TEMA 5. EJERCICIOS PROPUESTOS Y RESUELTOS

Branch and Bound

5.1 Branch&Bound. Se ha resuelto mediante el algoritmo Branch-and-Bound el siguiente problema lineal entero:

$$\left.\begin{array}{rr} \text{Max} & z = 20x_1 + 18x_2 + 5x_3 \\ \text{s.a:} & x_1 + x_2 + x_3 = 5 \\ & 7x_1 + 3{,}5x_2 + x_3 \leq 24{,}5 \\ & 3x_1 + 9x_2 + x_3 \leq 24{,}5 \\ & x_1\,, x_2\,, x_3 \geq 0 \text{ y enteras} \end{array}\right\}$$

A partir de las iteraciones resultantes (ver página siguiente), responde de manera razonada a las siguientes cuestiones:

a) Reproduce el árbol que genera el algoritmo al ser aplicado a este problema. En cada nodo indica la solución obtenida y si se produce alguna mejora en la cota de la solución óptima entera. Indica qué cota o restricción representa cada rama.

b) Segúnel algoritmo Branch-and-Bound, explica si hay en este árbol ramas que no deberían haberse generado y si, por otro lado, falta bifurcar algún nodo. JUSTIFICA tu respuesta en cualquier caso.

c) Si la información proporcionada por la ejecución del algoritmo lo permite, explica cuál es la solución óptima entera del problema; en caso contrario, indica cuál es la mejor solución entera encontrada.

d) Explica qué árbol se obtendría y cuál sería la solución óptimadel problema en el caso de que las variables x_2 y $x_3$tuviesen naturaleza continua (es decir, si sólo x_1 estuviese obligada a tomar valores enteros).

(P_1)	z = 85,9942		
Variable	Valor	Cota inf.	Cota sup.
x_1	2,4942	0	M
x_2	1,8140	0	M
x_3	0,6919	0	M

(P_2)	z= 77,8		
Variable	Valor	Cota inf.	Cota sup.
x_1	3	3	M
x_2	0,6	0	M
x_3	1,4	0	M

(P_3)	Problema imposible		
Variable	Valor	Cota inf.	Cota sup.
x_1		3	M
x_2		1	M
x_3		0	M

(P_4)	z = 73,75		
Variable	Valor	Cota inf.	Cota sup.
x_1	3,25	3	M
x_2	0	0	0
x_3	1,75	0	M

(P_5)	Problema imposible		
Variable	Valor	Cota inf.	Cota sup.
x_1		4	M
x_2		0	0
x_3		0	M

(P_6)	z = 70		
Variable	Valor	Cota inf.	Cota sup.
x_1	3	3	3
x_2	0	0	0
x_3	2	0	M

(P_7)	z = 80,1875		
Variable	Valor	Cota inf.	Cota sup.
x_1	2	0	2
x_2	1,9375	0	M
x_3	1,0625	0	M

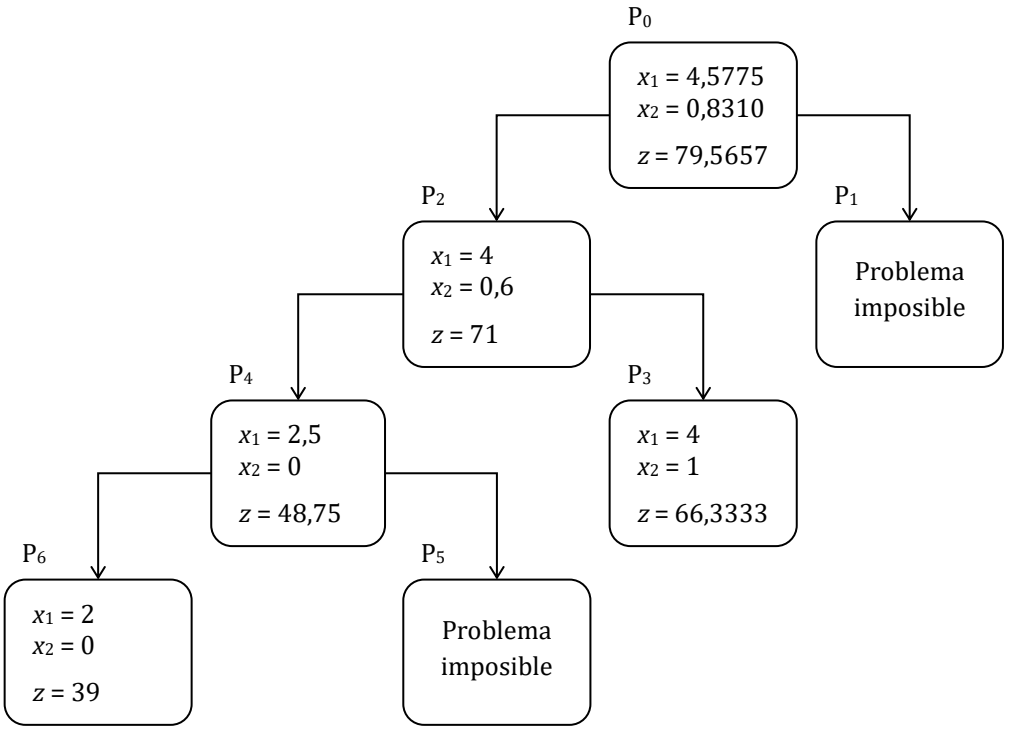
(P_8)	z = 77,25		
Variable	Valor	Cota inf.	Cota sup.
x_1	1,75	0	2
x_2	2	2	M
x_3	1,25	0	M

(P_9)	Problema imposible		
Variable	Valor	Cota inf.	Cota sup.
x_1		2	2
x_2		2	M
x_3		0	M

(P_{10})	z = 68,4375		
Variable	Valor	Cota inf.	Cota sup.
x_1	1	0	1
x_2	2,1875	2	M
x_3	1,8125	0	M

(P_{11})	z = 68		
Variable	Valor	Cota inf.	Cota sup.
x_1	2	0	2
x_2	1	0	1
x_3	2	0	M

5.2 Branch&Bound. Se ha aplicado el algoritmo Branch-and-Bound para resolver un problema lineal de maximización con dos variables enteras. El árbol resultante es el siguiente (los nodos están numerados según el orden en que han sido resueltos):



a) Completa la información del árbol: indica qué cota o restricción representa cada rama, así como el valor que va tomando en cada nodo la cota inferior de la solución óptima entera.

b) De acuerdo con el algoritmo Branch-and-Bound, explica si hay en este árbol ramas que no deberían haberse generado y si, por otro lado, falta bifurcar algún nodo. Justifica adecuadamente tu respuesta.

c) Si el árbol está completo, explica cuál es la solución o soluciones óptimas del problema. Si no lo está, indica cuál es la mejor solución entera encontrada.

d) ¿Se corresponde el orden de generación de los nodos con alguna de las técnicas expuestas en la asignatura? Justifica tu respuesta.

e) Dibuja o explica cómo sería el árbol resultante de aplicar el algoritmo Branch-and-Bound en el caso de que al problema se le añadieran las restricciones $2 \le x_1 \le 4$ y $2x_2 \le x_1$. Supón que se mantiene el criterio para recorrer el árbol visto en el apartado (d).

5.3 Branch&Bound. Se ha introducido en un software de optimización y resuelto mediante el algoritmo Branch-and-Bound el siguiente problema lineal entero de maximización:

Variable -->	X1	X2	Direction	R. H. S.
Maximize	12.5	17		
C1	1	2	<=	8
C2	1	-1	<=	4
LowerBound	0	0		
UpperBound	M	M		
VariableType	Integer	Integer		

Los resultados que se han logrado recoger (podrían estar incompletos) son los siguientes:

P_0:

	Decision Variable	Lower Bound	Upper Bound	Solution Value
1	X1	0	M	5,3333
2	X2	0	M	1,3333
	Current	OBJ(Maximize)	= 89,3333	

P_3:

	Decision Variable	Lower Bound	Upper Bound	Solution Value
1	X1	0	5,0000	4,0000
2	X2	2,0000	M	2,0000
	Current	OBJ(Maximize)	= 84,0000	

P_1:

	Decision Variable	Lower Bound	Upper Bound	Solution Value
1	X1	6,0000	M	
2	X2	0	M	
	This	node	is	infeasible

P_4:

	Decision Variable	Lower Bound	Upper Bound	Solution Value
1	X1	0	5,0000	5,0000
2	X2	0	1,0000	1,0000
	Current	OBJ(Maximize)	= 79,5000	

P_2:

	Decision Variable	Lower Bound	Upper Bound	Solution Value
1	X1	0	5,0000	5,0000
2	X2	0	M	1,5000
	Current	OBJ(Maximize)	= 88,0000	

a) A partir de la salida de Un software de optimización, reproduce el árbol generado por el algoritmo Branch-and-Bound. Indica qué cota o restricción representa cada rama, y qué solución se obtiene en cada nodo.

b) De acuerdo con el algoritmo Branch-and-Bound, explica si hay en este árbol ramas que no deberían haberse generado y si, por otro lado, falta bifurcar algún nodo. Justifica adecuadamente tu respuesta.

c) Si el árbol está completo, explica cuál es la solución o soluciones óptimas del problema. Si no lo está, indica cuál es la mejor solución entera encontrada.

d) Explica qué problema se ha resuelto en el nodo P_2.

5.4 Branch&Bound. Consideremos el siguiente problema de Programación Lineal Entera:

Variable -->	X1	X2	X3	Direction	R. H. S.
Minimize	1	-2	2		
C1	1	1	-2	<=	4
C2	2	-1	-1	>=	3
LowerBound	0	0	0		
UpperBound	M	4	M		
VariableType	Integer	Integer	Continuous		

A continuación se presentan los resultados obtenidos al resolver el problema anterior mediante un software de optimización:

1

Decision Variable	Lower Bound	Upper Bound	Solution Value
X1	0	M	2,3333
X2	0	4,0000	1,6667
X3	0	M	0
Current	OBJ(Minimize)	= -1,0000	

2

Decision Variable	Lower Bound	Upper Bound	Solution Value
X1	3,0000	M	3,0000
X2	0	4,0000	2,3333
X3	0	M	0,6667
Current	OBJ(Minimize)	= -0,3333	

3

Decision Variable	Lower Bound	Upper Bound	Solution Value
X1	3,0000	M	3,6667
X2	3,0000	4,0000	3,0000
X3	0	M	1,3333
Current	OBJ(Minimize)	= 0,3333	

4

Decision Variable	Lower Bound	Upper Bound	Solution Value
X1	4,0000	M	4,0000
X2	3,0000	4,0000	3,3333
X3	0	M	1,6667
Current	OBJ(Minimize)	= 0,6667	

5

Decision Variable	Lower Bound	Upper Bound	Solution Value
X1	4,0000	M	4,6667
X2	4,0000	4,0000	4,0000
X3	0	M	2,3333
Current	OBJ(Minimize)	= 1,3333	

6

Decision Variable	Lower Bound	Upper Bound	Solution Value
X1	5,0000	M	5,0000
X2	4,0000	4,0000	4,0000
X3	0	M	2,5000
Current	OBJ(Minimize)	= 2,0000	

7

Decision Variable	Lower Bound	Upper Bound	Solution Value
X1	4,0000	4,0000	
X2	4,0000	4,0000	
X3	0	M	
This	node	is	infeasible

8

Decision Variable	Lower Bound	Upper Bound	Solution Value
X1	4,0000	M	4,0000
X2	3,0000	3,0000	3,0000
X3	0	M	1,5000
Current	OBJ(Minimize)	= 1,0000	

9

Decision Variable	Lower Bound	Upper Bound	Solution Value
X1	3,0000	3,0000	
X2	3,0000	4,0000	
X3	0	M	
This	node	is	infeasible

10

Decision Variable	Lower Bound	Upper Bound	Solution Value
X1	3,0000	M	3,0000
X2	0	2,0000	2,0000
X3	0	M	0,5000
Current	OBJ(Minimize)	= 0	

11

Decision Variable	Lower Bound	Upper Bound	Solution Value
X1	0	2,0000	2,0000
X2	0	4,0000	1,0000
X3	0	M	0
Current	OBJ(Minimize)	= 0	

a) Construye el árbol obtenido tras aplicar el algoritmo Branch-and-Bound, numerando los nodos según el orden en el que se han ido generando. Indica en cada nodo: la solución obtenida, el valor de la función objetivo para dicha solución, y el valor de la cota de la solución entera del problema tras esa iteración. Indica en cada rama la restricción añadida al problema. ¿Cuál es la solución óptima entera del problema?

b) ¿Qué criterio se ha seguido para desarrollar el árbol en el apartado anterior? Construye el árbol de Branch-and-Bound que se obtendría al aplicar el otro criterio explicado en la asignatura.

Soluciones

5.1

a)

Véase el árbol resultante en la página siguiente.

b)

NO falta bifurcar ningún nodo, o lo que es lo mismo, todos los nodos terminales están correctamente saturados, de acuerdo con el procedimiento del algoritmo Branch-and-Bound; en concreto:

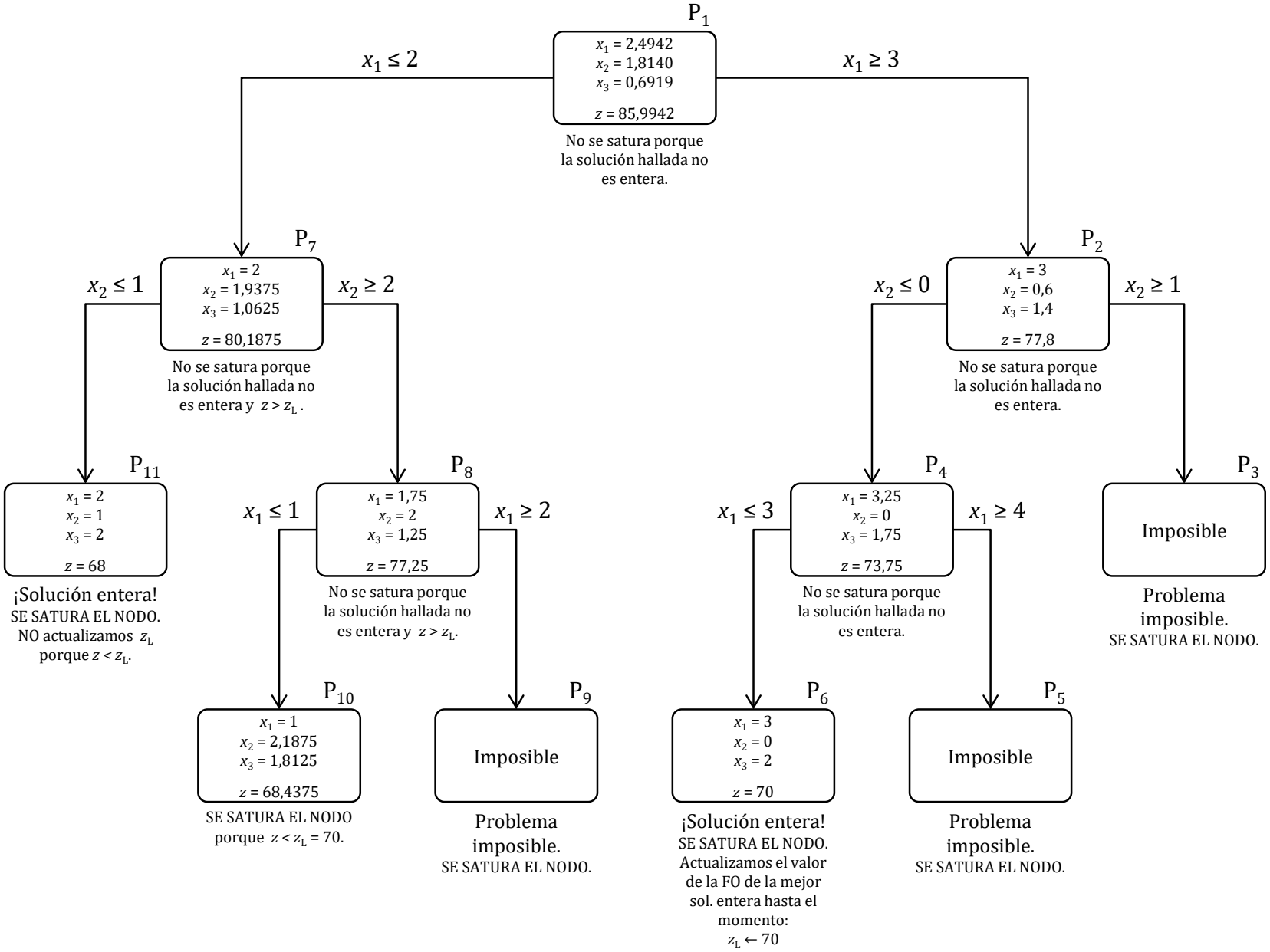
- Los nodos P_3, P_5 y P_9 se saturan porque en ellos se llega a un problema imposible.
- Los nodos P_6 y P_{11} se saturan porque en ellos se alcanzan sendas soluciones enteras del problema.
- El nodo P_{10} está correctamente saturado porque, aunque la solución en él obtenida NO es entera, ésta es peor que la mejor solución entera encontrada hasta ese momento (el nodo P_{10} se visita después del nodo P_6, en el que ya se ha obtenido una solución entera, que proporciona un valor dezmayor que el que ofrece P_{10} —70 vs 68,4375—).

Por otro lado, no "sobra" ninguna rama: todas los nodos no terminales han sido bifurcados de manera correcta: en todos ellos (P_1, P_2, P_4, P_7 y P_8) el subproblema lineal resultante no es imposible, ni presenta solución óptima entera ni presenta una solución óptima peor que la mejor solución entera encontrada hasta ese momento.

c)

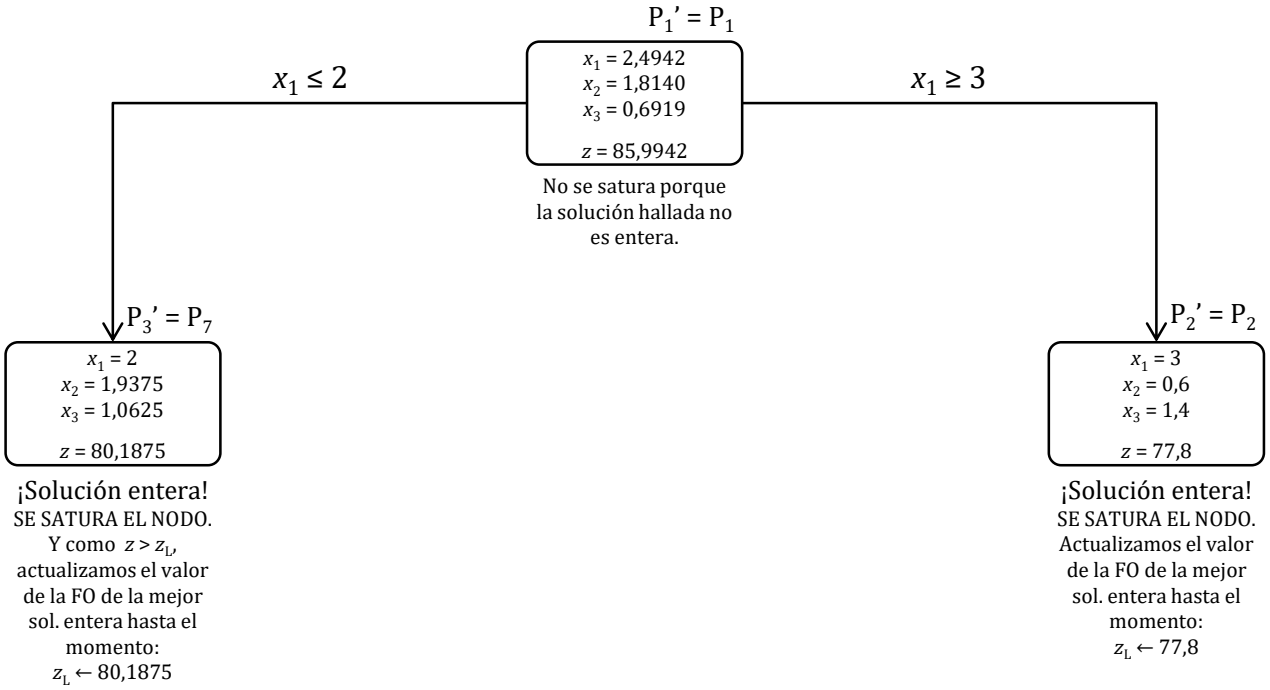
Según el apartado anterior, el árbol está completo, y por tanto es posible conocer cuál es la solución óptima entera del problema (la mejor de todas las soluciones enteras encontradas); en este caso, se alcanza en el nodo P_6:

$$\text{Sol. óptima entera: } (x_1^* = 3\ ,\ x_2^* = 0\ ,\ x_3^* = 2)\ ;\ z^* = 70$$


P1
x1 = 2,4942
x2 = 1,8140
x3 = 0,6919
z = 85,9942
No se satura porque la solución hallada no es entera.
x1 ≤ 2
x1 ≥ 3
P7
x1 = 2
x2 = 1,9375
x3 = 1,0625
z = 80,1875
No se satura porque la solución hallada no es entera y z > zL .
x2 ≤ 1
x2 ≥ 2
P2
x1 = 3
x2 = 0,6
x3 = 1,4
z = 77,8
No se satura porque la solución hallada no es entera.
x2 ≤ 0
x2 ≥ 1
P11
x1 = 2
x2 = 1
x3 = 2
z = 68
¡Solución entera!
SE SATURA EL NODO.
NO actualizamos zL porque z < zL.
P8
x1 = 1,75
x2 = 2
x3 = 1,25
z = 77,25
No se satura porque la solución hallada no es entera y z > zL.
x1 ≤ 1
x1 ≥ 2
P4
x1 = 3,25
x2 = 0
x3 = 1,75
z = 73,75
No se satura porque la solución hallada no es entera.
x1 ≤ 3
x1 ≥ 4
P3
Imposible
Problema imposible.
SE SATURA EL NODO.
P10
x1 = 1
x2 = 2,1875
x3 = 1,8125
z = 68,4375
SE SATURA EL NODO porque z < zL = 70.
P9
Imposible
Problema imposible.
SE SATURA EL NODO.
P6
x1 = 3
x2 = 0
x3 = 2
z = 70
¡Solución entera!
SE SATURA EL NODO.
Actualizamos el valor de la FO de la mejor sol. entera hasta el momento:
zL ← 70
P5
Imposible
Problema imposible.
SE SATURA EL NODO.

d)

Si el problema tuviera x_1 como única variable de naturaleza entera, entonces el árbol resultante habría sido el siguiente:



Las respectivas soluciones proporcionadas por los nodos P_2 y P_7 (ahora P_3') ya serían enteras (puesto que en ambas la variable que debe tomar valor entero lo toma), motivo por el cual se saturarían.

La mejor solución entera se alcanza en el nodo P_3'; por tanto, la solución óptima del problema en este caso sería:

Sol. óptima entera: $(x_1^* = 2\ ,\ x_2^* = 1{,}9375\ ,\ x_3^* = 1{,}0625)\ ;\ z^* = 80{,}1875$

□

5.2

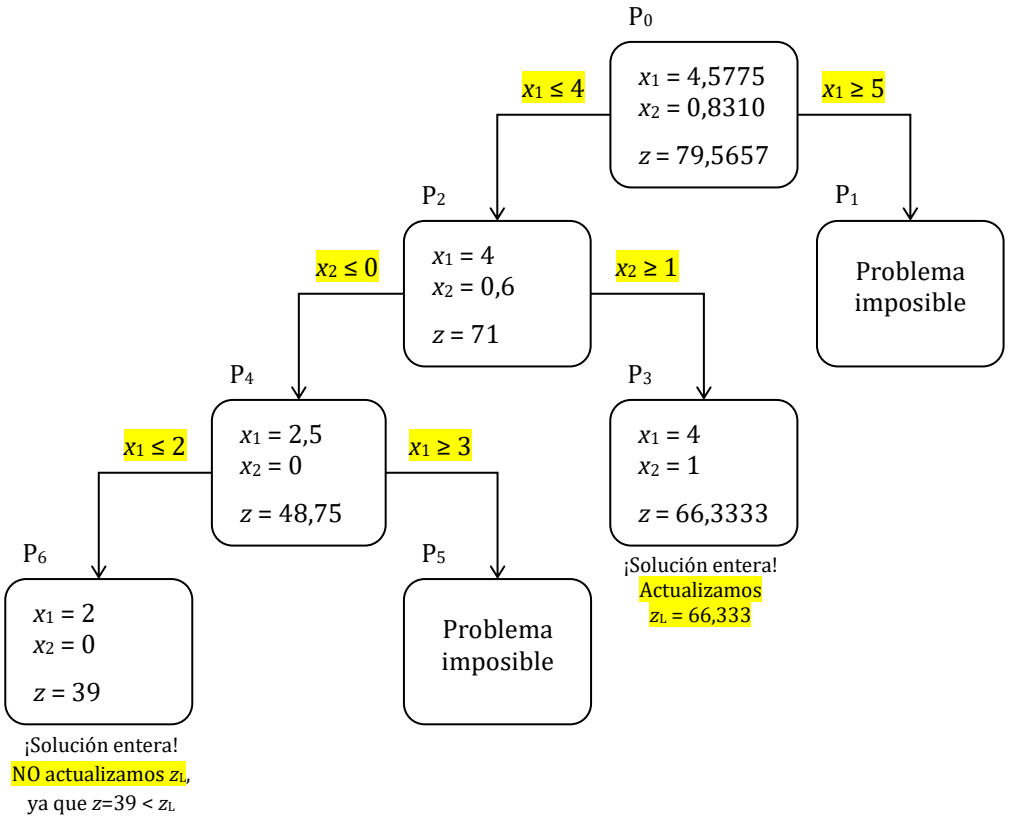
a)

Se pide completar la información del árbol, indicando las cotas en cada rama y las sucesivas actualizaciones de la cota inferior para la solución óptima entera.

En la primera bifurcación del árbol, tras la resolución del nodo raíz P_0, la variable elegida para bifurcar ha sido x_1, y no x_2 (que también era "elegible"). Esto se deduce del hecho de que, en la resolución del nodo P_2, es x_1 la que toma un valor entero (al bifurcar por una variable, dicha variable tomará en la solución de cada nodo "hijo" el valor de la cota añadida sobre ella).

La única actualización de la cota z_L para la solución óptima entera se produce en el nodo P_3. La resolución de ese subproblema genera una solución entera al problema, con $z = 66{,}3333$. Por eso, a partir de ese momento se rechazará cualquier nodo que ofrezca una solución (entera o no) con z peor (inferior, en este caso) a $z_L = 66{,}3333$.

En la resolución del subproblema P_6 se encuentra una nueva solución entera, que NO mejora la mejor (y única) solución entera encontrada hasta ese momento; por ello, no se actualiza el valor de z_L.



[×]

b)

Nos piden que validemos el árbol construido. Concretamente, se desea saber si falta bifurcar algún nodo y si, por otro lado, existen nodos que no deberían haber llegado a generarse.

El algoritmo Branch-and-Bound indica que debe saturarse un nodo cuando en él sucede una de las tres siguientes situaciones:

1. Se obtiene una solución entera.
2. El problema es imposible.

3. La solución obtenida NO es entera pero es PEOR (o no es mejor) que la mejor solución entera encontrada hasta el momento.

Según los criterios 1 y 2 enunciados, todos los nodos terminales del árbol están saturados de manera justificada: los nodos P_3 y P_6 presentan sendas soluciones enteras, mientras que los subproblemas P_1 y P_5 son imposibles. Esto quiere decir que NO hay ninguna rama pendiente de bifurcar; no falta bifurcar ningún nodo.

Sin embargo, según el criterio 3, el nodo P_4 también debería haber sido saturado, ya que la solución obtenida en dicho nodo es *peor* que la mejor solución entera que se ha encontrado hasta ese momento (z = 48,75 < z_L = 66,3333, y estamos maximizando). Por tanto, dicho nodo NO debería haberse bifurcado. Es decir, "sobran" los nodos P_5 y P_6.

ATENCIÓN: Incluso si el árbol se ha generado siguiendo la estrategia de la "cota más reciente" (o "recorrer el árbol en profundidad"), el tercer criterio para cerrar un nodo ("la solución obtenida NO es entera pero es PEOR que la mejor solución entera encontrada hasta el momento") debe aplicarse cuando corresponda. Los tres motivos para cerrar un nodo expuestos se aplican siempre, independientemente de qué estrategia concreta se siga para recorrer el árbol ("cota más reciente", "mejor cota", etc.); son propios del algoritmo Branch-and-Bound en sí.

[×]

c)

El árbol está completo (incluso con nodos de más, según se ha visto en el apartado anterior), y la mejor solución entera encontrada tras explorar toda la región factible es la producida por el nodo P_3, es decir:

Solución óptima entera: $(x_1^* = 4\ ,\ x_2^* = 1)\ ,\ z^* = 66{,}3333$.

Es la única solución óptima entera, ya que no hemos encontrado en la exploración ningún otro punto que proporcione el mismo valor para la función objetivo.

[×]

d)

Se pide deducir cuál de las dos estrategias estudiadas en la asignatura para recorrer el árbol es la que se ha utilizado en este ejercicio.

En este caso, el orden en que se han ido generando los nodos coincide tanto con el criterio de la "mejor cota" como con el de la "cota más reciente".

Coincide con el criterio de la "mejor corta" porque en cada iteración se ha tomado para ser bifurcado el nodo "abierto" con el mejor valor de z (en realidad, sólo hay un nodo "abierto" en cada paso) y se han generado y resuelto sus dos nodos "hijo".

Coincide también con el criterio de la "cota más reciente" porque se ha recorrido el árbol de manera "mecánica", profundizando siempre primero en la "rama" de la derecha o de "≥" hasta cerrarla, y siguiendo después por la última "rama" abierta pendiente de generar.

Es decir, en este caso ambas estrategias producen el mismo árbol.

En realidad, como en cada bifurcación uno de los dos nodos "hijo" acaba siendo "saturado", no hay ocasión de descartar ninguno de los dos criterios: el árbol construido es "compatible" con cualquiera de los dos.

[×]

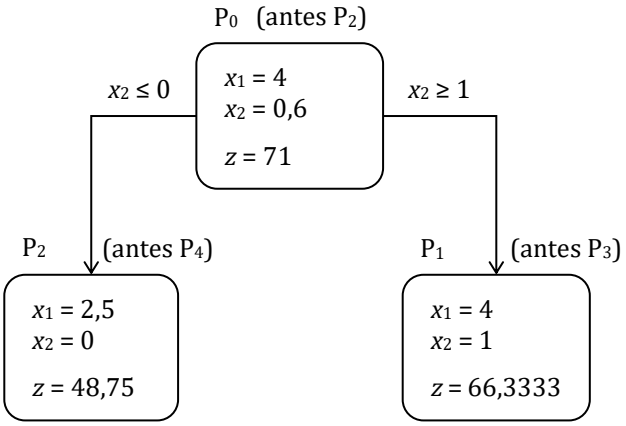
e)

Se añaden al problema las restricciones $2 \le x_1 \le 4$ y $2x_2 \le x_1$.

La restricción $x_1 \le 4$ directamente eliminaría los nodos P_0 y P_1, que son los únicos que incumplen por completo la nueva cota superior impuesta para x_1. Es decir, al resolver el problema lineal asociado al problema entero, directamente se obtendría como nodo "raíz" el nodo P_2 (dicho de otro modo: exigir $x_1 \le 4$ nos sitúa directamente en el nodo P_2).

Las restricciones $x_1 \ge 2$ y $2x_2 \le x_1$ puede que "eliminen" soluciones de la región factible, pero NO influyen en el árbol ni en la solución final, ya que no afectan a las soluciones obtenidas en cada paso del algoritmo (las soluciones que se obtenían en cada nodo, a partir de P_2, ya cumplen esas nuevas restricciones). Por tanto, en cada nodo a partir de P_2 la solución se mantendrá igual, y ningún nodo será eliminado como resultado de añadir estas dos nuevas condiciones.

En resumen, si volvemos a aplicar el algoritmo para el problema con las nuevas restricciones, manteniendo el mismo orden para generar las ramas y eliminando también los nodos que, como resultado del apartado (b), sabemos que no deberían haberse generado, el árbol resultante sería el siguiente:

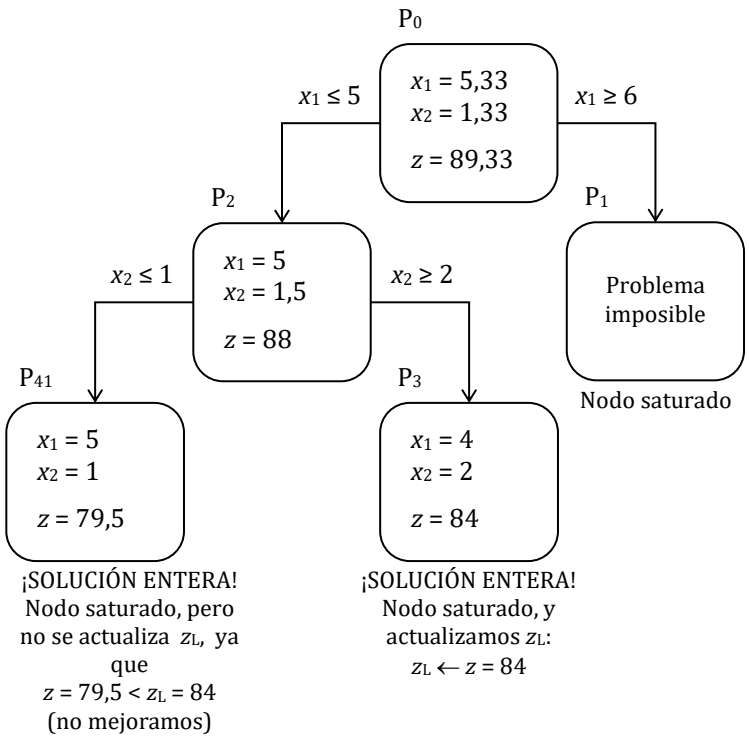

P0 (antes P2)
x1 = 4
x2 = 0,6
z = 71
x2 ≤ 0
x2 ≥ 1
P2 (antes P4)
x1 = 2,5
x2 = 0
z = 48,75
P1 (antes P3)
x1 = 4
x2 = 1
z = 66,3333

[×]

5.3

a)

El árbol que se obtiene a partir de las iteraciones de WinQSB mostradas es el siguiente:



□

b)

El árbol está completo: no le falta bifurcar ninguna rama, ya que todos los nodos terminales están bien saturados: el nodo P_1 se satura porque el problema es imposible, y los nodos P_3 y P_4 se saturan porque en ambos casos se alcanza una solución entera.

Por otro lado, todas las ramas o bifurcaciones están bien realizadas: P_0 y P_2 se bifurcan porque en ambas se obtiene una solución no entera y en el momento de bifurcarlas todavía no se había encontrado ninguna solución entera.

□

c)

El árbol está completo, como se ha visto en el apartado anterior. La solución óptima entera se alcanza en el nodo P_3, ya que es la mejor de las soluciones enteras

encontradas al aplicar el algoritmo (y es única, ya que el resto de soluciones enteras encontradas son peores):

$$\text{Sol. Ópt.: } (\ x_1^* = 4\ ,\ x_2^* = 2\ )\ ;\ \ z^* = 84.$$

□

d)

Según el algoritmo Branch-and-Bound, en cada nodo del árbol se proporciona la solución de un problema LINEAL (no entero); en concreto, la solución hallada en el nodo P_2 es la solución óptima del siguiente problema lineal:

$$\left.\begin{array}{rr} \text{Max} & z = 12{,}5x_1 + 17x_2 \\ \text{s.a:} & x_1 + 2x_2 \le 8 \\ & x_1 - x_2 \le 4 \\ & x_1 \le 5 \\ & x_1\ , x_2 \ge 0 \end{array}\right\},$$

es decir, el resultante de eliminar en el problema principal la condición de integridad de las variables y añadirle las cotas que conducen al nodo P_2.

5.4

a)

Construimos el árbol a partir de la información que proporcionan las pantallas de WinQSB. Hay que tener en cuenta lo siguiente:

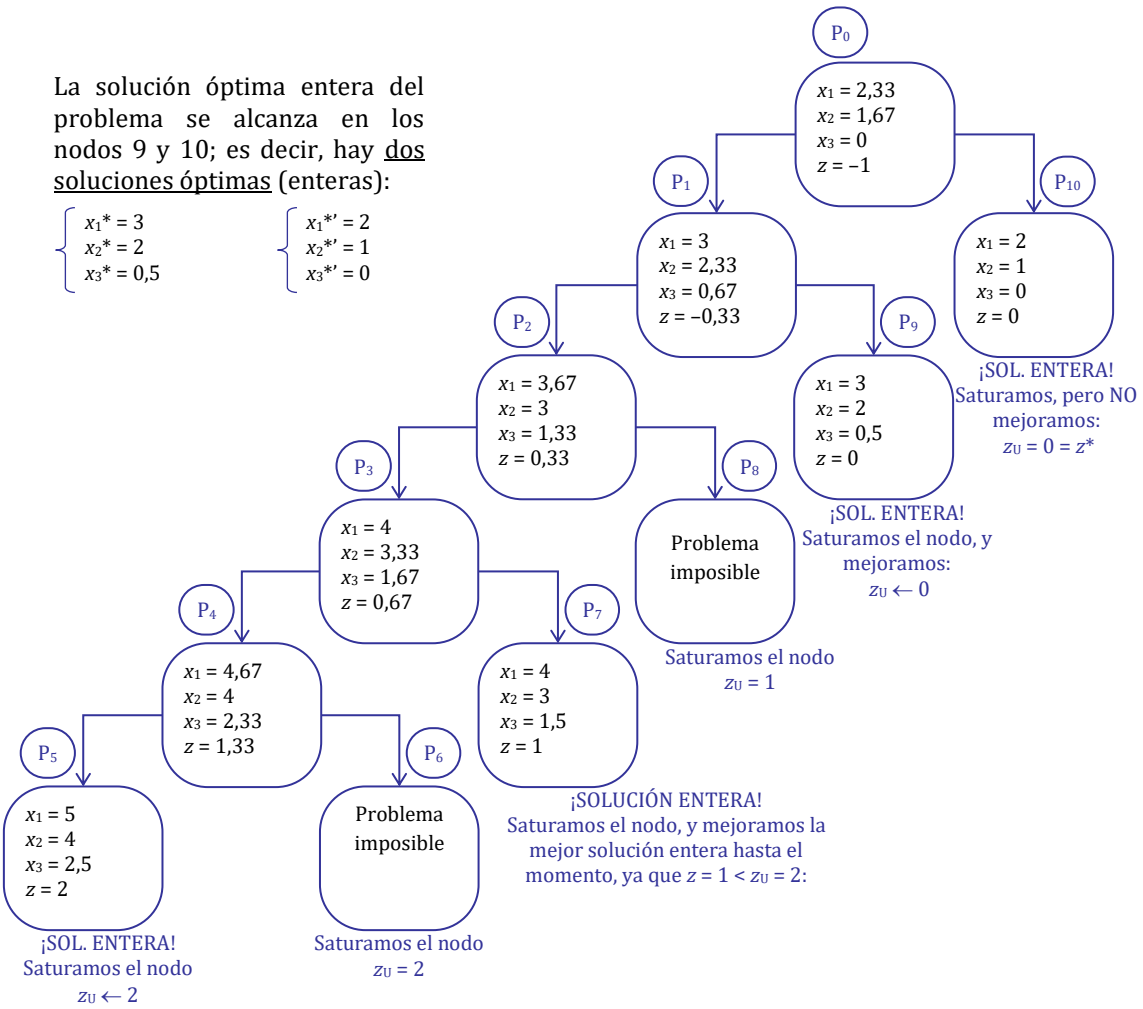
El problema de Programación Lineal Entera Mixta: la variable x_3 es continua, es decir, no se exige que sea entera; por tanto, no se puede bifurcar por ella, etc. Para que una solución del problema se considere "entera" bastará con que x_1 y x_2 tomen valores no decimales.

Hay que saber cuándo un nodo se satura o se "cierra" (las pantallas de WinQSB están cortadas).

Estamos en un problema de minimización; por ello, en cada bifurcación obtendremos soluciones con una z mayor (o igual), y podremos saturar un nodo cuando la solución encontrada en él dé un valor para z mayor que la mejor solución entera encontrada hasta ese momento.

La solución óptima entera del problema se alcanza en los nodos 9 y 10; es decir, hay <u>dos soluciones óptimas</u> (enteras):

$\begin{cases} x_1^* = 3 \\ x_2^* = 2 \\ x_3^* = 0{,}5 \end{cases}$ $\begin{cases} x_1^{*\prime} = 2 \\ x_2^{*\prime} = 1 \\ x_3^{*\prime} = 0 \end{cases}$



b)

El criterio que se ha seguido en el apartado anterior para construir el árbol es el de la <u>"cota más reciente"</u>, que consiste en recorrer el árbol "en profundidad", es decir, explorando una "rama" hasta llegar al final de ésta, y continuando siempre por el último nodo creado y no explorado.

El otro método visto en clase es el de la <u>"mejor cota"</u>, y consiste en continuar bifurcando siempre, de los nodos que todavía estén por explorar, el nodo que mejor valor presente para la función objetivo (menor, en este caso, ya que estamos minimizando), resolviendo sus dos nodos "hijo".

Según esto, el árbol que se habría generado, de haber seguido este otro método, habría sido el siguiente:

(Cambia el orden en que se visitan los nodos, y por ello se saturan nodos que con el método anterior no se saturaban, pero la solución óptima entera NO cambia, por supuesto)