APLICAÇÃO DO ALGORITMO *BRANCH AND BOUND* PARA SOLUÇÃO DO PROBLEMA DE PLANEJAMENTO DA EXPANSÃO DE SISTEMAS DE TRANSMISSÃO DE ENERGIA ELÉTRICA

Aldir Silva Sousa
Universidade de São Paulo
Escola de Engenharia de São Carlos
Departamento de Engenharia Elétrica
aldirss@sel.eesc.usp.br

Ricardo de Andrade Lira Rabêlo
Universidade de São Paulo
Escola de Engenharia de São Carlos
Departamento de Engenharia Elétrica
ricardor@sel.eesc.usp.br

Eduardo Nobuhiro Asada
Universidade de São Paulo
Escola de Engenharia de São Carlos
Departamento de Engenharia Elétrica
asada@sel.eesc.usp.br

RESUMO

Neste trabalho são apresentadas algumas das principais técnicas de aperfeiçoamento do algoritmo *branch and bound* e sua aplicação para resolução do problema de planejamento da expansão de sistema de transmissão de energia elétrica. Foi dado maior foco nas estratégias de seleção da variável de divisão e de seleção de novos subproblemas, uma vez que estes são os passos mais críticos do algoritmo. *branch and bound*. Neste trabalho apresentam-se também algumas sugestões que ajudaram a melhorar o desempenho do algoritmo *branch and bound* quando aplicado na resolução do problema de planejamento estudado.

PALAVRAS CHAVE. Programação Matemática, otimização combinatória, planejamento da expansão da transmissão, algoritmo *branch and bound*, transmissão de energia elétrica.

ABSTRACT

Some of the main techniques for improvement of the branch and bound algorithm and its application to the transmission expansion planning problem have been analyzed. In the algorithm, more attention on the selection of the branching variable and also on the subproblems have been given since these strategies represent critical steps of the algorithm. In this work some suggestions, which help improving the branch and bound algorithm performance, when applied to solve the transmission expansion problem, are presented.

KEYWORDS. Mathematical programming, combinatorial optimization, transmission expansion planning, branch and bound algorithm, Applications to Energy.

1 Introdução

O problema de planejamento da expansão de sistemas de transmissão de energia elétrica (PEST) tem como objetivo determinar o custo de investimento mínimo para a expansão do sistema de transmissão de energia visando o atendimento de aumento da demanda no horizonte futuro. O PEST é executado depois do estudo de previsão do aumento da geração e da demanda de energia elétrica. Após determinado em quanto e onde o sistema terá aumento de geração e demanda, surge o problema de determinar o menor custo de adição linhas de transmissão e/ou transformadores de tal forma que a demanda seja inteiramente, e de forma contínua, suprida pela geração.

O PEST pode ainda ser abordado quanto a uma perspectiva temporal. Quando não se considera o tempo, a solução do problema determina quais equipamentos (linhas de transmissão ou transformadores) devem ser adicionados ao sistema. Assim, pode-se afirmar que há somente um plano de planejamento, sendo chamado, o problema resolvido sob este panorama, de planejamento estático. Quando há uma perspectiva temporal, a solução deve determinar além do que é determinado no planejamento estático, o momento em que os equipamentos devem ser instalados na rede. Desta forma, o período de planejamento deve ser dividido em sub-períodos sendo necessário determinar quais equipamentos devem ser adicionados em cada sub-período de tal forma que o custo do investimento final seja mínimo. A abordagem em que se consideram os sub-períodos é chamada de planejamento multiestágio, Garver (1970); Romero *et al.* (2007). Os artigos Garcia *et al.* (1997); Romero *et al.* (2003); Escobar *et al.* (2004) e Asada *et al.* (2005) são exemplos de trabalhos que consideram o planejamento multiestágio.

As dificuldades de solução deste problema são devidas a sua natureza combinatória, às restrições de integralidade de algumas variáveis e à não-linearidade do problema. As metodologias sugeridas para solução do PEST podem ser divididas em dois grandes grupos, a saber, os métodos exatos e os métodos aproximados. Dentre os métodos exatos, de acordo com Romero *et al.* (2007) e Sousa e Asada (2008), o algoritmo *branch and bound* e o método de decomposição de Benders têm sido os mais usados. Porém, quando lidam com sistemas reais de grande porte, o emprego desses métodos não têm obtido desempenho esperado. Dentre os métodos inexatos, ainda segundo Romero *et al.* (2007) e Sousa e Asada (2008), os mais usados são os métodos heurísticos (especialmente os heurísticos construtivos), e as metaheurísticas. Os métodos heurísticos, quando lidam com sistemas de grande porte, encontram soluções em pouco tempo computacional, porém geralmente de baixa qualidade. Quando se consideram os sistemas de grande porte, as metaheurísticas têm apresentado os melhores resultados quando tempo computacional e qualidade de solução são consideradas.

O algoritmo *branch and bound* é um algoritmo proposto por Land e Doig (1960) para solução de problemas de programação inteira e inteira mista. Esse algoritmo é baseado na relaxação linear, aplicada para possibilitar uma enumeração implícita de toda a região factível do problema inteiro.

Este trabalho tem por objetivo avaliar algumas das estratégias de aperfeiçoamento do algoritmo *branch and bound* com o intuito de encontrar uma das soluções ótimas do PEST para sistemas de grande porte e sugerir algumas estratégias específicas para diminuir o tempo computacional de convergência do algoritmo *branch and bound* quando aplicado para resolução do PEST. Na Seção 2, será apresentado o modelo matemático do PEST utilizado; na Seção 3, será apresentado brevemente o algoritmo *branch and bound*; na Seção 4, serão mostrados os resultados dos testes realizados; na seção 5, será feita uma análise dos resultados obtidos e finalmente na seção 6 serão apresentadas as conclusões do trabalho.

2 Formulação Matemática

O problema de planejamento da expansão pode ser representado como um problema de otimização inteiro-mista de grande complexidade. Colocando-se de uma forma simples, o problema consiste na determinação de linhas de transmissão (quantidade e locação) visando atendimento de condições de operação do sistema elétrico para um horizonte futuro de operação. As condições de operação que definem a factibilidade de uma proposta de solução são representadas por restrições de igualdade e de desigualdade e remetem às equações de circuitos de Kirchhoff. Historicamente, este problema tem sido tratado, na maioria das vezes, com métodos heurísticos de otimização e também utilizando-se simplificações no modelo que representa o sistema elétrico. Dessas simplificações, três modelos (ou formulações) formam a base da maioria dos estudos presentes na literatura. O modelo cc tem como origem o problema de fluxo de potência cc em redes e representa (entre os modelos simplificados) o que mais se aproxima das condições de operação reais, Monticelli (1983). A partir do modelo cc, foram propostas outras simplificações, como por exemplo, a supressão da segunda Lei de Kirchhoff, que resulta no modelo de transportes. Neste trabalho utilizamos uma formulação de complexidade intermediária entre o cc e o transportes, chamado de híbrido linear, proposta por Villasana *et al.* (1985), que consegue gerar soluções factíveis ao modelo cc, considerado aqui como referência. Mais informações sobre os modelo de planejamento da expansão podem ser encontrados em Romero *et al.* (2002).

O modelo híbrido linear é representado da seguinte forma: (1-9).

$$\min \ v = \sum_{(i,j)\in\Omega} c_{ij} n_{ij} \tag{1}$$

sujeito a

$$Sf + S_0 f_0 + g = d \tag{2}$$

$$f_{ij}^0 - \gamma_{ij}(\theta_i - \theta_j) = 0 \qquad \forall (i,j) \in \Omega_1 \tag{3}$$

$$|f_{ij}^0| \leq n_{ij}^0 \overline{f}_{ij} \qquad \forall (i,j) \in \Omega_1 \tag{4}$$

$$|f_{ij}^{'}| \leq n_{ij} \overline{f}_{ij} \qquad \forall (i,j) \in \Omega \tag{5}$$

$$0 \leq g \leq \overline{g} \tag{6}$$

$$0 \leq n_{ij} \leq \overline{n}_{ij} \tag{7}$$

$$n_{ij} \ \mathit{inteiro} \ \forall (i,j); \ \ f_{ij} \ \mathit{irrestrito} \ \forall (i,j); \tag{8}$$

$$\theta_j \ \mathit{irrestrito} \ \forall j \tag{9}$$

onde v é o investimento total necessário para fazer o plano de expansão; c_{ij} representa o custo da linha $i-j$; n_{ij} (inteiro), cujo valor máximo é $\overline{n}_{ij}$, representa a variável linha $i-j$; S representa a matriz de incidência nó-ramo transposta do sistema elétrico inteiro; S_0 representa a matriz de incidência nó-ramo transposta das linhas existentes na configuração base; f_0 representa o vetor de fluxos de potência nas linhas existentes na configuração base, cujos elementos são f_{ij}^0; f representa o vetor de fluxos de todas as linhas adicionadas e possíveis de serem adicionadas, cujos elementos são $f_{ij}^{'}$; g representa o vetor com os elementos g_k (geração da barra k); $\overline{g}$ representa o vetor com valor máximo para cada elemento de g; d representa o vetor de demanda; γ_{ij} representa a susceptância da linha $i-j$; n_{ij}^0 representa o número de linhas existentes no ramo $i-j$ na configuração base; θ_j representa os ângulos de fase das barras que estão ligadas ao sistema elétrico na configuração base; Ω é o conjunto de todas as linhas possíveis de serem adicionadas ao sistemas e Ω_1 é o conjunto das linhas pertencentes à configuração base.

A formulação (1-9) é conhecida como modelo híbrido linear e representa um problema de programação linear-inteiro misto. A parcela de integralidade no problema é representada pelas variáveis que representa as linhas de transmissão (variável n_{ij}). O conjunto de equações (2) traduz a Lei das Correntes de Kirchhoff, e o conjunto de equações (3) traduz a Lei das Tensões de Kirchhoff para as linhas da configuração base. De acordo com a estratégia de Villasana *et al.* (1985), a formulação anterior pode ser interpretada como um problema com duas redes elétricas superpostas: uma rede em que todas as linhas de transmissão (existentes na configuração base e as possíveis de serem adicionadas) devem obedecer à lei das correntes de Kirchhoff e outra em que somente as linhas da configuração base (ou configuração inicial) devem obedecer à lei das tensões de Kirchhoff.

3 Algoritmo *Branch and Bound*

A idéia de usar o algoritmo *Branch and Bound* para resolver problemas de programação inteira usando relaxações de programação linear foi proposta por Land e Doig (1960). O processo consiste em manter uma lista de problemas lineares obtidos pela relaxação de algumas ou de todas as variáveis com restrições de integralidade, n_{ij}, neste trabalho. Para definir o algoritmo, faremos algumas convenções: N^i indica o *nó* ou *subproblema* e denota o problema associado a certa parte da região factível do modelo (1-9); v_{inf} é o limitante inferior do custo de investimento v; v_{sup} é o limitante superior do custo de investimento v; Para um nó N^i, v^i_{sup} , é a melhor solução inteira que N^i pode ter, e v^i_{inf} é o limitante inferior para a solução inteira; $\pounds$ é a lista de problemas que ainda precisam ser resolvidos e será chamada de *conjunto ativo*. O Algoritmo 1 apresenta uma versão básica do algoritmo *branch and bound*.

Algoritmo 1: Algoritmo *Branch and Bound* básico

1 **Inicialização:** $\pounds = \emptyset$ e $v_{sup} = \infty$; resolver (1-9) relaxando todas as restrições de integralidade. Vá para o passo (2).

2 **Parar:** Se a solução for factível e todas as variáveis n_{ij} forem inteiras, e $\pounds = \emptyset$, então pare. A solução ótima foi encontrada. Senão:
   a. Se a solução for factível e todas as variáveis n_{ij} forem inteiras, vá para o passo (4).
   b. Se a solução não for factível, vá para o passo (4).
   c. Se $\pounds \neq \emptyset$, vá para o passo (5).

3 **Dividir:** Seleciona-se uma variável n_{ij} e divide-se o problema em dois subproblemas: um em que se cria a restrição $n_{ij} \leq \lceil n_{ij} \rceil$ e outro com a restrição $n_{ij} \geq \lfloor n_{ij} \rfloor$. Adicionam-se ambos os subproblemas em $\pounds$.

4 **Podar:** Se N^i for factível e inteiro e $v^i_{sup} < v_{sup}$, então faça $v_{sup} = v^i_{sup}$ e remova de $\pounds$ todo N^j onde $v^j_{sup} \geq v_{sup}$. Senão, se N^i for factível e inteiro e $v^i_{sup} > v_{sup}$ ou se N^i for infactível vá para (5).

5 **Selecionar:** Seleciona-se e remove-se um novo subproblema N^i de $\pounds$. Resolve-se o PL equivalente ao nó N^i e volta-se para o passo (2).

A eficiência de uma implementação do algoritmo *branch and bound* é fortemente dependente dos passos (3) e (5) do Algoritmo 1.

3.1 Divisão do Problema

Se existem várias variáveis não inteiras após a solução da relaxação linear de N^i, deve-se selecionar uma para definir a divisão. Segundo Wolsey (2000), uma forma simples de selecionar uma variável para dividir seria selecionar a variável mais fracionária, cuja operação pode ser representada pela Eq. (10)

$$\arg \max_{ij \in C} \{\min(fr_{ij}, 1 - fr_{ij})\} \tag{10}$$

onde $fr_{ij} = n_{ij} - \lfloor n_{ij} \rfloor$ e C é o conjunto das variáveis n_{ij} não inteiras. Portanto, a variável com maior valor fracionário $fr_{ij} = \frac{1}{2}$ é a melhor para se dividir.

Outra forma é selecionar a variável não inteira de maior coeficiente na função objetivo Eq. (11). No PEST pode-se ainda selecionar a variável não inteira que tenha o maior custo e maior fluxo máximo (adicionar linhas com maior fluxo máximo poderá levar a uma configuração factível mais rapidamente, o que poderá ser usada para podar alguns subproblemas ativos levando a uma convergência mais rápida.) Eq. (12).

$$\arg\ \max_{ij \in C} \{[\min(fr_{ij}, 1 - fr_{ij})] c_{ij}\} \tag{11}$$

$$\arg\ \max_{ij \in C} \{[\min(fr_{ij}, 1 - fr_{ij})] c_{ij} \overline{f}_{ij}\} \tag{12}$$

Outra forma de selecionar a variável para dividir é usando métodos estimativos.

3.1.1 Método Estimativo

Segundo Linderoth e Savelsbergh (1999), o método estimativo funciona como segue: para cada variável inteira , são associados dois valores P_{ij}^- e P_{ij}^+ que objetivam medir o decrescimento por unidade no valor da função objetivo se n_{ij} for fixada no valor de seu limite inteiro inferior e no seu limite inteiro superior, respectivamente. Chama-se P_{ij}^- de pseudocusto inferior e P_{ij}^+ de pseudocusto superior. Uma forma de obter P_{ij}^- e P_{ij}^+ é simplesmente usar a degradação observada do valor da função objetivo. Sejam N^{i-} (nó com fixação do limite inferior) e N^{i+} (nó com fixação do limite superior) os nós decorrentes da divisão do nó N^i, os pseudocustos podem ser calculados como segue:

$$P_{ij}^- = \frac{v_{LP}^{i-} - v_{LP}^i}{fr_{ij}^i}; \quad P_{ij}^+ = \frac{v_{LP}^{i+} - v_{LP}^i}{1 - fr_{ij}^i}$$

A seleção da variável para divisão é dada pela Eq. (13).

$$\arg \max_{ij} \{P_{ij}^- fr_{ij} + P_{ij}^+ (1 - fr_{ij})\} \tag{13}$$

Quando da implementação, duas perguntas devem ser respondidas: 1) como os pseudocustos devem ser inicializados? 2) Como devem ser atualizados?

A questão sobre inicialização é importante, uma vez que, de acordo com Forrest *et al.* (1974), se no nó raiz uma variável que tem pouco ou nenhum efeito na solução do PL dos nós subseqüentes se divide, então o esforço computacional necessário para convergência do algoritmo pode até dobrar. Linderoth e Savelsbergh (1999) sugerem iniciar os pseudocustos das variáveis com o valor de seu coeficiente na função objetivo. Em testes realizados, verificamos que, como os custos das linhas são muito diferentes entre si, inicializar os pseudocustos com estes coeficientes pode aumentar o esforço computacional para convergência, uma vez que as linhas mais caras terão pseudocustos muito elevados. Assim, em nosso trabalho, foram realizados testes iniciando os pseudocustos com o valor do coeficiente das variáveis na função objetivo ($P_{ij}^- = P_{ij}^+ = c_{ij}$), e com o valor do coeficiente das variáveis na função objetivo dividido pelo maior coeficiente ($P_{ij}^- = P_{ij}^+ = c_{ij}/\{arg \ \max_{ij}\{c\}\}$). Desta forma, os coeficientes das variáveis na função objetivo somente terão grande impacto quando das primeiras iterações, uma vez que neste momento os pseudocustos ainda não terão sido calculados.

No decorrer do processo de solução, a variável n_{ij} pode ser dividida diversas vezes. Daí surge a questão sobre como se deve atualizar o pseudocusto desta variável. Bénichou *et al.* (1971) afirmam que os pseudocustos variam muito pouco em toda a árvore *branch and bound* e sugerem fixar P_{ij}^- e P_{ij}^+ nos valores observados quando da primeira vez que a variável n_{ij} foi dividida. Alternativamente, Forrest *et al.* (1974) sugerem fixar os pseudocustos nos valores obtidos na última vez que n_{ij} foi dividida ou na média de todas as vezes que foi n_{ij} dividida. Foram realizados testes com todas estas sugestões, como será observado na seção 4.

3.2 Seleção de Subproblemas

Linderoth e Savelsbergh (1999) afirmam que ao se selecionar um novo nó, há dois propósitos: encontrar boas soluções factíveis inteiras ou provar que não existem soluções inteiras melhores do que a solução atual v_{Inf}. Existem algumas estratégias para selecionar novos nós. As estratégias podem ser divididas em estratégias estáticas e estratégias dinâmicas. Dentre as estratégias estáticas, as mais utilizadas são:

- *Busca em profundidade*: Wolsey (2000) afirma que somente é possível podar a árvore significativamente com uma solução factível. Portanto, deve-se descer o mais rápido possível na árvore de enumeração para encontrar uma primeira solução factível. Como será observado nos testes, esta estratégia também tem a vantagem de gerar poucos subproblemas ao longo de toda a execução do algoritmo. Esta estratégia pode ser computacionalmente implementada usando a estrutura de dados *pilha*. Uma pilha é um contêiner em que os elementos são postos e removidos sempre do topo.
- *Busca em largura*: a busca em largura força uma expansão da árvore para os lados. A busca em largura pode ser implementada através da estrutura de dados *fila*. Uma fila é um contêiner em que os elementos são postos no final e removidos do início. Para melhorar o tempo computacional, quando da implementação, sugere-se o uso de tipos apontadores (um para o início da fila e outra para o fim) para diminuir o tempo necessário para adicionar e remover os subproblemas da fila.
- *Busca do melhor limite ou melhor primeiro*: com esta estratégia, pretende-se encontrar rapidamente uma solução factível para melhorar a eficiência da poda. Esta estratégia pode ser implementada computacionalmente através da estrutura de dados *heap*. Um *heap* é um contêiner em que os elementos se mantêm ordenados de tal forma que o menor elemento (de acordo com uma chave pré-definida, no nosso caso o valor de v) fica no inicio do *heap*. Assim como a estratégia busca em largura, a estratégia busca do melhor limite gera muitos subproblemas, exigindo-se memória em quantidade que pode se tornar proibitiva quando se lida com sistemas de grande porte.

Em Cormen *et al.* (2001) encontram-se mais detalhes sobre *heap*, fila e pilha. Foram realizados testes com as três estratégias estáticas definidas acima.

Uma estratégia dinâmica de seleção de novos subproblemas utiliza os pseudocustos calculados quando da divisão das variáveis. Neste trabalho, foi implementada a estratégia de melhor estimativa, sugerida por Bénichou *et al.* (1971). A melhor estimativa pode ser calculada de acordo com a Eq. (14).

$$E_i = v_{Inf}^i + \sum_{ij \in C} \min\{P_{ij}^- fr_{ij}, P_{ij}^+(1 - fr_{ij})\} \tag{14}$$

O subproblema de melhor estimativa é aquele apresentar o menor valor de E_i. Este problema deverá ser selecionado para verificação.

4 Testes

Para verificar a eficiência das estratégias sugeridas acima, foram realizados testes em sistemas reais de transmissão de energia elétrica. Os sistemas testados foram:

1. O sistema sul brasileiro (sul1), com redespacho da geração. Neste sistema, há mais geração do que demanda total. Abaixo, alguns dados importantes sobre o sistema.
   - Número de barras: 46.
   - Número de linhas (número de variáveis inteiras): 79.
   - Geração total: 10545 MW.
   - Demanda total: 6880 MW.
2. O sistema sul brasileiro (sul2), sem redespacho da geração. Neste sistema, a capacidade de geração é igual à demanda total, o que não possibilita folga nos geradores. Abaixo, alguns dados importantes sobre o sistema.
   - Número de barras: 46.
   - Número de linhas (número de variáveis inteiras): 79.
   - Geração total: 6880 MW.
   - Demanda total: 6880 MW.
3. O sistema colombiano (colo), plano 2012. Sem redespacho da geração. Abaixo, alguns dados importantes sobre o sistema.
   - Número de barras: 93.
   - Número de linhas (número de variáveis inteiras): 155.
   - Geração total: 14559 MW.
   - Demanda total: 14559 MW.

Todas as implementações computacionais foram desenvolvidas em Fortran 90 e compiladas com o compilador *Intel(R) Fortran Compiler 10.1.024 [IA-32].* Foi utilizado o pacote computacional MINOS 5.4 (desenvolvido em Fortran 77) para resolver os PLs necessários. Todos os testes foram realizados em um computador com processador *Intel*$^{(R)}$ *Core*^{TM} *2 CPU 6400 @ 2.13 GHz 2.13GHz, 3,00 GB* de RAM; sistema operacional Windows XP.

Sempre após a seleção da variável de divisão, verifica-se primeiro o subproblema com a restrição $n_{ij} \leq \lceil n_{ij} \rceil$ se $fr_{ij} \leq 1 - fr_{ij}$ para a variável selecionada. Se, para a variável selecionada para divisão, $fr_{ij} > 1 - fr_{ij}$, então se verifica primeiro o subproblema com a restrição $n_{ij} \geq \lfloor n_{ij} \rfloor$.

Os testes foram realizados da seguinte forma:

1. Testes realizados considerando os métodos estáticos de seleção de novos nós:
   (a) Busca em profundidade (FIFO);
   (b) Busca em largura (LIFO);
   (c) Melhor primeiro (MP).

   Em todos os casos, foram realizados testes usando os seguintes critérios de seleção de variável para dividir:

   i. Maior parte fracionária e maior custo (MFC), de acordo com a Eq. (11);

   ii. Maior parte fracionária, maior custo e maior fluxo máximo (MFCF), de acordo com a Eq. (12).

2. Testes foram realizados considerando os pseudocustos calculados para seleção de novos subproblemas e para definir a variável de dividir. Quando considerando os pseudocustos, foram realizados testes em que:
   (a) Os pseudocustos foram inicializados pelo os coeficientes da função objetivo (ICFO).
   (b) Os pseudocustos foram inicializados com o valor do coeficiente das variáveis na função objetivo dividido pelo maior coeficiente (ICFOMD).

   Em ambos os casos, foram realizados testes em que a atualização dos pseudocustos foi realizada de acordo com os critérios abaixo:
   i. Os pseudocustos foram atualizados pela média (PAM);
   ii. Os primeiros pseudocustos calculados eram considerados (PPC);
   iii. Os últimos pseudocustos calculados eram considerados (UPC).

Todos os métodos que convergiram identificaram as seguintes soluções ótimas:

- Sul1: Investimento de $v = US\$63.163.000$. Com a seguinte configuração ótima:
$$n_{20-23} = 1; n_{20-21} = 2; n_{42-43} = 1; n_{46-06} = 1; n_{05-06} = 2.$$
- Sul2: Investimento de $v = US\$141.350.000$. Com a seguinte configuração ótima:
$$n_{20-21} = 1; n_{42-43} = 2; n_{46-06} = 1; n_{25-32} = 1; n_{31-32} = 1;$$
$$n_{28-31} = 1; n_{28-30} = 1; n_{26-29} = 2; n_{24-25} = 2; n_{29-30} = 1;$$
$$n_{05-06} = 1.$$
- Colo: Investimento de $v = US\$470.361.000$. Com a seguinte configuração ótima:
$$n_{52-88} = 1; n_{43-88} = 2; n_{57-81} = 1; n_{14-31} = 1; n_{15-18} = 1; n_{55-84} = 1;$$
$$n_{55-62} = 1; n_{69-70} = 1; n_{09-69} = 1; n_{60-69} = 3; n_{31-72} = 3; n_{19-22} = 1;$$
$$n_{05-06} = 1; n_{19-58} = 1; n_{27-64} = 1; n_{19-66} = 3; n_{34-70} = 1; n_{50-54} = 1;$$
$$n_{68-86} = 1.$$

Nas tabelas mostradas, a coluna *No de PLs PEO* mostra o número de PLs executados até a solução ótima ser encontrada; *No de PL AC* mostra o número de PLs executados até a convergência do algoritmo; *Tempo (s)* mostra o tempo computacional em segundos até a convergência e *MNNA* é o maior número de subproblemas armazenados durante toda a execução. *NC* significa que o algoritmo não convergiu após executar mais de 15.000.000 PLs, e < 1 significa que o algoritmo convergiu em menos de um segundo.

4.1 Testes Realizados Considerando os Métodos Estáticos de Seleção de Novos Nós

A Tabela 1 apresenta os resultados quando da aplicação de FIFO com MFC. A Tabela 2 apresenta os resultados quando da aplicação de FIFO com MFCF.

Table 1: Aplicação de FIFO com MFC

Sistema	No de PLs PEO	No de PL AC	Tempo (s)	MNNA
Sul1	166	229	< 1	7
Sul2	516	3.949	4	12
colo	1.205.431	1.359.099	3.187	96

Table 2: Aplicação de FIFO com MFCF

Sistema	No de PLs PEO	No de PL AC	Tempo (s)	MNNA
Sul1	206	267	< 1	7
Sul2	182	1.165	2	11
colo	4.601.648	4.569.120	11.183	19

A Tabela 3 apresenta os resultados quando da aplicação de LIFO com MFC. A Tabela 4 apresenta os resultados quando da aplicação de LIFO com MFCF.

Table 3: Aplicação de LIFO com MFC

Sistema	No de PLs PEO	No de PL AC	Tempo (s)	MNNA
Sul1	89	185	< 1	18
Sul2	843	4.027	5	217
colo	NC	NC	NC	738.157

Table 4: Aplicação de LIFO com MFCF

Sistema	No de PLs PEO	No de PL AC	Tempo (s)	MNNA
Sul1	96	138	< 1	22
Sul2	429	1.190	2	88
colo	429.867	430.530	1.166	21.399

A Tabela 5 apresenta os resultados quando da aplicação de MP com MFC. A Tabela 6 apresenta os resultados quando da aplicação de MP com MFCF.

Table 5: Aplicação de MP com MFC

Sistema	No de PLs PEO	No de PL AC	Tempo (s)	MNNA
Sul1	86	194	< 1	24
Sul2	393	7.199	9	409
colo	NC	NC	NC	52.555

Table 6: Aplicação de MP com MFCF

Sistema	No de PLs PEO	No de PL AC	Tempo (s)	MNNA
Sul1	81	123	< 1	32
Sul2	490	1.175	1	101
colo	13.700.803	13.700.880	47.137	6.577

4.2 Testes Realizados Considerando os Pseudocustos

A Tabela 7 mostra os resultados quando da aplicação de ICFO com PAM, a Tabela 8 apresenta os resultados dos testes para ICFO com PPC e a Tabela 9, os resultados dos testes com ICFO com UPC.

Table 7: Aplicação de ICFO com PAM

Sistema	No de PLs PEO	No de PL AC	Tempo (s)	MNNA
Sul1	159	231	1	35
Sul2	1.666	3.154	4	154
colo	1.790	85.760	241	1.790

Table 8: Aplicação de ICFO com PPC

Sistema	No de PLs PEO	No de PL AC	Tempo (s)	MNNA
Sul1	162	241	1	35
Sul2	6.492	10.689	11	322
colo	NC	NC	NC	3.353

Table 9: Aplicação de ICFO com UPC

Sistema	No de PLs PEO	No de PL AC	Tempo (s)	MNNA
Sul1	159	232	< 1	35
Sul2	2.196	3.751	4	182
colo	62.777	113.752	325	3.711

A Tabela 10 mostra os resultados quando da aplicação de ICFOMD com PAM, a Tabela 11 apresenta os resultados dos testes para ICFOMD com PPC e a Tabela 12, os resultados dos testes com ICFOMD com UPC.

Table 10: Aplicação de ICFOMD com PAM

Sistema	No de PLs PEO	No de PL AC	Tempo (s)	MNNA
Sul1	143	234	< 1	35
Sul2	4.184	4.606	5	172
colo	11.676	66.011	181	1.202

Table 11: Aplicação de ICFOMD com PPC

Sistema	No de PLs PEO	No de PL AC	Tempo (s)	MNNA
Sul1	156	238	0	35
Sul2	5.714	6.440	7	252
colo	NC	NC	NC	29.659

Table 12: Aplicação de ICFOMD com UPC

Sistema	No de PLs PEO	No de PL AC	Tempo (s)	MNNA
Sul1	143	245	< 1	35
Sul2	1.604	4.188	4	148
colo	33.090	99.313	287	3.272

5 Análise dos Resultados Obtidos

Pelas tabelas mostradas, verificou-se que, considerando a seleção estática de novos subproblemas, a escolha da variável para divisão pela Eq. (11) se mostrou melhor do que pela Eq. (12) somente quando aplicada a FIFO com MFC. Mesmo neste caso, em uma aplicação em que o uso de memória é crítico, a seleção pela Eq. (12) pode ser mais vantajosa uma vez que por ela o algoritmo gera menos subproblemas. A seleção pela Eq. (12) para divisão se mostrou superior à pela Eq. (11) em todos os outros testes realizados, como se pode confirmar pelas tabelas (3,4) e (5,6). A seleção da variável para divisão considerando a Eq. (10) apresentou resultados muito ruins, o que motivou os autores a não mostrá-los neste trabalho.

Com testes realizados usando pseudocustos, mostrou-se que esta estratégia apresenta soluções melhores do que pela seleção estática. Dentre as estratégias de atualização, a que apresentou os piores resultados foi a PPC, para todos os testes realizados, como se pode observar através das tabelas 8 e 11. A estratégia de atualização que mostrou melhor desempenho foi a atualização pela média dos pseudocustos (PAM).

Como se pode ser observado pelas tabelas 7-12, iniciar os pseudocustos com o valor dos coeficientes das variáveis da função objetivo dividido pelo maior coeficiente pode proporcionar uma ligeira melhoria no desempenho do algoritmo (em relação à inicialização dos pseudocustos somente pelo valor dos coeficientes da função objetivo), além de, em geral, gerar menos subproblemas. Dentre todas as estratégias testadas, a que apresentou melhor desempenho quando testada com o sistema colombiano, que é um sistema de grande porte, foi a estratégia ICFOMD com PAM. A estratégia ICFOMD com PAM, quando testada com o sistema de grande porte, gerou menos subproblemas e executou o menor número de PLs até a convergência. Conseqüentemente, foi a que convergiu no menor espaço de tempo.

6 Conclusões

Este trabalho teve por objetivo realizar experimentos para definir as melhores estratégias do algoritmo *branch and bound* para solução do problema de planejamento da expansão de sistemas de transmissão de energia elétrica. Além disso, propor algumas estratégias simples que melhoram ligeiramente a convergência do algoritmo. Vários testes foram realizados considerando os sistemas reais colombiano e sul brasileiro. Com os resultados obtidos podem-se ser realizadas análises para determinar os melhores estratégias para aplicar ao planejamento da expansão da transmissão de energia elétrica. Em geral, verificou-se que a estratégia ICFOMD com PAM apresentou os melhores resultados quando aplicada ao sistema de grande porte testado.

Agradecimentos

Os autores gostariam de agradecer à FAPESP pelo apoio de equipamentos e financeiro fornecidos, os quais possibilitaram a realização deste trabalho.

Referências

Asada, E. N., Carreno, E., Romero, R. e Garcia, A. V. (2005). A branch-and-bound algorithm for the multi-stage transmission expansion planning. In *Proc. IEEE Power Engineering Society General Meeting*, pages 171–176.

Bénichou, M., Gauthier, J. M., Girodet, P.and Hentges, G., Ribière, G. e Vincent, O. (1971). Experiments in mixed-integer linear programming. *Mathematical Programming*, **1**, 76–94.

Cormen, T. H., Leiserson, C. E. e Rivest, R. L. (2001). *Introduction to algorithms*. MIT Press.

Escobar, A. H., Gallego, R. A. e Romero, R. (2004). Multistage and coordinated planning of the expansion of transmission systems. *IEEE Transactions on Power Systems*, **19**(2), 735–744.

Forrest, J. J. H., Hirst, J. P. H. e Tomlin, J. A. (1974). Practical solution of large scale mixed integer programming problems with umpire. *Management Science*, **20**, 773–776.

Garcia, B.-L., Mahey, P. e LeBlanc, L. J. (1997). Iterative improvement methods for a multiperiod network design problem. *European Journal of Operational Research*.

Garver, L. L. (1970). Transmission network estimation using linear programming. *IEEE Transactions on Power Apparatus and Systems*, (7), 1688–1697.

Land, A. e Doig, A. (1960). An automatic method of solving discrete programming problems. *Econometrica: Journal of the Econometric Society*, pages 497–520.

Linderoth, J. e Savelsbergh, M. (1999). A computational study of search strategies for mixed integer programming. *INFORMS Journal on Computing*, **11**, 173–187.

Monticelli, A. (1983). *Fluxo de carga em redes de energia elétrica*. Editora Edgard Blücher Ltda.

Romero, R., Monticelli, A., Garcia, A. e Haffner, S. (2002). Test systems and mathematical models for transmission network expansion planning. *IEE Proceedings-Generation, Transmission and Distribution*, **149**(1), 27–36.

Romero, R., Rocha, C., Mantovani, M. e Mantovani, J. (2003). Analysis of heuristic algorithms for the transportation model in static and multistage planning in network expansion systems. *IEE Proceedings. Generation, Transmission and Distribution*, **150**(5), 521–526.

Romero, R., Asada, E. N., Carreno, E. e Rocha, C. (2007). Constructive heuristic algorithm in branch-and-bound structure applied to transmission network expansion planning. *IET Generation, Transmission & Distribution*, **1**(2), 318–323.

Sousa, A. S. e Asada, E. N. (2008). Tomada de decisão fuzzy aplicada ao planejamento da expansão de sistemas de transmissão. In *XVII Congresso Brasileiro de Automática, Juiz de Fora - MG*.

Villasana, R., Garver, L. L. e Salon, S. J. (1985). Transmission network planning using linear programming. *IEEE Transactions on Power Apparatus and Systems*, (2), 349–356.

Wolsey, L. A. (2000). *Integer Programming*. John Wiley & Sons, Inc.