6.3.1 O Algoritmo *Branch-and-Bound* para Solução de Problemas de Programação Inteira

Muitos problemas de programação inteira são resolvidos de forma eficiente pelo método *branch-and-bound*. O algoritmo *branch-and-bound* completo para solução de problemas de programação inteira é descrito a seguir.

Figura 6.6 O algoritmo *branch-and-bound* para solução de problemas de programação inteira.

Início. O problema original de PI relaxado, sem as restrições de integralidade, é representado por (S_0). Crie uma lista contendo S_0, isto é, $L = \{S_0\}$. Se o problema estudado for de maximização, assuma que $z^* = -\infty$ (Z^* representa o melhor valor da função objetivo encontrado até o momento, isto é, o valor de z da solução incumbente). Se for um problema de minimização, assuma que $z^* = +\infty$. Assume-se também que $x^* = \phi$ (x^* representa a solução incumbente).

Passo 1. Se ainda existir um subproblema na lista, vá para o passo 2. Caso contrário, vá para o passo 5.

Passo 2. (Seleção). Remova um subproblema da lista, de acordo com a estratégia de busca adotada, e resolva-o pelo método Simplex ou por computador.

Passo 3. (Eliminação).

Passo 3a. Se a solução obtida apresentar um valor da função objetivo igual ou pior que z^*, ou for infactível, descarte-a e retorne ao passo 1.

Passo 3b. Se a solução não for inteira (e o valor da função objetivo for melhor que z^*), vá para o passo 4.

Passo 3c. Se a solução for inteira (e o valor da função objetivo for melhor do que z^*), atualize z^* e x^* (essa solução passa a ser a solução incumbente e a antiga é descartada). Descarte da árvore cada subproblema que apresentou uma solução igual ou pior à nova solução incumbente; consequentemente, os subproblemas gerados a partir desse nó original são removidos da lista. Retorne ao passo 1.

Passo 4. (Ramificação). Para o problema em análise (que acabou de ser resolvido), escolha uma variável x_j que não satisfaz as condições de integralidade. Suponha que x_j assumiu um valor igual a 2.65. Particionar o problema em dois (criar dois subproblemas a partir do problema original), adicionando uma nova restrição para cada subproblema: $x_j \leq 2$ para um deles e $x_j \geq 3$ para o outro (como x_j só pode assumir valores inteiros, esse valor pode ser qualquer inteiro menor ou igual a 2 ou maior ou igual a 3). Coloque os novos subproblemas na lista L que contém todos os subproblemas que devem ser resolvidos e retorne ao passo 1.

Passo 5. (Fim). Se $z^* = -\infty$ (problema de maximização) ou $z^* = +\infty$ (problema de minimização), não existe solução factível. Caso contrário, pare, a solução incumbente x^* é ótima.

Exemplo 6.2

Aplicar o algoritmo *branch-and-bound*, utilizando a estratégia de busca em largura, para solução do seguinte problema de programação inteira:

$$\begin{aligned}
&\max z = 4x_1 + 5x_2\\
&\text{sujeito a:}\\
&\qquad 5x_1 + 7x_2 \leq 44\\
&\qquad 4x_1 + 2x_2 \leq 27\\
&\qquad 6x_1 + 3x_2 \leq 24\\
&\qquad\quad x_1, x_2 \geq 0\\
&\qquad\quad x_1, x_2 \text{ são inteiros}
\end{aligned} \tag{6.4}$$

Solução

Início

O problema (4) relaxado, sem as restrições de integralidade, é representado por (S_0). Cria-se a lista $L = \{S_0\}$. Como estamos diante de um problema de maximização, assume-se que $z^* = -\infty$; a solução incumbente é representada por $x^* = \varnothing$.

Primeira iteração

Passo 1. Como existe um subproblema na lista, vá para o passo 2.

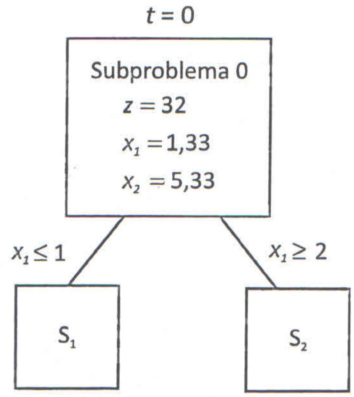
Passo 2 (Seleção). Remove-se o subproblema 0 da lista ($L = \varnothing$) e o mesmo é resolvido pelo método Simplex. O resultado da relaxação linear do problema (4) é x_1=1,33, x_2=5,33 com z=32 que corresponde ao limitante superior (o valor ótimo de z do problema de PI $\leq$ 32). Infelizmente, a solução ótima do problema relaxado não corresponde à solução ótima do problema original de PI, já que as variáveis não assumem valores inteiros.

Passo 3b. Como a solução não é inteira e o valor da função objetivo é maior que $z^* = -\infty$, o subproblema 0 será ramificado.

Passo 4 (Ramificação). S_0 é particionado em dois subproblemas: S_1 e S_2. Escolhe-se uma das variáveis (nenhuma delas satisfaz a condição de integralidade), no caso x_1. Uma nova restrição é adicionada para cada subproblema: $x_1 \leq 1$ para S_1 e $x_1 \geq 2$ para S_2. Os novos subproblemas são adicionados à lista que passa a ser representada como $L = \{S_1, S_2\}$. Retorna-se ao passo 1.

O resultado da primeira iteração está representado na Figura 6.7.

Figura 6.7 Resultado da primeira iteração.



Segunda iteração

Passo 1. Como há subproblemas na lista $(L = \{S_1, S_2\})$, o algoritmo continua.

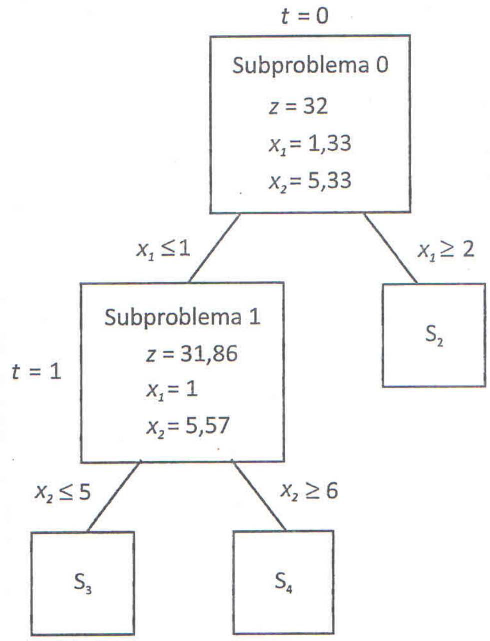
Passo 2 (Seleção). Utilizando a estratégia de busca em largura, remove-se o subproblema 1 da lista $(L = \{S_2\})$ e resolve-se o mesmo pelo método Simplex. O subproblema 1 corresponde ao subproblema 0 mais a restrição $x_1 \leq 1$.

Passo 3b. Como a solução desse subproblema não é inteira ($x_1 = 1$, $x_2 = 5{,}57$), mas seu valor de $z = 31{,}86$ é maior que $z^* = -\infty$, o subproblema 1 será ramificado.

Passo 4. Ramificação. S_1 é particionado em 2 subproblemas: S_3 e S_4. Escolhe-se a variável x_2 que não satisfaz a restrição de integralidade e adiciona-se uma restrição para cada subproblema: $x_2 \leq 5$ para S_3 e $x_2 \geq 6$ para S_4. Os novos subproblemas são adicionados à lista que passa a ser representada como $L = \{S_2, S_3, S_4\}$. Retorna-se ao passo 1.

O resultado da segunda iteração está representado na Figura 6.8.

Figura 6.8 Resultado da segunda iteração.



Terceira iteração

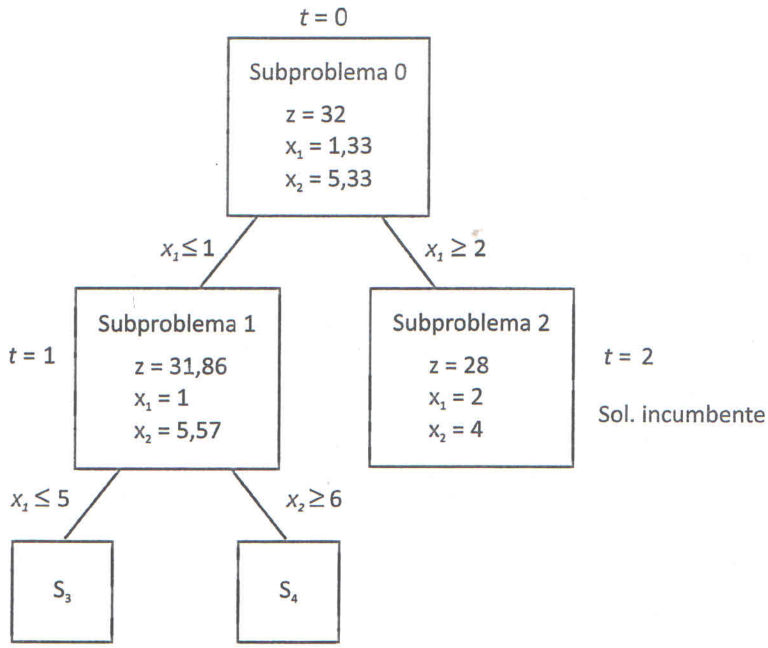
Passo 1. Como há subproblemas na lista $\left(L = \{S_2, S_3, S_4\}\right)$, o algoritmo continua.

Passo 2 (Seleção). Utilizando a estratégia de busca em largura, remove-se o subproblema 2 da lista $\left(L = \{S_3, S_4\}\right)$ e ele é resolvido pelo método Simplex. O subproblema 2 corresponde ao subproblema 0 mais a restrição $x_1 \geq 2$.

Passo 3c. Como a solução desse subproblema é inteira ($x_1=2$, $x_2=4$,) e seu valor de $z = 28$ é maior que $z^* = -\infty$, atualiza-se o valor de z^* e $\boldsymbol{x}^*$ da solução incumbente ($z^* = 28$, $x_1^* = 2$, $x_2^* = 4$). O limite inferior é, portanto, 28. Essa solução é, portanto, uma solução candidata que corresponde à solução incumbente (melhor solução encontrada até o momento). Retorna-se ao passo 1.

O resultado da terceira iteração está representado na Figura 6.9.

Figura 6.9 Resultado da terceira iteração.



Quarta iteração

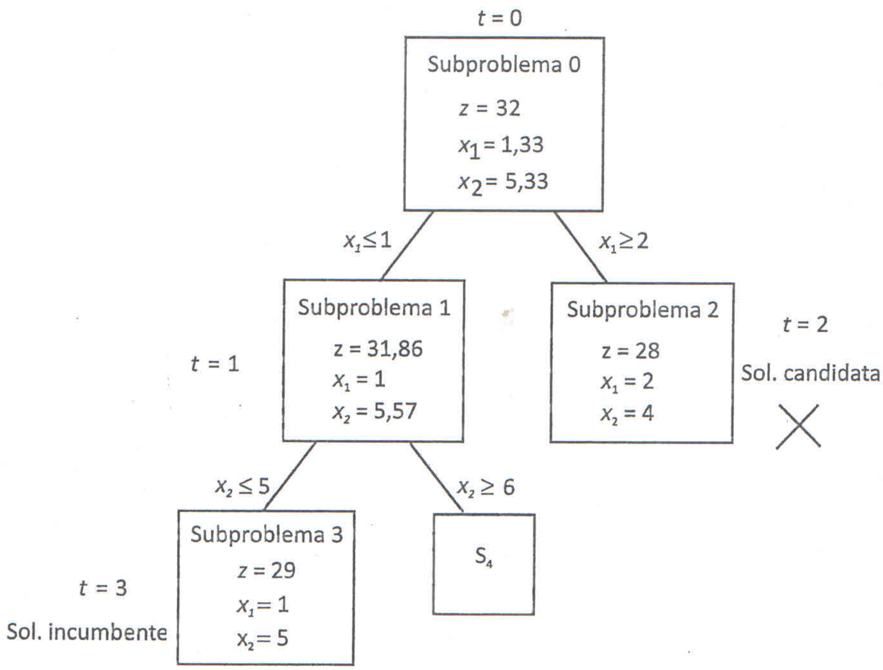
Passo 1. Como há subproblemas na lista $\left(L=\{S_3,S_4\}\right)$, o algoritmo continua.

Passo 2 (Seleção). Utilizando a estratégia de busca em largura, o próximo subproblema a ser removido da lista é S_3 $\left(L=\{S_4\}\right)$. O subproblema 3 corresponde ao subproblema 1 mais a restrição $x_2 \leq 5$ (ou o subproblema 0 mais as restrições $x_1 \leq 1$ e $x_2 \leq 5$). Ele é resolvido, então, pelo método Simplex.

Passo 3c (Eliminação). Como a solução desse subproblema é inteira ($x_1 = 1$ e $x_2 = 5$) e seu valor de $z = 29$ é maior que $z^* = 28$, atualiza-se o valor de z^* e x^* da solução incumbente ($z^* = 29$, $x_1^* = 1$, $x_2^* = 5$). Essa solução torna-se, portanto, a nova solução incumbente e a solução candidata encontrada em S_2 é descartada. O limite inferior passa a ser 29. Retorna-se ao passo 1.

O resultado da quarta iteração está representado na Figura 6.10.

Figura 6.10 Resultado da quarta iteração.



Quinta iteração

Passo 1. Como há subproblemas na lista $\left(L=\{S_4\}\right)$, o algoritmo continua.

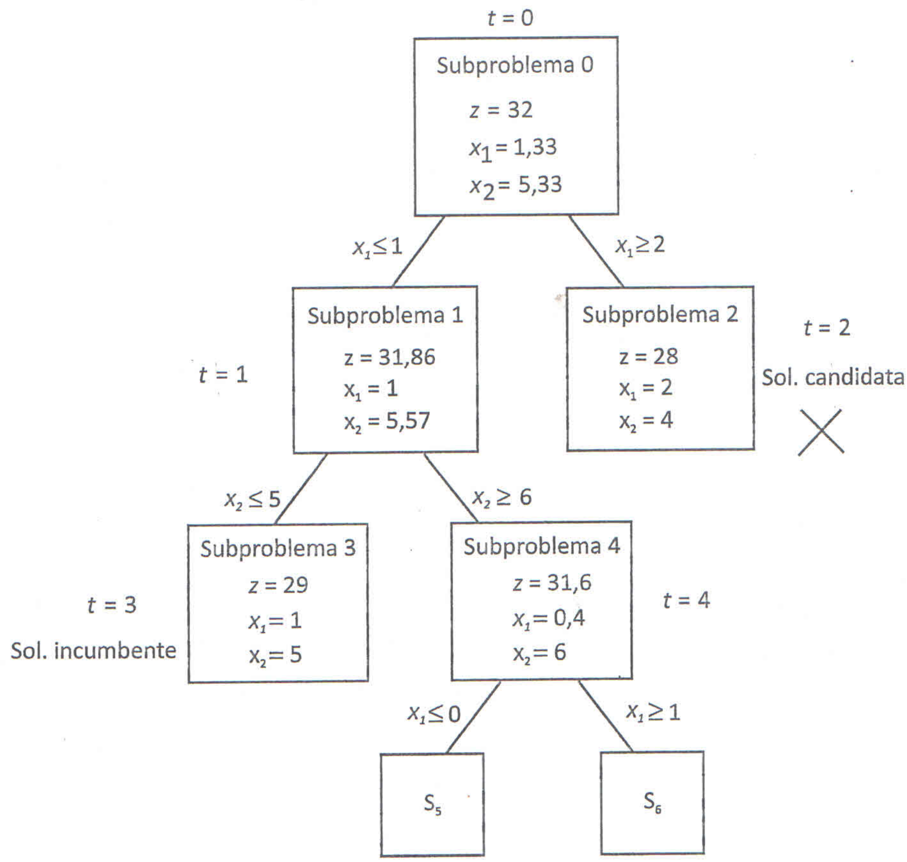
Passo 2 (Seleção). Remove-se o subproblema 4 da lista (L = ∅) que corresponde ao subproblema 1 mais a restrição $x_2 \geq 6$ (ou o subproblema 0 mais as restrições $x_1 \leq 1$ e $x_2 \geq 6$). S_4 é resolvido pelo método Simplex.

Passo 3b. Como a solução desse subproblema não é inteira, mas seu valor de z é maior que $z^* = 29$, o subproblema 4 é ramificado.

Passo 4 (Ramificação). S_4 é particionado em dois subproblemas: S_5 e S_6. Escolhe-se a variável x_1 que não satisfaz a restrição de integralidade e adiciona-se uma restrição para cada subproblema: $x_1 \leq 0$ para S_5 e $x_1 \geq 1$ para S_6. Os novos subproblemas são adicionados à lista que passa a ser representada como $L=\{S_5, S_6\}$. Retorna-se ao passo 1.

O resultado da quinta iteração está representado na Figura 6.11.

Figura 6.11 Resultado da quinta iteração.



Sexta iteração

Passo 1. Como há subproblemas na lista $\left(L=\{S_5, S_6\}\right)$, o algoritmo continua.

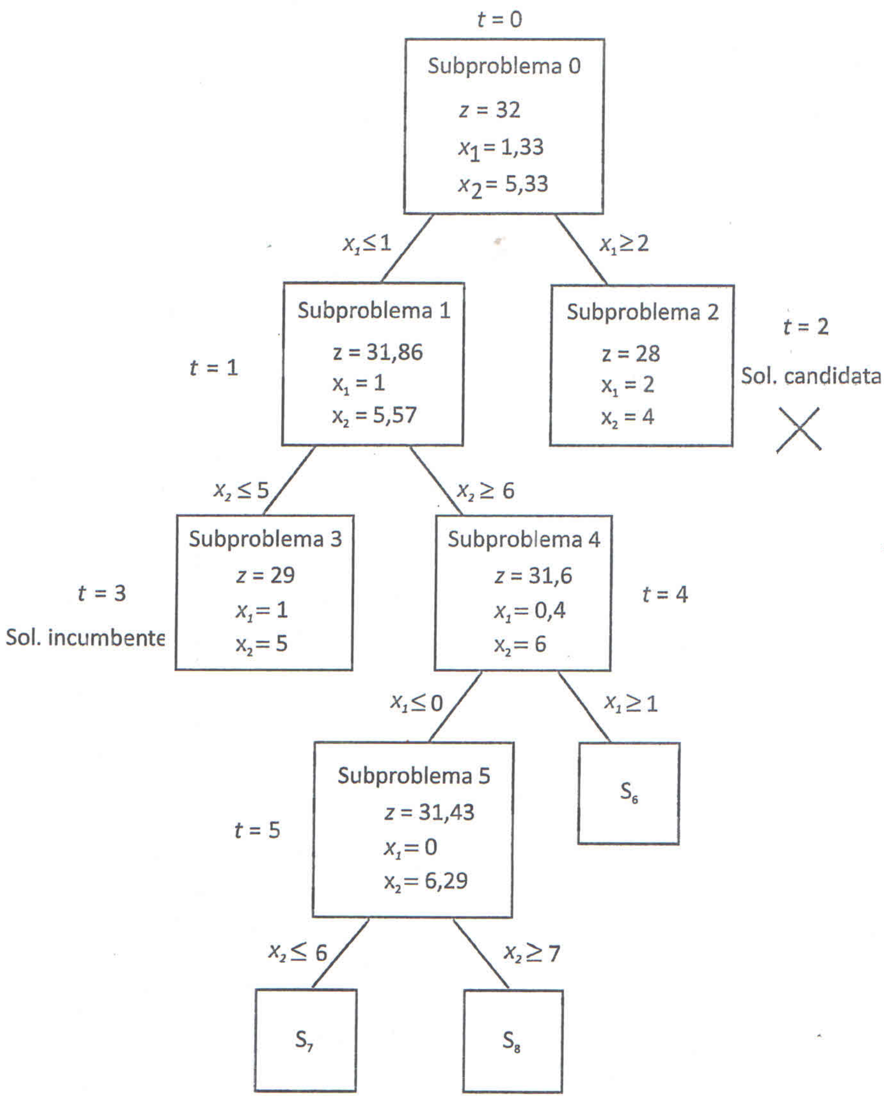
Passo 2 (Seleção). Utilizando a estratégia de busca em largura, o próximo subproblema a ser removido da lista é S_5 $\left(L=\{S_6\}\right)$ que corresponde ao subproblema 4 mais a restrição $x_1 \leq 0$. O subproblema S_5 é, então, resolvido pelo método Simplex.

Passo 3b. Como a solução desse subproblema não é inteira, mas seu valor de z é maior que $z^* = 29$, o subproblema 5 será ramificado.

Passo 4 (Ramificação). S_5 é particionado em dois subproblemas: S_7 e S_8. Escolhe-se a variável x_2 que não satisfaz a restrição de integralidade e adiciona-se uma restrição para cada subproblema: $x_2 \leq 6$ para S_7 e $x_2 \geq 7$ para S_8. Os novos subproblemas são adicionados à lista que passa a ser representada como $L=\{S_6, S_7, S_8\}$. Retorna-se ao passo 1.

O resultado da sexta iteração está representado na Figura 6.12.

Figura 6.12 Resultado da sexta iteração.



Sétima iteração

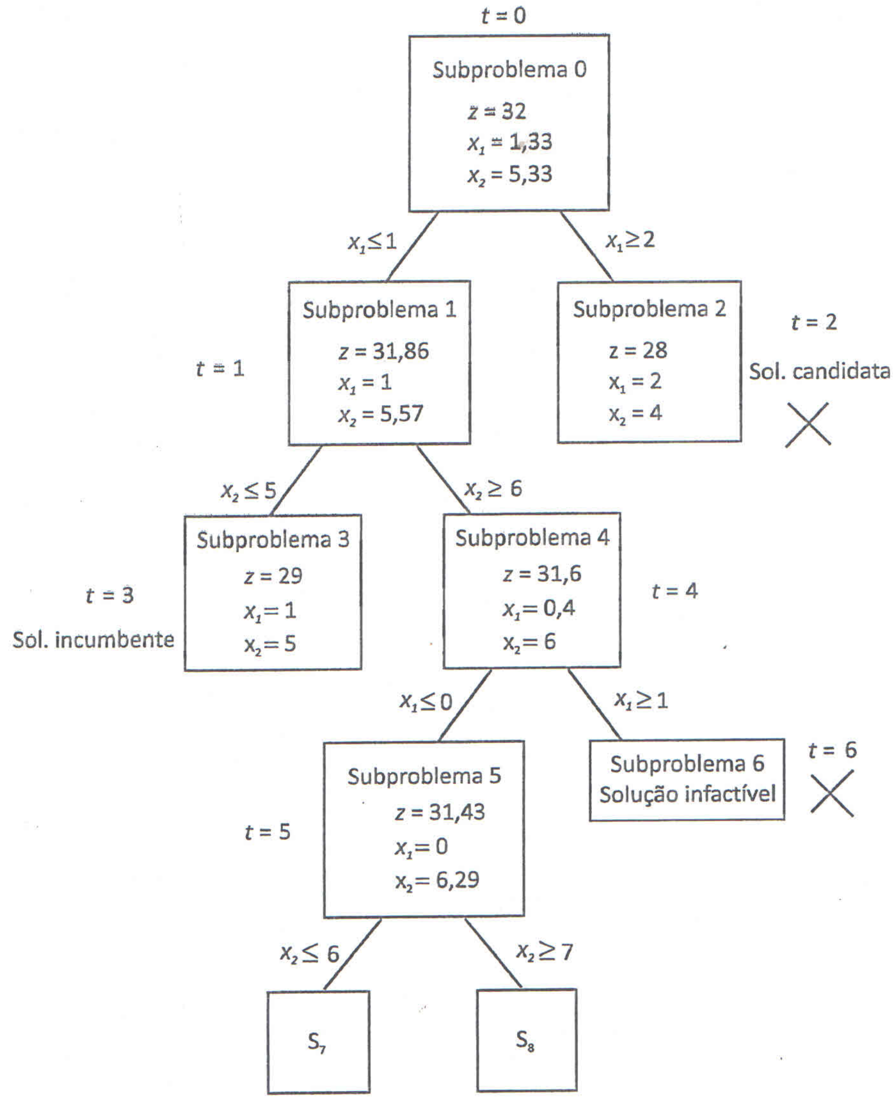
Passo 1. Como há subproblemas na lista $\left(L=\{S_6, S_7, S_8\}\right)$, o algoritmo continua.

Passos 2 (Seleção). Pela estratégia de busca em largura, o próximo subproblema a ser removido da lista é S_6 $\left(L=\{S_7, S_8\}\right)$ que corresponde ao subproblema 4 mais a restrição $x_1 \geq 1$. S_6 é, então, resolvido pelo método Simplex.

Passo 3a (Eliminação). Como a solução desse subproblema é infactível, ele é descartado. Retorna-se ao passo 1.

O resultado da sétima iteração está representado na Figura 6.13.

Figura 6.13 Resultado da sétima iteração.



Oitava iteração

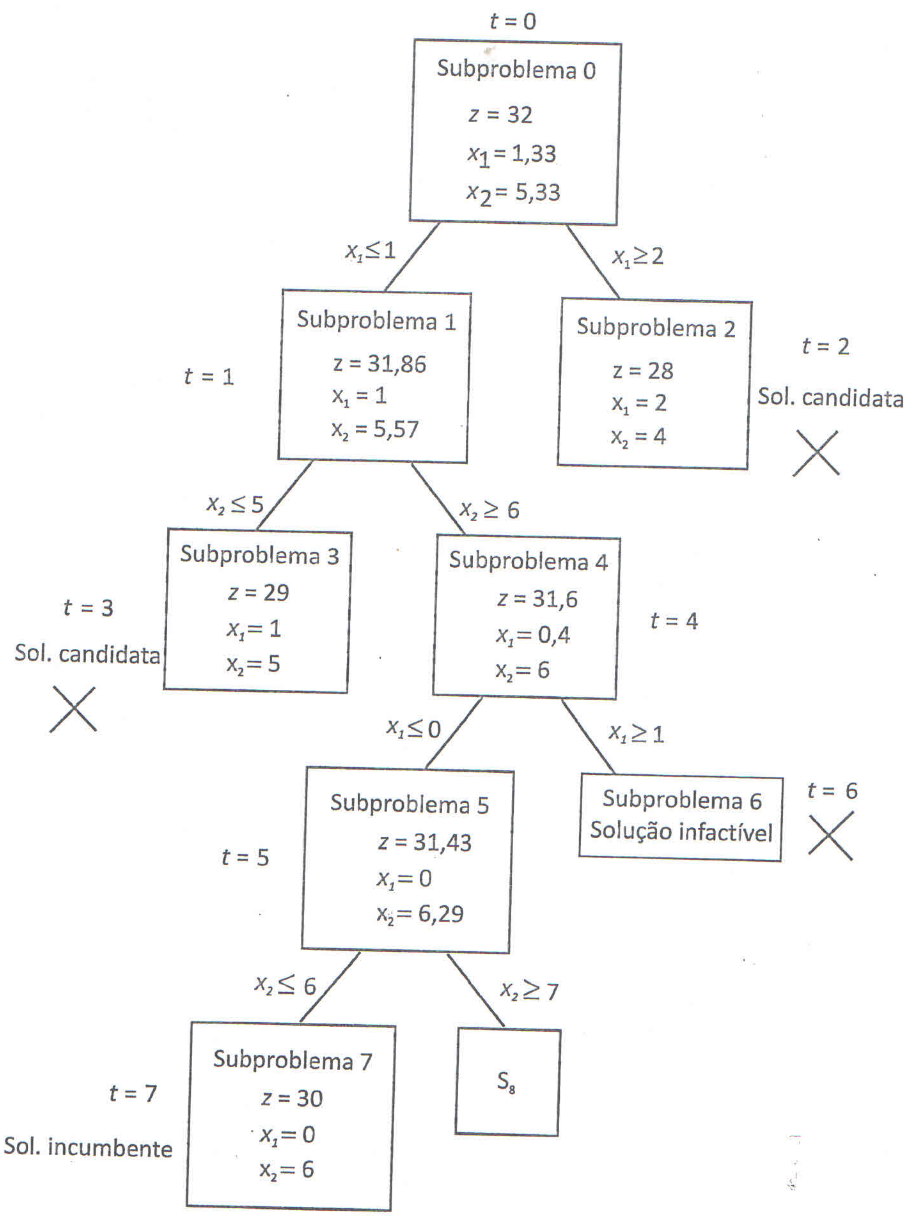
Passo 1. Como há subproblemas na lista $\left(L=\{S_7,S_8\}\right)$, o algoritmo continua.

Passo 2 (Seleção). O próximo subproblema a ser removido da lista é S_7 $\left(L=\{S_8\}\right)$ que corresponde ao S_5 mais a restrição $(L=\{S_8\})$; o mesmo é resolvido pelo método Simplex.

Passo 3c (Eliminação). Como a solução desse subproblema é inteira e seu valor da função objetivo $z = 30$ é maior que $z^* = 29$, atualiza-se o valor de z^* e x^* da solução incumbente ($z^* = 30$, $x_1^* = 0$, $x_2^* = 6$). Essa solução torna-se, portanto, a nova solução incumbente e a solução candidata encontrada em S_3 é descartada. O limite inferior passa a ser 30. Retorna-se ao passo 1.

O resultado da oitava iteração está representado na Figura 6.14.

Figura 6.14 Resultado da oitava iteração.



Nona iteração

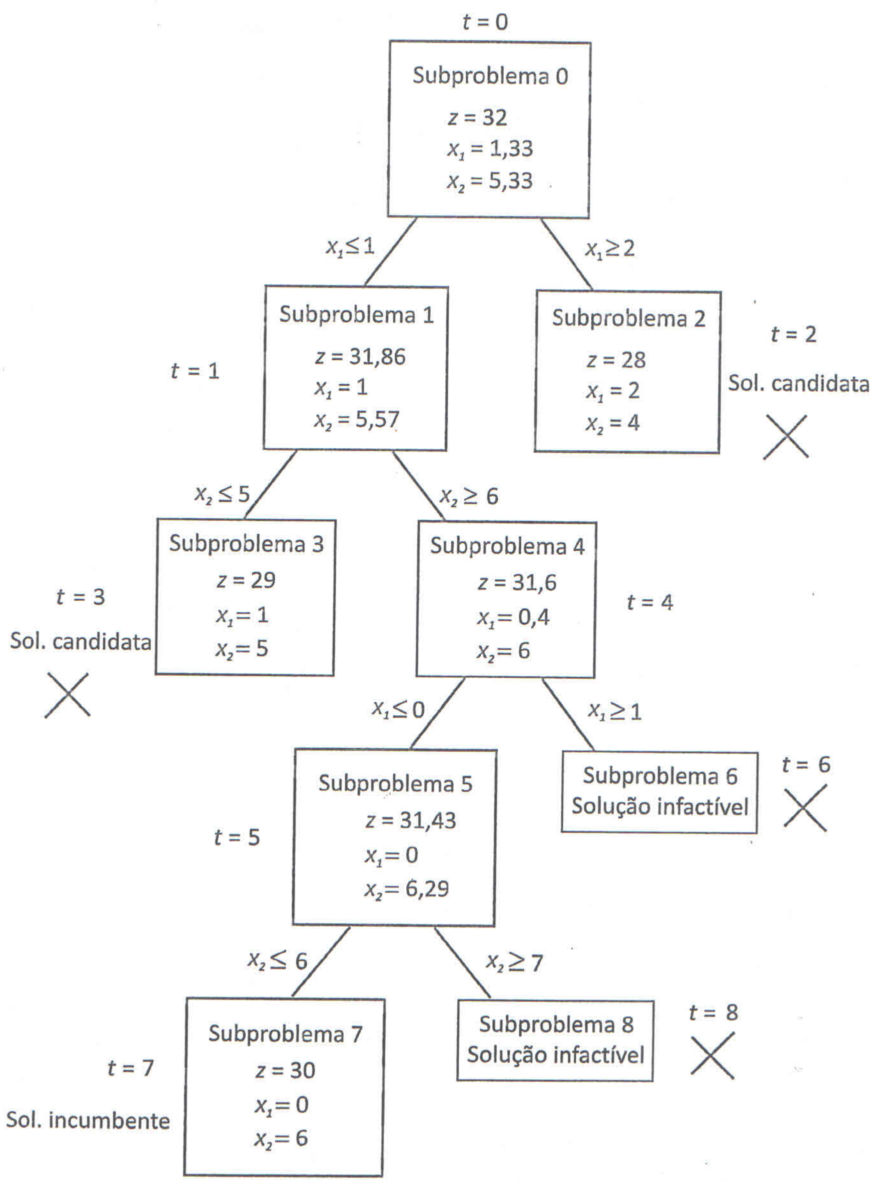
Passo 1. Como há subproblemas na lista $(L = \{S_8\})$, o algoritmo continua.

Passo 2 (Seleção). Remove-se o único elemento da lista $(L = \varnothing)$ e S_8 é resolvido pelo método Simplex.

Passo 3a (Eliminação). Como a solução desse subproblema é infactível, ele é descartado. Retorna-se ao passo 1.

O resultado da nona iteração está representado na Figura 6.15.

Figura 6.15 Resultado da nona iteração.



Décima iteração

Passo 1. Como não há subproblemas na lista ($L = \varnothing$), vá para o passo 5.
Passo 5. (Fim). A solução incumbente (S_7) é ótima: $x_1^* = 0$, $x_2^* = 6$ com $z^* = 30$.

Exemplo 6.3

Aplicar o algoritmo *branch-and-bound*, utilizando a estratégia de busca em profundidade, para solução do mesmo problema de programação inteira apresentado no Exemplo 6.2:

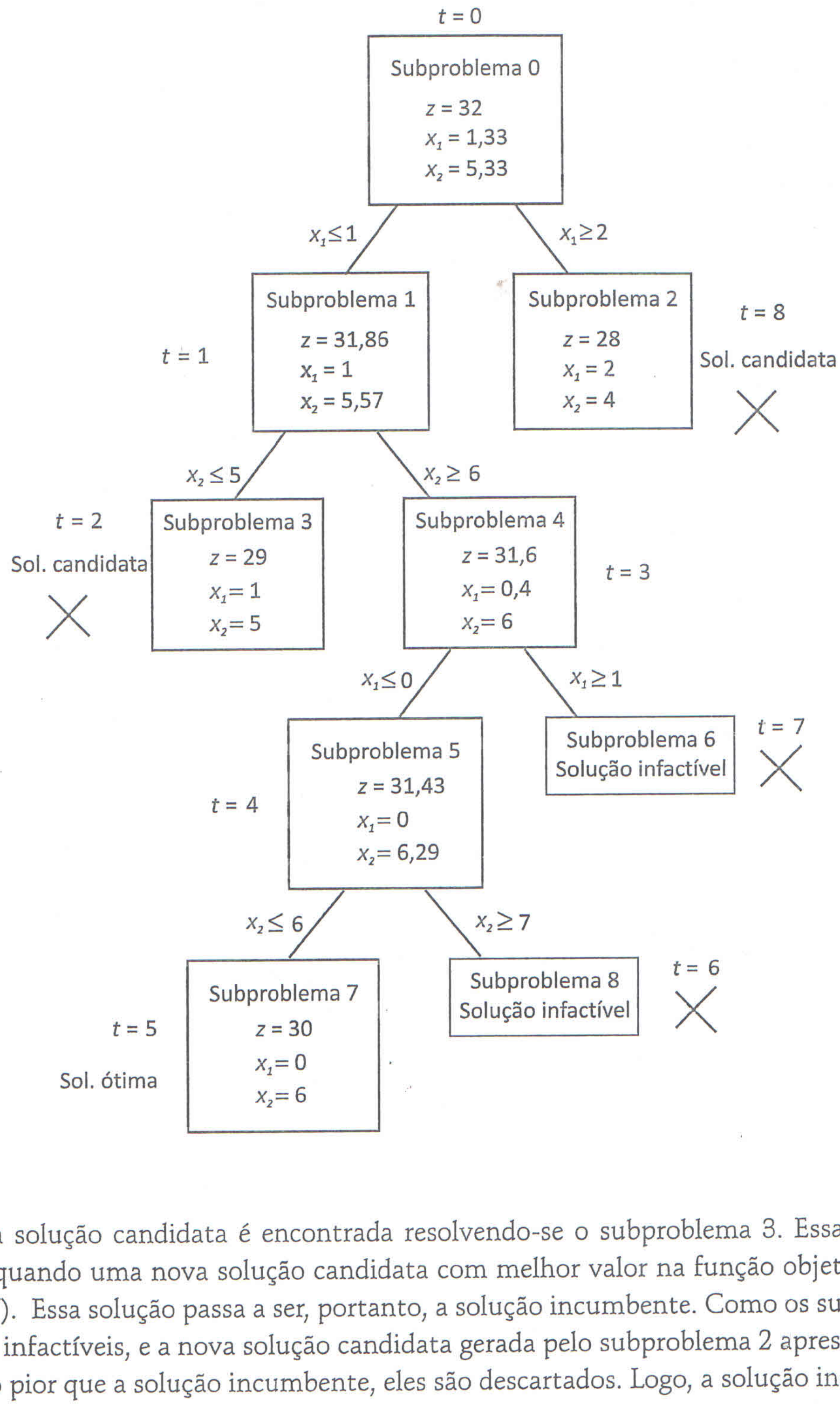
$$
\begin{aligned}
\max z = 4x_1 + 5x_2 & \\
5x_1 + 7x_2 &\leq 44 \\
4x_1 + 2x_2 &\leq 27 \\
6x_1 + 3x_2 &\leq 24 \\
x_1, x_2 &\geq 0 \\
x_1, x_2 & \text{ são inteiros}
\end{aligned}
\tag{6.4}
$$

➡ Solução

Os subproblemas gerados pela estratégia de busca em profundidade, coincidentemente, serão os mesmos que aqueles gerados pela estratégia de busca em largura (inclusive a ordem com que esses subproblemas são criados). Isso nem sempre acontece. Por outro lado, a ordem em que esses subproblemas são resolvidos difere de uma estratégia para outra.

A Figura 6.16 apresenta o resultado do algoritmo *branch-and-bound* para solução do exemplo anterior, utilizando a estratégia de busca em profundidade.

Figura 6.16 Algoritmo *branch-and-bound* usando a estratégia de busca em profundidade.



A primeira solução candidata é encontrada resolvendo-se o subproblema 3. Essa solução passa a ser descartada quando uma nova solução candidata com melhor valor na função objetivo é encontrada (subproblema 7). Essa solução passa a ser, portanto, a solução incumbente. Como os subproblemas 8 e 6 geram soluções infactíveis, e a nova solução candidata gerada pelo subproblema 2 apresenta um valor da função objetivo pior que a solução incumbente, eles são descartados. Logo, a solução incumbente gerada pelo subproblema 7 é ótima.