OTIMIZAÇÃO VIA BRANCH-AND-BOUND: TEORIA, ALGORITMOS E EXEMPLOS DE APLICAÇÃO

https://doi.org/10.22533/at.ed.654112526022

Data de aceite: 25/02/2025

Joelson Lopes da Paixão
Programa de Pós-Graduação em Engenharia Elétrica
Universidade Federal de Santa Maria – UFSM
Santa Maria - RS, Brasil

Gabriel Henrique Danielsson
Programa de Pós-Graduação em Engenharia Elétrica
Universidade Federal de Santa Maria – UFSM
Santa Maria - RS, Brasil

Alzenira da Rosa Abaide
Programa de Pós-Graduação em Engenharia Elétrica
Universidade Federal de Santa Maria – UFSM
Santa Maria - RS, Brasil

RESUMO: Este capítulo apresenta uma análise teórica e aplicada do método *Branch-and-Bound* (B&B) e sua relevância na otimização de sistemas de engenharia elétrica, com ênfase na resolução de problemas de programação linear inteira (PLI). O B&B é um método estruturado que organiza a busca pela solução ótima por meio da subdivisão do problema original em subproblemas menores, utilizando relaxações, heurísticas e estratégias de poda para eliminar regiões inviáveis do espaço de busca. Esse processo reduz significativamente o esforço computacional e melhora a eficiência da resolução. Inicialmente, são discutidos os fundamentos teóricos do método, abordando sua estrutura algorítmica e critérios de avaliação, bem como as estratégias de busca empregadas para garantir soluções ótimas. Em seguida, é demonstrada uma aplicação prática utilizando o software LINDO (*Linear Interactive and Discrete Optimizer*), ferramenta amplamente utilizada na pesquisa operacional. Para validar a eficácia do método, um estudo de caso detalhado é apresentado, envolvendo a resolução de um problema de maximização em PLI, com construção da árvore de enumeração, definição de limites superiores e inferiores e eliminação de subproblemas inviáveis. Os resultados confirmam que o método B&B é uma abordagem robusta e eficaz para a resolução de problemas combinatórios e de alocação de recursos, sendo amplamente utilizado na engenharia, ciência de dados e pesquisa operacional. Sua capacidade de

explorar seletivamente o espaço de busca e evitar a enumeração exaustiva o torna essencial para a obtenção de soluções ótimas em sistemas complexos.

PALAVRAS-CHAVE: Otimização; Programação Linear Inteira; *Branch-and-Bound.*

1 | INTRODUÇÃO

O método *Branch-and-Bound* (B&B) é amplamente aplicado à programação linear inteira (PLI), sendo fundamental para a solução de problemas onde as variáveis devem assumir valores inteiros. Problemas de Programação Inteira Pura (PIP) podem apresentar um número extremamente elevado de soluções viáveis, tornando inviável a abordagem exaustiva para a determinação da solução ótima. Assim, o método B&B busca estruturar o espaço de soluções de modo a analisar apenas uma fração reduzida das combinações possíveis, garantindo eficiência computacional na busca pela solução ótima [1].

O princípio central do método consiste na divisão do conjunto de soluções viáveis, fragmentando o problema original em subproblemas menores, chamados de nós da árvore de decisão. Cada subproblema gerado pode ser limitado, descartando-se aqueles que não contêm a solução ótima, reduzindo o espaço de busca e acelerando o processo de otimização [1,2]. Esse procedimento torna o B&B um dos métodos mais apropriados para problemas de programação linear inteira, especialmente em casos em que as variáveis são binárias (0 ou 1), comuns em aplicações de engenharia e ciência da computação [2].

A abordagem adotada neste capítulo busca apresentar, de forma estruturada, os conceitos fundamentais do método B&B, desde seus princípios teóricos até sua implementação prática. Além de explorar a lógica matemática subjacente, o capítulo também demonstra um exemplo aplicado ao software LINDO, permitindo visualizar a resolução de um problema real de PLI e comparar a qualidade dos resultados obtidos.

2 | ALGORITMO *BRANCH-AND-BOUND*

O algoritmo B&B é muito usual na resolução de problemas lineares, principalmente do tipo programação inteira pura (PLIP), onde todas as variáveis são inteiras, ou programação inteira mista (PLIM), no caso varia entre variáveis inteiras ou contínuas reais [5].

O método consiste em solucionar o problema para as cadeias menores, onde cada subproblema gera um limite inferior (problema de minimização) ou superior (problema de maximização) sobre o valor da função objetivo (FO). As soluções dos subproblemas devem ser distribuídas até que seja obtida a solução ótima do problema original [3].

O algoritmo B&B apresenta uma estrutura básica em formato de árvore, onde os nós representam os subproblemas e os ramos interconectando os nós, representando as restrições a serem adicionadas ao sistema [3].

Umas das formas de apresentar o método é descrever as três etapas básicas para o desenvolvimento do algoritmo [1]:

- **Ramificação:** Entre com os subproblemas ainda sem avaliação, selecione aquele que foi criado mais recentemente. Desempates feitos de acordo com aquele que tiver o maior limite. Ramifique a partir do nó para esse subproblema para criar dois subproblemas novos, fixando a próxima variável em (0 ou 1).
- **Limitação:** Para cada novo subproblema, obtenha seu limite aplicando o método simplex ao seu relaxamento PL e arredondando para baixo o valor de Z para a solução ótima resultante.
- **Avaliação:** Para cada novo subproblema, aplique os três testes de avaliação sintetizados anteriormente e descarte aqueles subproblemas que são avaliados por qualquer um dos testes.

Utilizando como exemplo um problema de programação binária [3]:

$$Min\ z = 2x1 + 3x2 + 2x3$$

Sujeito a:

$$2x1 + 2x2 \geq 2;$$
$$6x1 + 4x2 \geq 2;$$
$$x1, x2\ e\ x3\ \epsilon\ \{0, 1\}$$

Na Fig. 1 apresenta o problema raiz relaxado sem considerar as restrições de que as variáveis de decisão são binárias, representado por (S_0). Conforme descemos a árvore, as restrições novas são acrescentadas. O subproblema 1, no caso, representa o problema raiz relaxado (S_0) adicionado a restrição $x_1 = 0$. O subproblema (S_6) representa as restrições $x_1 = 1$ e $x_2 = 1$, ou o subproblema 2 mais a restrição $x_2 = 1$; assim sucessivamente, originando a árvore de enumeração completa ou explícita, apresentando todas as soluções candidatas do problema raiz [3].

A solução é dita candidata quando todas as variáveis assumem valores binários ou quando a PLI assume todos os valores inteiros [3].

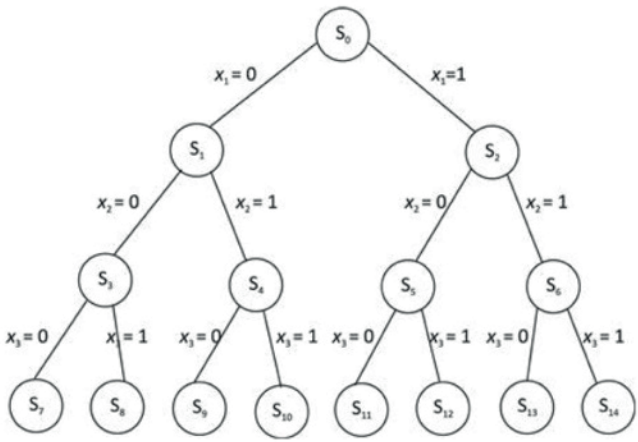

Fig. 1 - Árvore de enumeração completa ou explícita.

Porém é usual utilizar limites superiores e inferiores, pois não é prático considerar todas as possíveis soluções, descartando subproblemas infactíveis que não apresentem a solução ótima (enumeração implícita).

Para problemas de minimização, o limite superior pode ser buscado através de heurísticas que gerem soluções eficientes, ou também a melhor solução encontrada pelo algoritmo até o momento; diferente no caso para buscar o limite inferior, que poderá ser encontrado pela relaxação do problema, descartando uma ou mais restrições.

Para problemas de maximização, é o inverso, o limite inferior é encontrado através de heurísticas eficientes, ou a melhor solução encontrada pelo algoritmo até o momento e para o limite superior pela relaxação do problema.

A próxima etapa é definir qual o subproblema é candidato, para isso é necessário percorrer a árvore e escolher. A melhor estratégia é:

- Busca por profundidade: escolhe o nó gerado mais recentemente e mais profundo da árvore, dessa forma encontra-se de um lado da árvore uma solução candidata.
- Busca por largura: escolhe o nó de nível mais antigo e mais alto ainda não investigado, e quando esse nó é ramificado, resolve-se todos os subproblemas gerados por essa ramificação, porém esse tipo de busca pode ter um gasto excessivo de memória;
- Busca pelo nó promissor: escolhe o nó com o melhor valor da F.O, ou seja, o que tem maior probabilidade de encontrar a solução ótima.

A Fig. 2 apresenta o problema de programação binária da Fig. 1 na busca em profundidade, onde os números contidos nos nós representam a ordenação com que os problemas seriam solucionados.

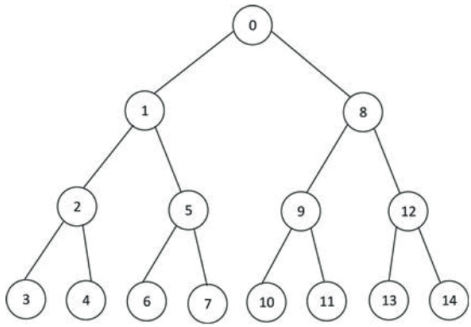

Fig. 2 – Estratégia de busca em profundidade para o problema da Fig. 1.

Na Fig. 3 apresenta como a busca em largura seria para o problema binário da Fig. 1.

Outra estratégia seria a busca em largura e a do nó mais promissor de forma mista, de forma que a cada nível inicia-se com o nó anterior (mais antigo), produzido pelo nó mais promissor. Exemplo:

O nós 1 e 2 do primeiro nível são gerados pelo problema raiz relaxado, então escolha o nó mais antigo, que seria o nó 1. Usando a estratégia de busca em largura, o seguinte nó é o 2, analise qual desses nós apresenta o melhor valor da F.O. Se escolheu o nó 2, o próximo escolhido será o nó 5, pois é o mais antigo e produzido pelo nó mais promissor, continuando a estratégia, segue-se para os nós 6, 3, 4 (passando por todos os nós no mesmo nível da árvore) continuadamente.

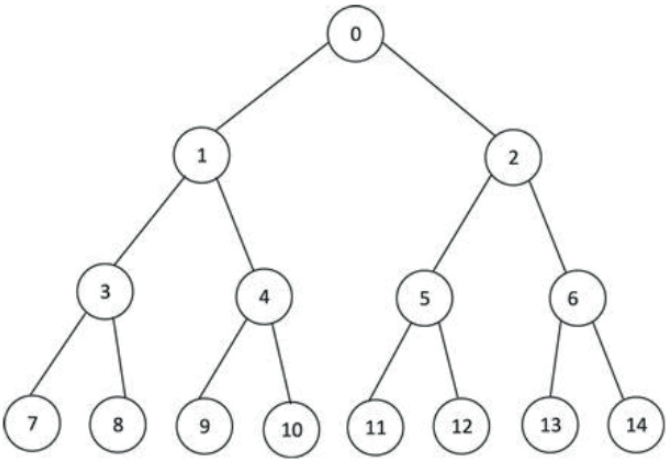

Fig. 3 – Estratégia de busca em largura para o problema da Fig. 1.

Temos na Fig. 4 a árvore de enumeração completa que já pode ser podada nos casos que os subproblemas são infactíveis ou não dispõem da solução ótima, usando limites superiores e inferiores, dessa forma adotando a estratégia de busca em largura, onde *t* representa a ordem que os problemas são solucionados. Podemos observar que nas Fig. 2 e 3 *t* representa o número presente; que agora será representado como na Fig. 4.

Então resolvendo o problema o problema raiz relaxado S_0, desconsiderando as restrições como as variáveis são binárias, chega-se ao limite inferior de z = 2,33, acarretando que o valor ótimo de z não pode ser inferior a 2,33 do problema raiz de PB. Considere que foi originado uma heurística para o problema, chegando a solução:

$$x1 = 1, x2 = 0 \; com \; z = 4{,}5$$

Esse valor de z será o limitante superior, ou seja, indica que o valor ótimo máximo de z será 4,5 do problema de PB, representando a solução incumbente, considerada a melhor encontrada até a ocasião, de forma que qualquer subproblema com valor superior a 4,5 é descartado. Solucionado o subproblema 0, segue para o subproblema 1, acompanhado dos subproblemas 2 e 3. Como o problema 3 chegou-se a uma solução infactível, descarta-se, pois se novas restrições forem acrescidas, o problema continuará infactível. Mesmo caso os subproblemas 7 e 9, descartados por gerarem soluções infactíveis. Logo o subproblema 6 produziu uma solução candidata, onde todas as variáveis levam valores binários,

entretanto inadequada por apresentar comportamento inferior que a solução incumbente z > 4,5, então descartada, não adiantando acrescentar restrições ao subproblema 6 produzirá soluções redundantes ou inferiores. De mesma forma o subproblema 8. Solucionando o subproblema 10, chega-se à solução igual a incumbente produzida pela heurística, onde podemos finalizar que essa solução é ótima.

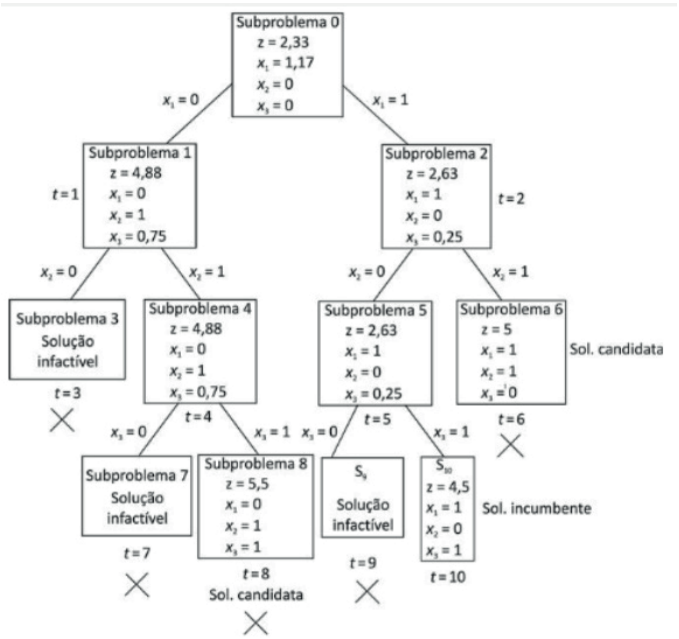

Fig. 4 – Podando a árvore por infactibilidade e pelo subproblema não ser promissor.

3 | EXEMPLO DE MODELAGEM E RESOLUÇÃO

Aplicando a técnica B&B em um problema de PLI puro, aplicando a técnica demonstrada nas referências desse artigo.

$$Max\ Z = 5{,}x1 + 8x2$$

sujeito a

$$x1 + x2 \leq 6$$
$$5x1 + 9x2 \leq 45$$
$$x1; x2\ \epsilon\ Z_{+}$$

Traçando as duas restrições na Fig. 5, temos o gráfico a região com as soluções viáveis inteiras:

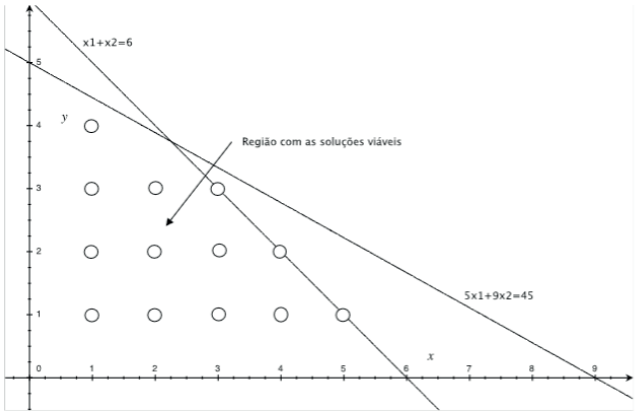

Fig. 5 – Restrições traçadas e região com soluções viáveis.

Utilizando-se do método gráfico, podemos lançar um valor para Z, por exemplo 20 e achamos a solução ótima para o problema relaxado, Fig. 6.

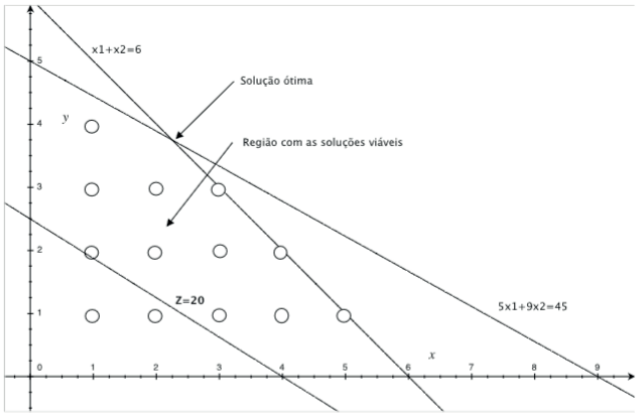

Fig. 6 – Solução ótima do problema relaxado.

Na Fig. 7, substituindo os valores do ponto (2,25; 3,75), encontramos Z= 41,25, que no caso é a solução relaxada, mas não a solução do PLI. No caso o Z calculado é o limitante superior da solução como pode ser observado no gráfico da Fig. 7.

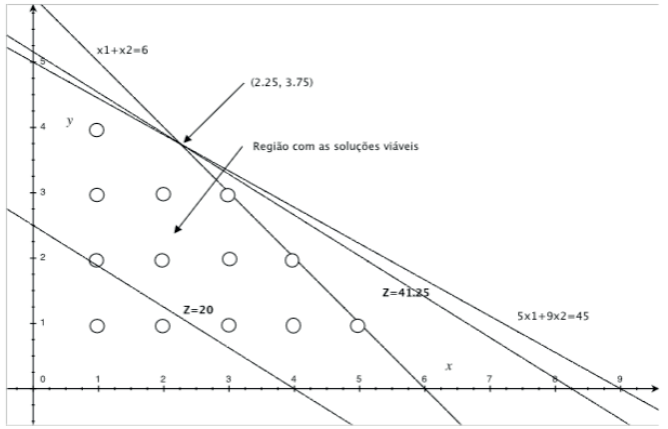

Fig. 7 – Valor de Z no gráfico.

Na Fig. 9 a simulação da interação do problema relaxado através do *Software* LINDO.

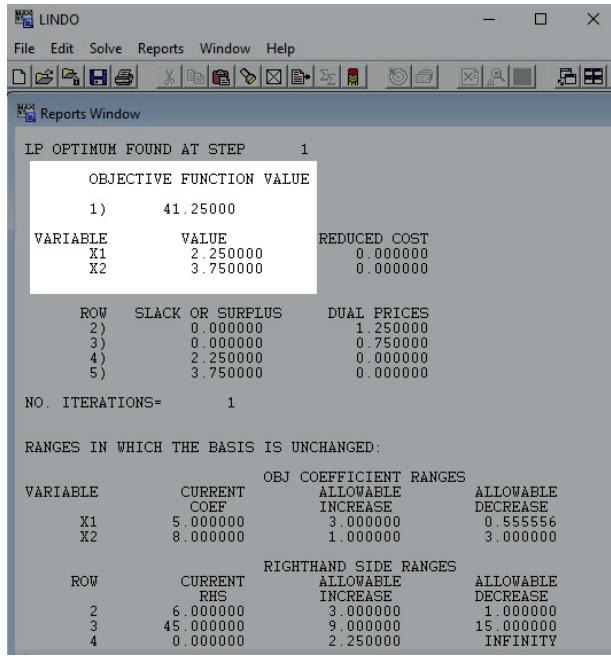

Fig. 9 – Resultado do problema relaxado.

A ideia para a solução desse problema é particionar a região viável da solução relaxada usando o método *B&B*, resolvendo em subproblemas e tornando-o cada vez menor, até obter a solução ótima, ou seja, a solução inteira ótima.

Escolhendo umas das variáveis x1 ou x2, seleciono x2, que é uma variável que

apresenta o valor fracionado dessa solução ótima relaxada, onde cada ponto inteiro viável deve ter x2 ≤ 3 ou x2 ≥ 4, que irão ser as restrições. Com isso podemos criar dois subproblemas conforme Tabela 1:

Subproblema 1	Subproblema 2
Max Z= 5x1+8x2	Max Z= 5x1+8x2
s.a	s.a
x1+x2≤6	x1+x2≤6
5x1+9x2≤45	5x1+9x2≤45
x2≤3	x2≥4
x1, x2≥0	x1≥0

Tabela 1 – Subproblemas 1 e 2

Logo realizando as interações dos subproblemas, encontramos as soluções conforme as Fig. 10 e 11 respectivamente.

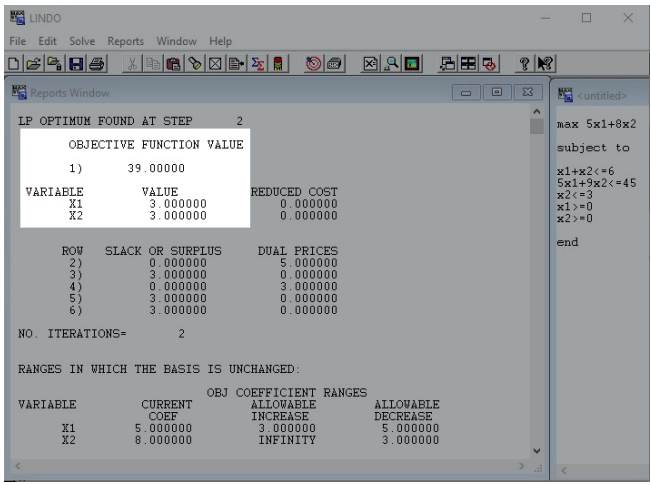

Fig. 10 – Resultado do subproblema 1.

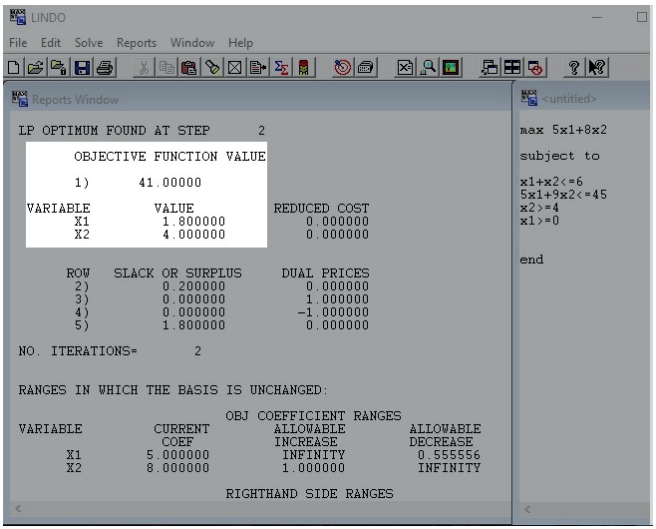

Fig. 11 – Resultado do subproblema 2.

Representando no gráfico da Fig. 12 agora as regiões com soluções viáveis delimitadas pelas restrições dos subproblemas 1 e 2, obtemos um espaço no gráfico que não é mais viável, a região não hachurada.

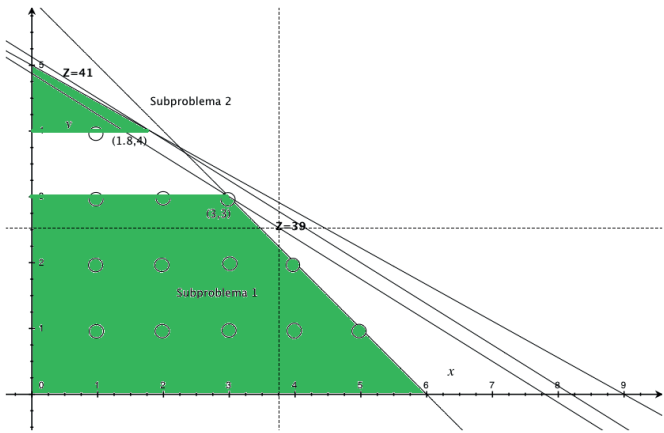

Fig. 12 – Gráfico com delimitações das regiões impostas de S1 e S2.

A primeira solução gerada onde são irrestritas ao topo da árvore, Fig. 13, e na sequência as ramificações com o primeiro e segundo subproblema para obter o histórico da resolução. Na solução 1, encontramos a primeira solução inteira (incumbente) que satisfaz todas as restrições de integralidade das variáveis do problema original; na solução 2, o valor maior que a solução 1, valor muito bom, mas ainda apresenta em x1 uma variável com valor contínuo. Ramificar a solução 1 também não agrega, pois já temos uma solução com valores inteiros, mas não melhor que a da incumbente.

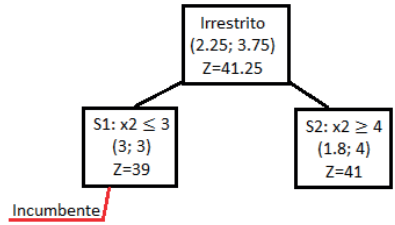

Fig. 13 – Árvore com os subproblemas 1 e 2.

Se o valor da solução 2 fosse um valor menor que da solução 1, não teria o porquê de continuar, então Z≥39 da solução 1 será o valor limitante inferior e o Z≤41o limitante superior da PLI, não mais o valor Z=41.25 encontrado no início. Dessa forma, prosseguindo ramificando a S2, escolhemos x1 para colocar na restrição, pois x2 já tem um valor inteiro.

A Tabela 2, apresenta os subproblemas com as novas restrições que podem ser verificadas no gráfico da Fig. 12 (a parte vazia entre as hachuras em verde).

Subproblema 3	Subproblema 4
Max Z= 5x1+8x2	Max Z= 5x1+8x2
S.a	S.a
x1+x2≤6	x1+x2≤6
5x1+9x2≤45	5x1+9x2≤45
x2≥4	
x1≤1	x2≥4
x1≥0	x1≥2

Tabela 2 – Subproblemas 3 e 4

Resolvendo os subproblemas temos na Fig. 14 a solução do subproblema 3, com Z= 40.55 e (1; 4.44) para x1 e x2. Já o subproblema 4 é infactível.

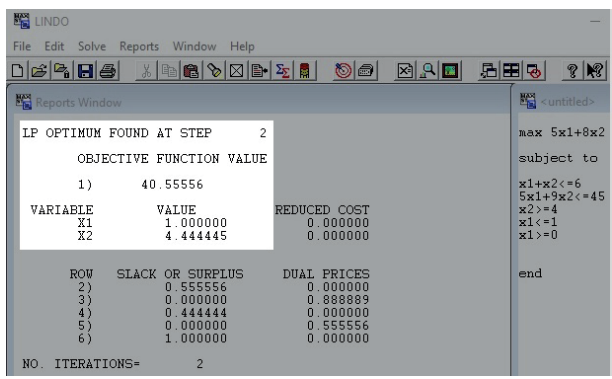

Fig. 14 – Resultado da interação do subproblema 3.

Na área hachurada em verde no gráfico da Fig. 15 é a solução viável delimitada imposta pelas restrições do subproblema 3 e a área infactível em azul do subproblema 4, imposta pelo x≥2.

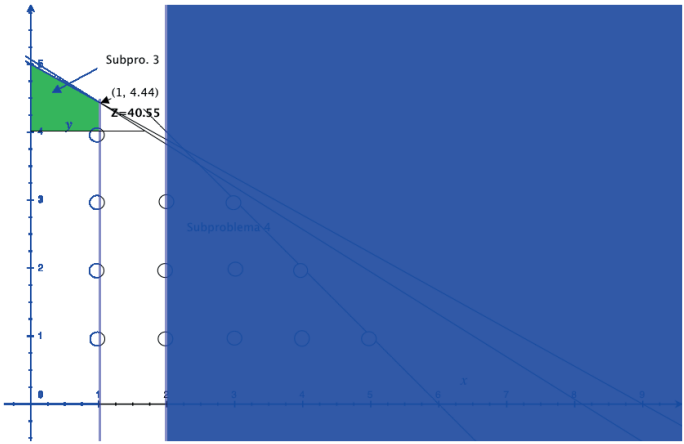

Fig. 15 – Gráfico das soluções viáveis do subproblema 3.

A solução do subproblema 3 relaxado é agora Z=40.55, conforme a árvore da Fig. 16, tornando-se o novo limitante superior, mas ainda temos uma variável contínua que é X2 (4.44). Buscamos ainda uma solução inteira, que o idealizamos que seja maior que Z=39, dessa forma aumentaríamos a lucratividade (se o cálculo fosse para achar uma solução ótima para lucros de uma empresa).

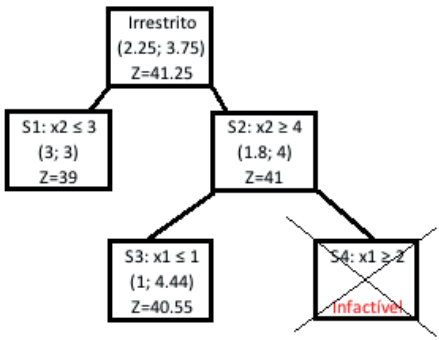

Fig. 16 – Árvore com os subproblemas 3 e 4.

Ramificando o subproblema 3, obtém-se no subproblema 5 a restrição (x2≥4) que já apresentava nas interações anteriores. Então no caso não faz mais sentido; retiramos essas restrições e colocamos a restrição (x=4). No subproblema 6 também eliminamos a restrição (x2≥4) por também não fazer mais sentido.

Subproblema 5	Subproblema 6
Max Z= 5x1+8x2	Max Z= 5x1+8x2
s.a	s.a
x1+x2≤6	x1+x2≤6
5x1+9x2≤45	5x1+9x2≤45
x2≥4	x2≥4
x1≤1	x1≤1
x2≤4	x2≥5
x2=4	
x1≥0	x1≥0

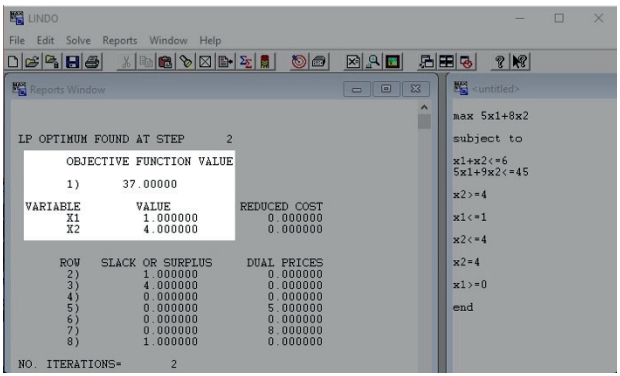

Fig. 17 – Resultado da interação do subproblema 5.

A nova restrição em S5 é x≤4, então quando colocamos x=4 a região viável está no segmento (0; 4) em vermelho no gráfico da Fig. 18. A Fig. 17 apresenta o resultado do subproblema 5 com x1 e x2 (1;4) com solução ótima Z=37. Esse valor é descartado, pois é menor que a incumbente, ou seja, 37 ≤ 39.

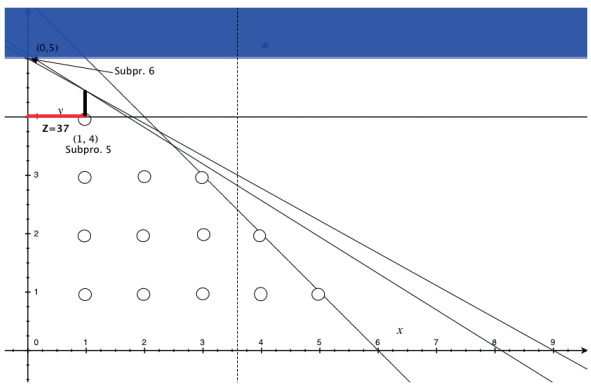

Fig. 18 – Gráfico das soluções viáveis do subproblema 5 e 6.

No subproblema 6 temos somente um ponto devido a restrição imposta x≥5, onde a região viável é o ponto (0, 5) com solução ótima Z=40, Fig. 19, onde este é uma solução inteira maior que a incumbente anterior, substituindo-a.

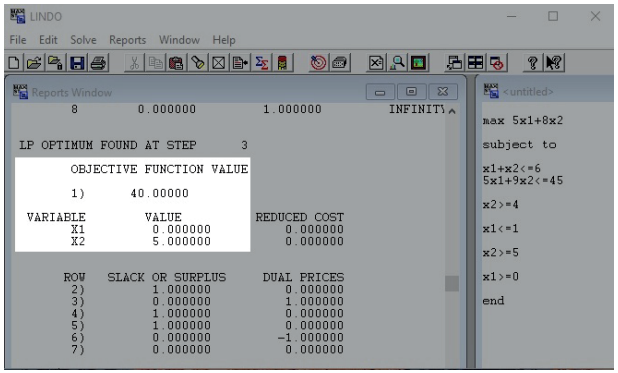

Fig. 19 – Resultado da interação do subproblema 6.

Assim na Fig. 20, a árvore com todas as ramificações e soluções possíveis na Fig. 20.

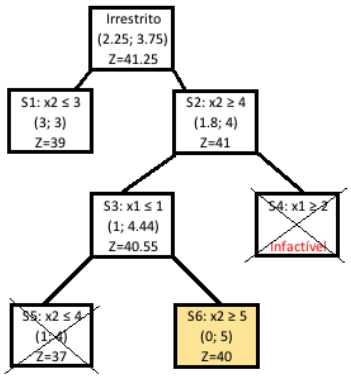

Fig. 20 – Gráfico com delimitações das

Sem subproblemas a serem ramificados, a nova incumbente é a solução ótima para o PLI.

O LINDO dispõe de uma linha de comando gin2, que calcula pelo método B&B, facilitando muito o operador do *software* na hora de verificar a exatidão da solução ótima por este método de otimização.

4 I CONCLUSÕES

Neste artigo foi estudado o método B&B, colocando seus principais conceitos para resolução de problemas de PLI, destacando sua técnica em gerar subproblemas no formato

de uma árvore, onde os nós representam os subproblemas e os ramos as novas restrições a serem impostas ao problema.

Portanto para o entendimento do método, foi utilizado um problema de programação binária, demonstrando desde o problema relaxado com sua solução ótima contínua até a aplicação do método, montando a árvore de soluções factíveis, com os subproblemas e suas restrições, até chegar na solução viável ótima inteira. Dessa forma para aplicar o conhecimento, pegamos um problema da maximização da função objetivo, onde foi necessário apresentar seis subproblemas, calculando com o *software* LINDO, a fim de praticar ao mesmo tempo o uso desse *software,* colocando todos os resultados em forma gráfica para melhor entendimento das restrições impostas pelo problema.

Por fim, o método apresenta um algoritmo eficiente, apesar de ter muitas etapas, que podem ser muito extensas e complexas dependendo do problema.

REFERÊNCIAS

[1] F. S. Hillier, G. J. Lieberman. Introdução à pesquisa operacional. 8. ed. São Paulo: McGraw-Hill, 2006.

[2] Andrade, Eduardo Leopoldino. Introdução à Pesquisa Operacional - Métodos e modelos para Análise de Decisões, 5.ed. Rio de Janeiro: LTC, 2015.

[3] FÁVERO, Luiz. Pesquisa Operacional para Cursos de Engenharia 1ED. 1st. Rio de Janeiro: Campos Elsevier, 2012.

[4] Lachtermacher, G. Pesquisa Operacional na Tomada de Decisões. São Paulo: LTC, 2016.

[5] N. Kagan, H. P. Schmidt, C. C. B. Oliveira, H. Kagan. Métodos de otimização aplicados a sistemas elétricos de potência. São Paulo: Edgard Blücher, 2017.