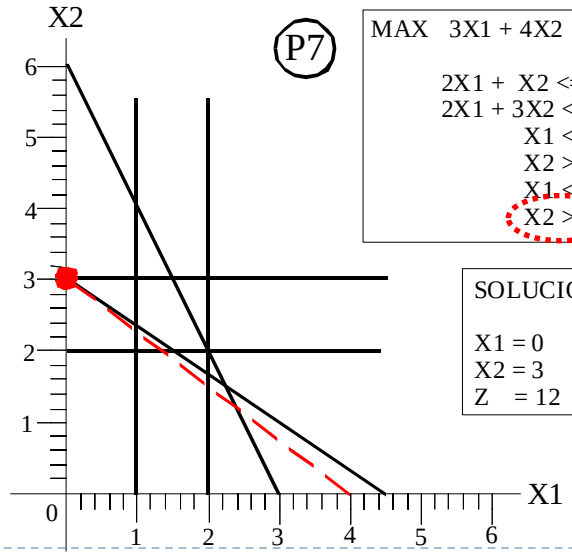
P6

MAX 3X1 + 4X2

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

SOLUCIÓN:

X1 = 1
X2 = 2.33
Z = 12.33



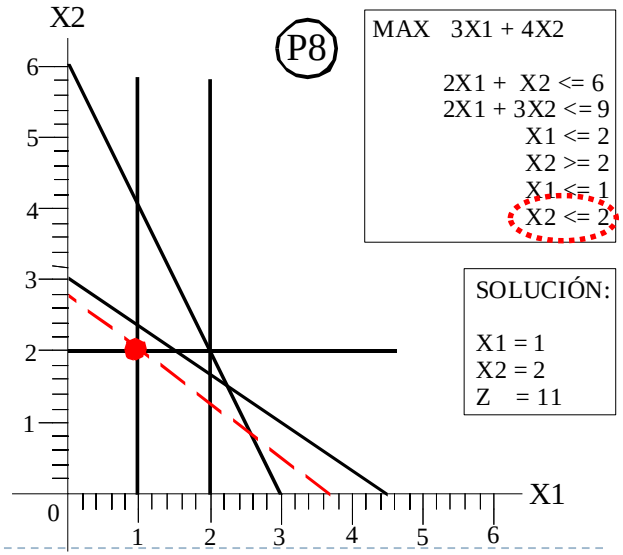
P7

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

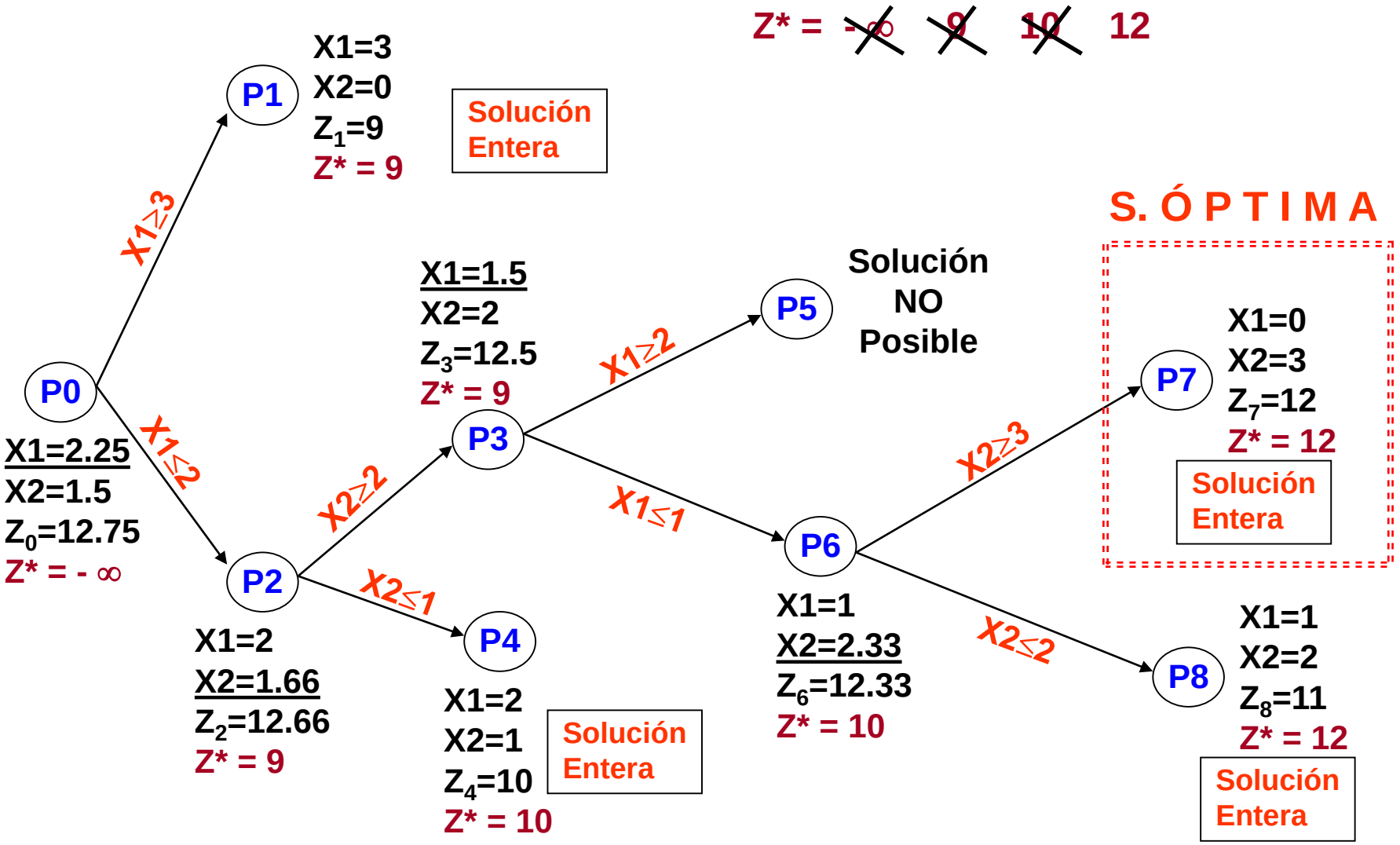
MAX 3X1 + 4X2

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

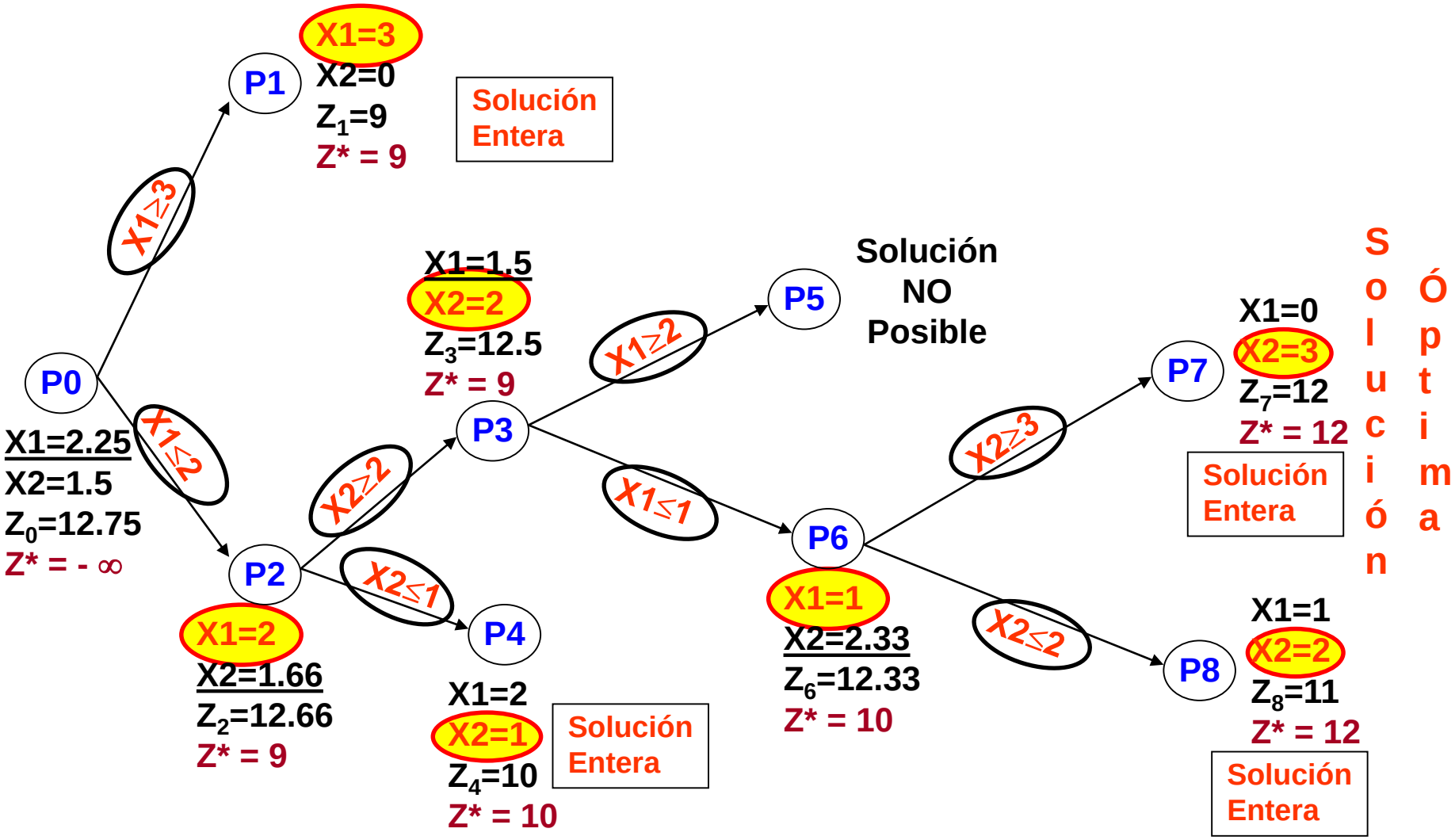
SOLUCIÓN:

X1 = 1
X2 = 2
Z = 11

5.2 Técnicas de programación entera: algoritmo de bifurcación y acotación



5.2 Técnicas de programación entera: algoritmo de bifurcación y acotación



5.2 Técnicas de programación entera: algoritmo de bifurcación y acotación

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)

5.2 Técnicas de programación entera: algoritmo de bifurcación y acotación

¿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.

¿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

5.2 Técnicas de programación entera: algoritmo de bifurcación y acotación

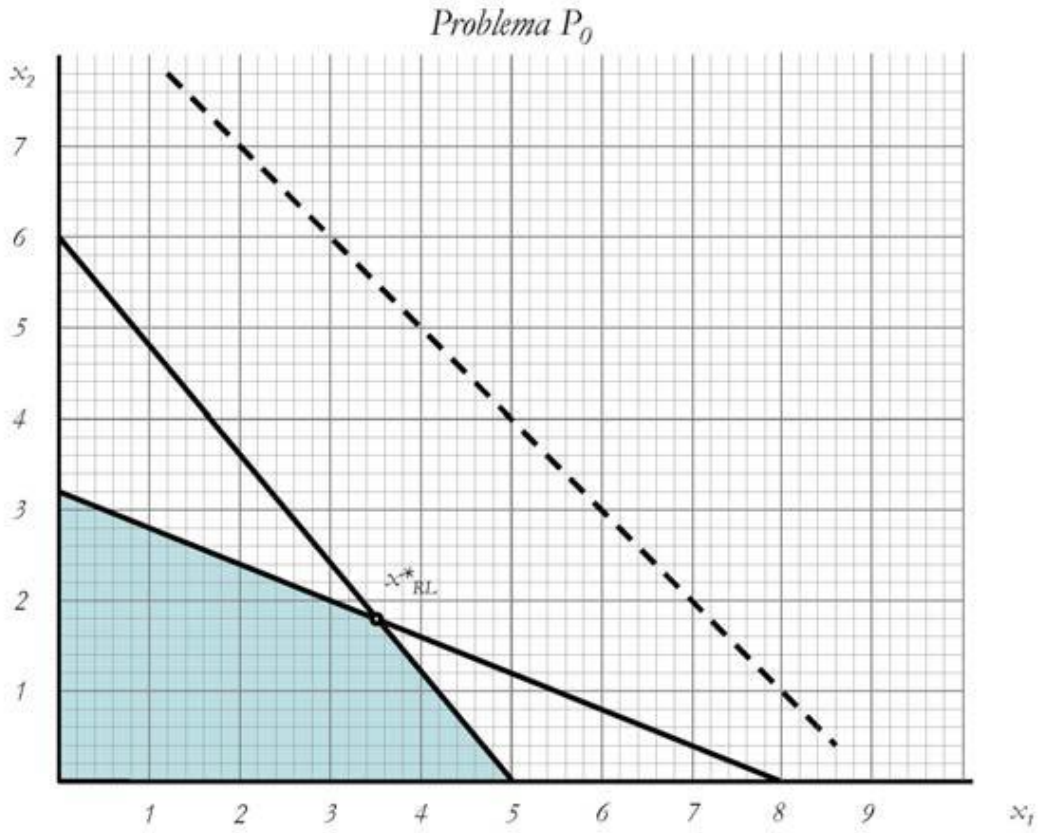
- EJERCICIO:

Dado el problema:

$$\text{Max } z = x1 + x2$$
$$\text{sujeto a:}$$
$$2x1 + 5x2 \leq 16$$
$$6x1 + 5x2 \leq 30$$
$$x1, x2 \geq 0 \text{ y enteros}$$

Se conoce la solución óptima de la relajación lineal correspondiente, $x*$: x1= 7/2 y x2= 9/5 que se muestra en la siguiente imagen:

5.2 Técnicas de programación entera: algoritmo de bifurcación y acotación



- Resolver de forma gráfica el problema entero utilizando *Branch&Bound*, dibujando el árbol correspondiente.

Considera el siguiente problema lineal entero:

$$\left.\begin{array}{rr} \text{Max} & z = 3x_1 + 4x_2 + 5x_3 \\ \text{s.a:} & 2x_1 + x_2 + x_3 \leq 6 \\ & 2x_1 + 3x_2 + 4x_3 \leq 9 \\ & x_1 + 3x_2 + 2x_3 \leq 7 \\ & x_1\,, x_2\,, x_3 \geq 0 \\ & x_1\,, x_2\,, x_3 \text{ enteras} \end{array}\right\}.$$

Se ha aplicado el algoritmo de bifurcación y acotación (branch and bound) para encontrar la solución óptima de este problema. Las iteraciones resultantes se muestran en la página siguiente.

A partir de la información proporcionada por el algoritmo, responde a las siguientes cuestiones de forma razonada.

a) Representa en forma de árbol los pasos seguidos por el algoritmo. Indica claramente qué solución representa cada nodo del árbol y a qué iteración corresponde, y qué cota es la que se aplica en cada arista o rama. [1 punto]

b) De acuerdo con las normas que rigen el algoritmo, ¿debería haberse bifurcado el nodo (P_9)? Justifica tu respuesta. [0,5 puntos]

c) Explica qué problema lineal se ha resuelto en la iteración (P_6). [0,5 puntos]

d) Si se hubiera seguido el criterio de la «mejor cota», ¿cuál habría sido la secuencia de visita de los diferentes problemas? [0,5 puntos]

(P_1) $z = 12{,}75$			
Variable	Valor	Cota inf.	Cota sup.
x_1	2,25	0	M
x_2	1,5	0	M
x_3	0	0	M

(P_2) $z = 9$			
Variable	Valor	Cota inf.	Cota sup.
x_1	3	3	M
x_2	0	0	M
x_3	0	0	M

(P_3) $z = 12{,}667$			
Variable	Valor	Cota inf.	Cota sup.
x_1	2	0	2
x_2	1,667	0	M
x_3	0	0	M

(P_4) $z = 11$			
Variable	Valor	Cota inf.	Cota sup.
x_1	1	0	2
x_2	2	2	M
x_3	0	0	M

(P_5) $z = 12{,}5$			
Variable	Valor	Cota inf.	Cota sup.
x_1	2	0	2
x_2	1	0	1
x_3	0,5	0	M

(P_6) $z = 12{,}33$			
Variable	Valor	Cota inf.	Cota sup.
x_1	2	0	2
x_2	0,333	0	1
x_3	1	1	M

(P_7) $z = 12$			
Variable	Valor	Cota inf.	Cota sup.
x_1	1	0	2
x_2	1	1	1
x_3	1	1	M

(P_8) $z = 12{,}25$			
Variable	Valor	Cota inf.	Cota sup.
x_1	2	0	2
x_2	0	0	0
x_3	1,25	1	M

(P_9) $z = 11{,}5$			
Variable	Valor	Cota inf.	Cota sup.
x_1	0,5	0	2
x_2	0	0	0
x_3	2	2	M

(P_{10}) $z = 11$			
Variable	Valor	Cota inf.	Cota sup.
x_1	2	0	2
x_2	0	0	0
x_3	1	1	1

(P_{11}) $z = 10$			
Variable	Valor	Cota inf.	Cota sup.
x_1	2	0	2
x_2	1	0	1
x_3	0	0	0

5.3 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

5.3 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

5.3 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.

5.3 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**)

5.3 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

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**

5.3 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.