SEMANA ACADÊMICA
REVISTA CIENTÍFICA
ISSN 2236-6717

ALGORITMO DE BUSCA DISPERSA PARA RESOLUÇÃO DE PROBLEMAS DE OTIMIZAÇÃO DISCRETOS E CONTÍNUOS

André Mendes Garcia[1]
https://orcid.org/0000-0001-7395-2190

RESUMO

Os problemas de otimização são muito complexos de serem resolvidos, pois podem apresentar múltiplas soluções ótimas, e em determinados casos apenas uma solução ótima global. A formulação matemática desses problemas é composta de variáveis contínuas e/ou discretas, e possuem funções lineares e/ou não-lineares. A solução destes problemas por modelagem matemática, através de *solvers*, podem não convergir para uma solução factível e/ou demandar muito tempo computacional, considerando um número elevado de variáveis. Desta forma, a utilização de meta-heurísticas para solução desses problemas é uma opção bastante atraente e utilizada na literatura. O algoritmo da meta-heurística busca dispersa (BD) vem se destacando na literatura como um bom algoritmo de otimização combinatória por tratar da qualidade e diversidade das soluções. O objetivo deste trabalho é apresentar duas metodologias utilizando o algoritmo de BD para solução de problemas com variáveis contínuas e discretas. Para tanto, a BD é implementada para solucionar o problema do caixeiro viajante (PCV), o qual é um problema que contém 100% de variáveis discretas, e também, uma outra versão da BD é implementada para solucionar o problema de fluxo de potência ótimo (FPO), o qual possui variáveis contínuas e discretas. Foram utilizadas duas instâncias do PCV e três instâncias do problema de FPO. Os resultados mostram que a BD resolve os dois tipos de problemas, e, em alguns casos apresentou resultados melhores se comparado às soluções dos mesmos problemas utilizando os modelos matemáticos clássicos através de *solvers*. Conclui-se que, a implementação do algoritmo da BD não requer muito esforço, e que se pode alcançar bons resultados para solução de problemas de otimização com variáveis de qualquer natureza.

Palavras-chave

Algoritmo; Busca dispersa; Meta-heurística; Otimização.

[1] Professor do Centro Universitário de Adamantina-SP. Doutor em Engenharia Elétrica pela UNESP de Ilha Solteira. São Paulo, andre@fai.com.br

http://dx.doi.org/10.35265/2236-6717-238-12778
FORTALEZA-CE. EDIÇÃO 238. V.11. ANO 2023.

SEMANA ACADÊMICA
REVISTA CIENTÍFICA
ISSN 2236-6717

Scattered Search Algorithm for Solving Discrete and Continuous Optimization Problems

ABSTRACT

Optimization problems are very complex to be solved, as they can present multiple optimal solutions and, in certain cases, only a globally optimal solution. The mathematical formulation of these problems is composed of continuous and discrete variables with linear and nonlinear functions. The solution of these problems by mathematical modeling through solvers may not converge to a feasible solution or may demand much computational time, when many variables are considered. Thus, using metaheuristics to solve these problems is desirable according to the literature. The scatter search (SS) metaheuristic algorithm has been highlighted in the literature as an excellent combinatorial optimization algorithm for dealing with quality and diversity solutions. This work aims to present two methods using the SS algorithm to solve problems with continuous and discrete variables. Therefore, the SS algorithm is implemented to solve the traveling salesperson problem (TSP), which is a problem that contains only discrete variables, and also, another version of SS is implemented to solve the optimal power flow (OPF) problem, which has continuous and discrete variables. Two instances of the TSP and three instances of the OPF problem were used. The results show that the SS solves both problems efficiently and presents better results in some cases than the solutions of the same problems using classical mathematical models through solvers. It is concluded that implementing the SS algorithm does not require much effort and that good results can be achieved for solving optimization problems with variables of any nature.

Keywords

Algorithm; Scattered search; Metaheuristics; Optimization.

Submetido em: 10/08/2023 – **Aprovado em**: 10/09/2023 – **Publicado em**: 19/09/2023

1 INTRODUÇÃO

Solucionar problemas de otimização combinatória é uma das áreas da pesquisa operacional (PO) que mais demanda pesquisas, pois os problemas são complexos, possuem variáveis discretas e/ou contínuas, e a formulação matemática pode conter funções lineares e/ou não lineares. A solução desses problemas pode ser feita por modelos matemáticos, através de *solvers*, ou pela utilização de algoritmos de meta-heurísticas. Dada a complexidade do problema, a solução por modelos matemáticos utilizando *solvers*, pode não apresentar uma solução factível ou exigir um tempo computacional muito elevado. A solução desses problemas por meta-heurísticas, também não garantem soluções factíveis, e em alguns casos também podem exigir grandes tempos computacionais. Entretanto, a utilização de meta-heurísticas nesses problemas vem sendo cada vez mais tratada na literatura e bons resultados são obtidos. Há uma grande variedade de meta-heurísticas desenvolvidas, algumas baseiam-se em fenômenos da natureza, como o algoritmo genético, e outras não. Algumas trabalham com apenas uma solução e outras com um conjunto de soluções, onde essas soluções evoluem com as iterações dos algoritmos.

Neste trabalho a meta-heurística BD será utilizada em dois problemas diferentes, um com variáveis discretas e outro com variáveis contínuas e discretas, sendo o problema clássico da ciência da computação, o caixeiro viajante e o problema da área de engenharia elétrica fluxo de potência ótimo, respectivamente.

O objetivo deste trabalho, é contribuir com a literatura apresentando os passos da BD para se trabalhar com variáveis de diferentes naturezas.

1.1 Problema do Caixeiro Viajante (PCV)

O problema do caixeiro viajante (PCV), também conhecido como TSP (*The travelling salesman problem*), é um dos mais tradicionais problemas do ramo da Pesquisa Operacional. Consiste basicamente em um conjunto de cidades, e todas estas cidades devem ser visitadas por um agente viajante. Partindo-se de uma cidade de origem, este agente viajante deve visitar todas as outras cidades uma única vez e voltar à cidade de origem, considerando que o *tour* percorrido por este agente deve ter a menor distância possível.

Este problema é considerado muito importante porque está relacionado com modelos matemáticos de outros problemas importantes, e, geralmente é muito difícil de encontrar a solução ótima quando a instância do problema é muito grande, ou seja, quando o número de cidades é muito grande.

Existem na literatura diversas formulações matemáticas para resolver o problema, e podem ser categorizadas como problemas de Programação Linear Inteira (PLI) ou

Programação Linear Inteira Mista (PLIM). Nos trabalhos de Gavish e (1978) e Orman e Williams (2007) são apresentadas diversas formulações para o problema.

Há também na literatura muitos trabalhos que tratam o problema como um problema de otimização combinatorial sem considerar a formulação matemática, utilizando heurísticas e meta-heurísticas (BENEVIDES, 2011).

1.2 Problema de Fluxo de Potência Ótimo (FPO)

O problema de FPO, é um problema clássico da engenharia elétrica, e, que consiste basicamente em otimizar uma ou mais funções objetivo que definirão a melhor estratégia de operação de um sistema de energia elétrica, assegurando o atendimento das restrições operacionais e físicas de todo o sistema (CARPENTIER, 1962), (CARPENTIER, 1979).

Diferentes problemas o FPO pode ser modelado para resolver, dentre eles podem ser citados: Minimização dos custos de geração energia; das perdas nas linhas de transmissão; da emissão de poluição na geração; do número de ações de controle; da injeção de potência reativa; do corte de carga; dentre outros.

Dependendo do objetivo a ser alcançado, a modelagem matemática dos problemas de FPO na maioria das vezes apresenta um problema de programação não linear inteira mista (PNLIM), o que torna o problema de difícil solução. Na literatura, há basicamente três categorias de metodologias para resolver esse problema de otimização: metodologias clássicas de otimização, que resolvem o problema utilizando métodos matemáticas clássicos (SUN et al., 1984), (GRANVILLE, 1994), (HUA WEI et al., 1998), (TORRES; QUINTANA, 1998), (POTLURI; HEDMAN, 2012), (KHANABADI; GHASEMI; DOOSTIZADEH, 2013), (LOW, 2014); metodologias estocásticas, também chamadas de não determinísticas, que se utilizam de métodos heurísticos e meta-heurísticas para resolver o problema; e as metodologias híbridas, que integram os métodos clássicos de otimização com as meta-heurísticas (IBA, 1994), (YURYEVICH; KIT PO WONG, 1999), (BAKIRTZIS et al., 2002), (LEE; VLACHOGIANNIS, 2005), (CAPITANESCU; WEHENKEL, 2014), (SURENDER REDDY; BIJWE, 2016), (MOHAMED et al., 2017), (JIANG et al., 2017), (ATTIA; EL SEHIEMY; HASANIEN, 2018), (SALKUTI, 2018).

Este problema é caracterizado como um problema de programação não linear inteira-mista (PNLIM), uma vez que na função objetivo, as restrições do balanço de potência ativa e reativa do sistema, de acordo com a lei das correntes de *Kirchhoff*, e as expressões dos fluxos de potência ativa e reativa são não lineares. Além disso, as variáveis de controle dos *taps* dos transformadores são inteiras e as variáveis de chaveamento do *shunt* são binárias.

As expressões (1)–(17) representam a modelagem matemática do problema.

$$\min v = \sum_{i \in \Omega_g} \left[c_i^q \left(P_i^g\right)^2 + c_i^l P_i^g + c_i^c \right] (\text{US\$/h}) \tag{1}$$

sujeito a:

$$P_i^g - P_i^d - \sum_{kji \in \Omega_l} P_{kji}^{para} - \sum_{kij \in \Omega_l} P_{kij}^{de} - V_i^2 G_i^{sh} = 0 \qquad \forall i \in \Omega_b \qquad (2)$$

$$Q_i^g - Q_i^d - \sum_{kji \in \Omega_l} Q_{kji}^{para} - \sum_{kij \in \Omega_l} Q_{kij}^{de} + V_i^2 B_i^{sh} \omega_i = 0 \qquad \forall i \in \Omega_b \qquad (3)$$

$$P_{kij}^{de} = sw_{kij} \left[G_{kij} \left(1 + \frac{r_{kij}^{\%} t_{kij}}{\overline{nt}_{kij}} \right)^2 V_i^2 - \left(1 + \frac{r_{kij}^{\%} t_{kij}}{\overline{nt}_{kij}} \right) V_i V_j \left(G_{kij} \cos\left(\theta_i - \theta_j + \phi_{kij}\right) + B_{kij} \operatorname{sen}\left(\theta_i - \theta_j + \phi_{kij}\right) \right) \right] \qquad (4)$$

$$P_{kij}^{para} = sw_{kij} \left[G_{kij} V_j^2 - \left(1 + \frac{r_{kij}^{\%} t_{kij}}{\overline{nt}_{kij}} \right) V_i V_j \left(G_{kij} \cos\left(\theta_i - \theta_j + \phi_{kij}\right) - B_{kij} \operatorname{sen}\left(\theta_i - \theta_j + \phi_{kij}\right) \right) \right] \qquad (5)$$

$$Q_{kij}^{de} = sw_{kij} \left[-\left(B_{kij} + B_{kij}^{shl}\right) \left(1 + \frac{r_{kij}^{\%} t_{kij}}{\overline{nt}_{kij}} \right)^2 V_i^2 + \left(1 + \frac{r_{kij}^{\%} t_{kij}}{\overline{nt}_{kij}} \right) V_i V_j \left(B_{kij} \cos\left(\theta_i - \theta_j + \phi_{kij}\right) - G_{kij} \operatorname{sen}\left(\theta_i - \theta_j + \phi_{kij}\right) \right) \right] \qquad (6)$$

$$Q_{kij}^{para} = sw_{kij} \left[-\left(B_{kij} + B_{kij}^{shl}\right) V_j^2 + \left(1 + \frac{r_{kij}^{\%} t_{kij}}{\overline{nt}_{kij}} \right) V_i V_j \left(B_{kij} \cos\left(\theta_i - \theta_j + \phi_{kij}\right) + G_{kij} \operatorname{sen}\left(\theta_i - \theta_j + \phi_{kij}\right) \right) \right] \qquad (7)$$

$$\forall kij \in \Omega_l$$

$$\underline{V} \leq V_i \leq \overline{V} \qquad \forall i \in \Omega_b \qquad (8)$$

$$\underline{P}_i^g \leq P_i^g \leq \overline{P}_i^g \qquad \forall i \in \Omega_b \qquad (9)$$

$$\underline{Q}_i^g \leq Q_i^g \leq \overline{Q}_i^g \qquad \forall i \in \Omega_b \qquad (10)$$

$$\left(P_{kij}^{de}\right)^2 + \left(Q_{kij}^{de}\right)^2 \leq \overline{S}_{kij}^2 \qquad \forall kij \in \Omega_l \qquad (11)$$

$$\left(P_{kij}^{para}\right)^2 + \left(Q_{kij}^{para}\right)^2 \leq \overline{S}_{kij}^2 \qquad \forall kij \in \Omega_l \qquad (12)$$

$$-\overline{nt}_{kij} \leq t_{kij} \leq \overline{nt}_{kij} \qquad \forall kij \in \Omega_l \qquad (13)$$

$$\theta_{i^*} = 0 \quad (14)$$

$$\omega_i \in \{0,1\} \quad \forall i \in \Omega_b \quad (15)$$

$$sw_{kij} \in \{0,1\} \quad \forall kij \in \Omega_l \quad (16)$$

$$t_{kij} \in \mathbb{Z} \quad \forall kij \in \Omega_l \quad (17)$$

Nesse modelo, os conjuntos são:

- Ω_b : Conjunto de barras do sistema, indexado pelo índice i, sendo que i^* é a barra de referência;
- Ω_g : Conjunto de barras geradoras, indexado pelo índice i;
- Ω_l : Conjunto de linhas, indexado pelo índice kij que representa a linha ou transformador k no ramo ij, sendo que k é utilizado para representar componentes em paralelo em um mesmo ramo ij.

Os parâmetros são:

- c_i^q, c_i^l, e c_i^c : Coeficientes de custo de geração quadrático, linear e constante, respectivamente, na barra i;
- P_i^d : Potência ativa demandada na barra i;
- Q_i^d : Potência reativa demandada na barra i;
- G_i^{sh} e B_i^{sh} : Condutância e susceptância *shunt* conectadas à barra i;
- G_{kij} e B_{kij} : Condutância e susceptância série da linha k no ramo ij;
- B_{kij}^{shl} : Susceptância *shunt* da linha kij;
- $r_{kij}^{\%}$: Passo de regulação para o *tap* do transformador k no ramo ij;
- $\overline{nt}_{kij}$: Número de *taps* do transformador k no ramo ij;
- $\overline{S}_{kij}$: Limite de fluxo de potência aparente da linha k no ramo ij;
- $\underline{V}$: Limite mínimo magnitude de tensão;
- $\overline{V}$: Limite máximo magnitude de tensão;
- $\underline{P}_i^g$: Limite mínimo de geração de potência ativa na barra i;
- $\overline{P}_i^g$: Limite máximo de geração de potência ativa na barra i;
- $\underline{Q}_i^g$: Limite mínimo de geração de potência reativa na barra i;
- $\overline{Q}_i^g$: Limite máximo de geração de potência reativa na barra i.

As variáveis contínuas são:

- P_i^g : Potência ativa gerada na barra i;
- Q_i^g : Potência reativa gerada na barra i;
- V_i : Magnitude de tensão na barra i;
- θ_i : Ângulo de tensão na barra i;
- ϕ_{kij} : Ângulo de defasagem do transformador defasador k no ramo ij.

E as variáveis discretas são:

- ω_i : Variável binária de chaveamento do *shunt* conectado à barra i;
- t_{kij} : Posição do *tap* do transformador k no ramo ij;

- sw_{kij} : Variável binária que determina se a linha k no ramo ij está ligada ou não. Neste caso o valor de todas essas variáveis estará configurado para 1, ou seja, considera-se neste trabalho que todas as linhas de transmissão estejam operantes.

No modelo matemático apresentado, a equação (1) representa a função de custo de geração, as equações (2) e (3) representam os balanços de potência ativa e reativa do sistema, de acordo com a lei das correntes de *Kirchhoff*. Já as equações (4)–(7) representam os cálculos dos fluxos de potência ativa e reativa nas linhas, em conformidade à lei das tensões de *Kirchhoff* para cada laço fundamental. As expressões (8), (9) e (10) representam os limites de tensão e geração de potência ativa e reativa, respectivamente. Os limites de capacidade das linhas de transmissão são impostos pelas expressões (11) e (12). Por fim, a expressão (13) denota os limites das posições dos *taps* dos transformadores e a expressão (14) impõe que o ângulo de tensão na barra de referência (*slack*) deve ser zero. As restrições (15)–(17) indicam a natureza discreta das variáveis de operação relacionadas com os compensadores *shunt*, estado de operação das linhas e posição dos *taps* nos transformadores (GARCIA, 2019).

1.3 Meta-heurística Busca Dispersa (BD)

O algoritmo de busca dispersa (BD) é uma meta-heurística da característica evolucionária, que trabalha com população de soluções e não se baseia em fenômenos da natureza. Proposto por Glover (1977), tem sido frequentemente utilizado em problemas de otimização. O método utiliza estratégias de intensificação e diversificação de busca, mostrando ser eficiente nos mais diversos problemas de otimização (LAGUNA; MARTÍ, 2003). Na Figura 1 o algoritmo BD é apresentado em fluxograma do algoritmo.

Figura 1. Fluxograma da meta-heurística BD

Fonte: (GARCIA, 2019)

Como pode ser observado na Figura 1, o algoritmo inicia com o carregamento dos parâmetros do problema a ser otimizado e a definição dos parâmetros gerais da meta-heurística, como seguem:

- $PSize$: Número de elementos do conjunto P de soluções
- $b = b1 + b2$: Número de elementos do conjunto de referência $RefSet$
- $b1$: Número de elementos de qualidade do conjunto $RefSet$
- $b2$: Número de elementos de diversidade do conjunto $RefSet$

1.3.1 Construção do Conjunto P

O tamanho padrão do conjunto P é considerado como o máximo entre 100 e $5b$. A construção deste conjunto deve levar em conta a diversidade e qualidade das soluções e para isso é utilizada a técnica de randomização controlada utilizando memória baseada em frequência, como é detalhado em (LAGUNA; MARTÍ, 2003), (GARCIA, 2019) e (GARCIA; MACEDO; ROMERO, 2018).

1.3.2 Construção do Conjunto RefSet Inicial

A construção do conjunto $RefSet$ inicial consiste em inserir em $RefSet$ os $b1$ elementos de P de melhor qualidade e os $b2$ elementos de P mais diversos em relação aos elementos já alocados no conjunto $RefSet$. De acordo com (LAGUNA; MARTÍ, 2003), inicialmente inicia-se $b = 10$, com $b1 = 5$ e $b2 = 5$, em seguida, por realização de testes, os parâmetros $b1$ e $b2$ devem ser ajustados para melhor atender o problema.

Para obter os $b2$ elementos com diversidade, é utilizada a distância euclidiana entre os elementos já constantes do conjunto $RefSet$ e os elementos do conjunto P. Dessa forma, para cada elemento constante no conjunto P, exceto os já constantes em $RefSet$, é encontrada a menor distância euclidiana entre os elementos constantes no conjunto $RefSet$. Seleciona-se o elemento de P que possuir a maior distância entre as mínimas (LAGUNA; MARTÍ, 2003) , (GARCIA, 2019) e (GARCIA; MACEDO; ROMERO, 2018).

1.3.3 Construção do Conjunto Pool

O conjunto $Pool$ é composto por combinações entre os elementos do conjunto $RefSet$. Como pode ser visto em (LAGUNA; MARTÍ, 2003), existem várias técnicas de combinação entre soluções para popular $Pool$. Neste trabalho será utilizada uma combinação matemática simples de 2 a 2 elementos de $RefSet$ (GARCIA, 2019) e (GARCIA; MACEDO; ROMERO, 2018). O processo consiste em selecionar elementos de $RefSet$ e gerar subconjuntos S, e a partir dos subconjuntos S, realiza-se as combinações para popular o conjunto $Pool$, como é exemplificado na Figura 2.

Após a construção de cada conjunto, P, $RefSet$ e $Pool$, os elementos são ordenados da solução de melhor para a solução de pior qualidade.

1.3.4 Atualização do Conjunto RefSet

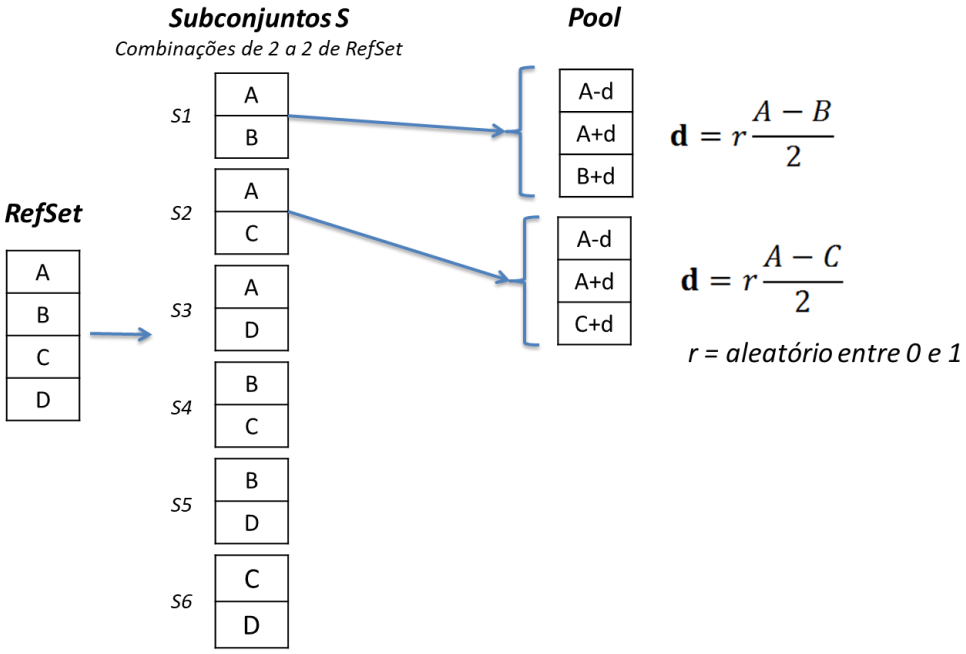
Após concluir a montagem do conjunto $Pool$, o conjunto $RefSet$ é atualizado com as b melhores soluções da união dos dois conjuntos $Pool \cup RefSet$.

Se o conjunto $RefSet$ sofre alteração, reordena-se o conjunto da solução de melhor qualidade para a de pior, um novo conjunto $Pool$ é construído, e o processo entra num looping, como pode ser observado na Figura 1.

Caso contrário, se não houve atualização de $RefSet$, e os critérios de parada não são satisfeitos, o conjunto P é reconstruído, os elementos de diversidade de $RefSet$ são substituídos, e o processo novamente entra em looping (Figura 1).

Ao término da execução do algoritmo, a solução ótima se encontra no primeiro elemento de $RefSet$.

Figura 2. Exemplo da formação dos subconjuntos S e Pool



Fonte: (GARCIA, 2019)

2 SOLUÇÃO DO PROBLEMA DO CAIXEIRO VIAJANTE PELA META-HEURÍSTICA BUSCA DISPERSA

Nesta Seção serão detalhadas as particularidades do algoritmo da meta-heurística busca dispersa, apresentado na Figura 1, para a solução do problema do caixeiro viajante, o qual foi implementado na linguagem C/C++.

O problema é tratado como um problema de otimização combinatorial e não serão utilizados modelos matemáticos. A Figura 3 mostra um exemplo de vetor como proposta de solução para o problema do caixeiro viajante fornecido pela meta-heurística.

Figura 3. Vetor proposta de solução para o problema do caixeiro viajante

5	10	1	4	2	3	6	7	8	9

Fonte: Elaboração do autor.

Neste vetor exemplo (Figura 3) é ilustrada uma solução de uma instância do problema com dez cidades, numeradas de 1 a 10. Observa-se que a cidade de partida é a de número 5, em seguida a de número 10 e assim por diante. A última cidade a ser visitada é a de número 9, e como se sabe que após a última cidade visitada o caixeiro viajante volta para a cidade de origem, no caso a cidade 5, não há a necessidade de armazená-la.

As instâncias dos problemas estão armazenadas em arquivos do tipo texto, onde cada linha do arquivo contém três colunas, sendo a primeira o número da cidade, a segunda a coordenada geográfica x e a terceira a coordenada geográfica y da cidade. As distâncias entre as cidades são calculadas com base na distância euclidiana, como mostra a expressão (18), e neste exemplo é calculada a distância entre as cidades 1 e 2.

$$d_{12} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{18}$$

Para facilitar a construção do algoritmo e também para obter um melhor desempenho na consulta das distâncias entre as cidades, na etapa inicial do algoritmo o arquivo texto contendo a instância do problema é lido e todas as distancias entre todas as cidades são armazenadas em uma matriz simétrica, onde um elemento desta matriz, por exemplo $M[3,2]$, contém a distância entre a cidade 3 e 2. Considera-se neste trabalho instâncias simétricas do problema do caixeiro viajante, ou seja, a distância entre as cidades 3 e 2 é igual à distância entre as cidades 2 e 3, e sendo assim, $M[3,2] = M[2,3]$.

O conjunto P inicial é obtido utilizando dois algoritmos, um algoritmo guloso com base na menor distância entre as cidades e o outro com base na randomização controlada com memória baseada em frequência.

Considerando N o número de cidades da instância do problema, o algoritmo guloso irá gerar o número N soluções de P para $N <= PSize$. A solução 1 de P iniciará o *tour* da cidade 1, a solução 2 de P iniciará o *tour* da cidade 2, e assim por diante.

Quando N for menor que $PSize$, as demais soluções $PSize - N$ serão geradas com o algoritmo que utiliza a randomização controlada com memória baseada em frequência para obter elemento a elemento de cada solução. Para isto, as cidades são consideradas como números contínuos, e após obter um elemento da solução, este valor contínuo é arredondado para ser um número inteiro e logo em seguida, é feita uma verificação de duplicidade de cidade na solução. Se houver duplicidade, este elemento da solução é descartado e um outro elemento com base na menor distância da cidade anterior é obtido.

A construção do conjunto P após a primeira iteração, utiliza-se apenas a randomização controlada com memória baseada em frequência, descrito anteriormente.

Nesta implementação foi considerado $PSize = 100$, e para o conjunto $RefSet$ foi utilizada a configuração $b = 10$, sendo $b1 = 1$ e $b2 = 9$.

A parcela dos elementos do conjunto $RefSet$ responsáveis pela diversidade são obtidos da mesma forma apresentada na seção de BD anterior, o que é padrão da meta-heurística busca dispersa. Neste caso, as cidades são consideradas como números contínuos para obter as distâncias euclidianas.

Para gerar as soluções do conjunto $Pool$, também são realizadas combinações de dois a dois elementos de $RefSet$ (conjunto S (x' e x'')), gerando-se assim três soluções testes (*trial*), conforme exemplificado na Figura 2.

A expressão (19) é utilizada para realizar a combinação das variáveis entre as soluções x' e x''.

$$\begin{aligned} &x_i \text{ da combinação 1} = x_i' - \text{round}(d_{x_i}) \\ &x_i \text{ da combinação 2} = x_i' + \text{round}(d_{x_i}) \\ &x_i \text{ da combinação 3} = x_i'' + \text{round}(d_{x_i}) \\ &\qquad\qquad para\ i = 1, 2, \dots n \end{aligned} \tag{19}$$

Em que:

- r : Número aleatório entre 0 e 1;
- x_i' : *i*-ésima cidade da solução x' do subconjunto S;
- x_i'' : *i*-ésima cidade da solução x'' do subconjunto S;
- $d_{x_i} = r\frac{x_i'' - x_i'}{2}$: Corresponde a um incremento da variável x_i;
- n : Número de cidades da instância do problema.

Após obter o valor de x_i de cada combinação, este número é arredondado para um número inteiro, o limite entre 1 e n é ajustado, e em seguida é verificada a duplicidade de cidades no respectivo vetor solução. Caso haja duplicidade, esta cidade é descartada e uma nova cidade é obtida com base na menor distância da cidade anterior da solução.

3 SOLUÇÃO DO PROBLEMA DO FLUXO DE POTÊNCIA ÓTIMO PELA META-HEURÍSTICA BUSCA DISPERSA

Nessa seção é apresentada a metodologia utilizando a meta-heurística BD para resolver o problema de FPO. A implementação foi feita utilizando a linguagem de programação C/C++ (GARCIA, 2019).

A formulação matemática utilizada para o problema de FPO é a mesma apresentada nas expressões (1)–(17), ou seja, será considerada a otimização no custo de produção de energia elétrica.

As incógnitas do problema de FPO que a metodologia soluciona são apresentadas na Tabela 1. Além disso, são calculados os fluxos de potência, ou seja, as variáveis P_{kij}^{de}, P_{kij}^{para}, Q_{kij}^{de} e Q_{kij}^{para} do problema, definidas respectivamente pelas expressões (4)–(7).

A construção do conjunto P nesse caso utiliza a mesma metodologia apresentada na Seção 1.3, ou seja, utilizando randomização controlada com memória baseada em frequência para as variáveis contínuas. Já os valores das variáveis ω_i e t_i são obtidos de maneira

totalmente aleatória. Nesse trabalho foi considerado $PSize = 100$. Para o conjunto $RefSet$ foi utilizada a configuração $b = 10$, sendo $b1 = 2$ e $b2 = 8$.

Tabela 1 – Variáveis do problema do FPO calculadas pela metodologia BD-FPO

Variável	Descrição	Tipo
V_i	Magnitude de tensão na barra i	Contínua
P_i^g	Potência ativa gerada na barra i	Contínua
Q_i^g	Potência reativa gerada na barra i	Contínua
θ_i	Ângulo de tensão na barra i	Contínua
ω_i	Variável binária de chaveamento do *shunt* i	Binária
${t_i}^2$	Posição do *tap* do transformador i, entre –16 e 16	Inteira

Fonte: (GARCIA, 2019).

Os elementos de diversidade do conjunto $RefSet$ são obtidos agora considerando as variáveis binárias e inteiras. A distância euclidiana entre os elementos de $RefSet$ e os elementos de P é calculada utilizando (20).

$$d = \sqrt{\sum_{i\in\Omega_b}(V_i' - V_i'')^2} + \sqrt{\sum_{i\in\Omega_b}(\theta_i{}' - \theta_i{}'')^2} + \sqrt{\sum_{i\in\Omega_g}\left(P_i^{g\prime} - P_i^{g\prime\prime}\right)^2} + \sqrt{\sum_{i\in\Omega_g}\left(Q_i^{g\prime} - Q_i^{g\prime\prime}\right)^2} + \sqrt{\sum_{i=1}^{nTaps}(t_i' - t_i'')^2} + \sqrt{\sum_{i=1}^{nShunts}|\omega_i' - \omega_i''|} \quad (20)$$

Em que:

- Ω_b : Conjunto de barras do sistema;
- Ω_g : Conjunto de barras geradoras;
- V_i' : Módulo de tensão na barra i da solução presente no conjunto P;
- V_i'' : Módulo de tensão na barra i da solução presente no conjunto $RefSet$;
- $P_i^{g\prime}$: Potência ativa gerada na barra i da solução presente no conjunto P;
- $P_i^{g\prime\prime}$: Potência ativa gerada na barra i da solução presente no conjunto $RefSet$;
- t_i' : Posição do *tap* no transformador i da solução presente no conjunto P;
- t_i'' : Posição do *tap* no transformador i da solução presente no conjunto $RefSet$;
- ω_i' : Valor binário do *shunt* i da solução presente no conjunto P;
- ω_i'' : Valor binário do *shunt* i da solução presente no conjunto $RefSet$;
- $nTaps$: Número de *taps* dos transformadores;
- $nShunts$: Número de *shunts* no sistema.

[2] Considera-se que as variáveis referentes ao ajuste dos *taps* dos transformadores estão organizadas em um vetor indexado pelo índice i.

Para cada elemento constante no conjunto P, é encontrada a menor distância euclidiana (20) entre os elementos constantes no conjunto $RefSet$. Seleciona-se o elemento de P que possuir a maior distância entre as mínimas.

Os subconjuntos S são constituídos de elementos de $RefSet$, os quais são combinados e as novas soluções geradas são armazenadas em $Pool$. A combinação aqui também é realizada utilizando combinação de dois a dois elementos de S (x' e x''), gerando-se assim três soluções testes que são incluídas em $Pool$. As expressões (21)–(23) ilustram essa combinação para as variáveis contínuas e inteiras do problema.

Combinação das variáveis contínuas de magnitude de tensão:

$$\begin{aligned} &V_i \text{ da combinação 1} = V_i' - d_{V_i} \\ &V_i \text{ da combinação 2} = V_i' + d_{V_i} \\ &V_i \text{ da combinação 2} = V_i'' + d_{V_i} \end{aligned} \qquad (21)$$

$$\forall i \in \Omega_b$$

Combinação das variáveis contínuas dos ângulos de tensão:

$$\begin{aligned} &\theta_i \text{ da combinação 1} = \theta_i' - d_{\theta_i} \\ &\theta_i \text{ da combinação 2} = \theta_i' + d_{\theta_i} \\ &\theta_i \text{ da combinação 3} = \theta_i'' + d_{\theta_i} \end{aligned} \qquad (22)$$

$$\forall i \in \Omega_b$$

Combinação das variáveis inteiras que definem a posição dos *taps*:

$$\begin{aligned} &t_i \text{ da combinação 1} = t_i' - \text{round}(d_{t_i}) \\ &t_i \text{ da combinação 2} = t_i' + \text{round}(d_{t_i}) \\ &t_i \text{ da combinação 3} = t_i'' + \text{round}(d_{t_i}) \end{aligned} \qquad (23)$$

$$para\ i = 1, 2, \dots nTaps$$

Em que:

- $d_{x_i} = r\frac{x_i'' - x_i'}{2}$: Corresponde a um incremento da variável "x_i", que pode ser V_i, θ_i ou t_i;
- r : Número aleatório entre 0 e 1;
- V_i' : Módulo de tensão na barra i da solução x'do subconjunto S;
- V_i'' : Módulo de tensão na barra i da solução x''do subconjunto S;
- θ_i' : Ângulo de tensão na barra i da solução x'do subconjunto S;
- θ_i'' : Módulo de tensão na barra i da solução x''do subconjunto S;
- $P_i^{g\prime}$: Potência ativa gerada na barra i da solução x'do subconjunto S;
- $P_i^{g\prime\prime}$: Potência ativa gerada na barra i da solução x''do subconjunto S;
- t_i' : Posição do *tap* do transformador i da solução x'do subconjunto S;
- t_i'' : Posição do *tap* do transformador i da solução x''do subconjunto S;
- Ω_g : Conjunto de barras de geração;
- $\Omega_{g\prime}$: Conjunto de barras de geração exceto a barra de referência;
- $nTaps$: Número de posições dos *taps* dos transformadores.

Caso algum valor de c_i fique abaixo do limite inferior da variável na respectiva barra/ramo, o limite inferior é assumido, e caso fique acima do limite superior, o limite superior é assumido. Após obter três soluções testes, as soluções são incorporadas ao conjunto $Pool$.

A técnica utilizada para calcular a parcela de combinação das variáveis binárias das soluções é através do cálculo de um valor, chamado aqui de $score$ para cada variável binária, com base no valor da função objetivo de cada solução a ser combinada (LAGUNA; MARTÍ, 2003) e (GARCIA, 2019).

O cálculo desse valor $score$ é demonstrado na expressão (24).

$$score_i = \frac{VFobj_1 \times \omega_i' + VFobj_2 \times \omega_i''}{VFobj_1 + VFobj_2}, para\ i = 1, 2, .., nShunts \quad (24)$$

Em que:

- $VFobj_1$: Valor da função objetivo da solução x';
- $VFobj_2$: Valor da função objetivo da solução x'';
- ω_i' : Valor do *i-ésimo shunt* (0 ou 1) da solução x';
- ω_i'' : Valor do *i-ésimo shunt* (0 ou 1) da solução x'';
- $nShunts$: Número de *Shunts* do sistema.

O valor dos *shunts* de cada uma das soluções *trial* teste é determinado de acordo com a expressão (25):

$$\omega_i = \begin{cases} 1, & se\ r \leq score(i) \\ 0, & se\ r > score(i) \end{cases} \quad (25)$$
$$para\ i = 1, 2, \ldots, nShunts$$

Em que:

- r : Valor aleatório entre 0 e 1;
- ω_i: Valor do *i-ésimo shunt* de cada solução *trial*;

A atualização do conjunto $RefSet$ é idêntica à maneira utilizada na otimização de funções multimodais restritas, ou seja, após concluir a montagem do conjunto $Pool$, o conjunto $RefSet$ é atualizado com as b melhores soluções da união dos dois conjuntos $Pool \cup RefSet$.

Um método de melhoria local é utilizado após a inclusão dos elementos de diversidade no conjunto $RefSet$, e é aplicada a todos seus elementos. O algoritmo simplex de *Nelder-Mead* é utilizado nas variáveis de magnitude e ângulo de tensão em todas as barras (GARCIA, 2019). Para as variáveis que representam a posição dos *taps* dos transformadores, é aplicado o método com base na sensibilidade da função objetivo. Aumenta-se em uma unidade o valor de um *tap* e se a função objetivo melhorar continua-se aumentando enquanto a melhoria continuar, até o valor máximo. Caso o valor da função objetivo não melhore aumentando, faz-se o contrário, diminui-se em uma unidade o valor de um *tap* e enquanto a melhoria continuar, continua-se diminuindo, até o valor mínimo. Isso é realizado para todos os ramos com transformadores.

Já no caso dos *shunts*, o valor binário é trocado de 0 para 1 ou de 1 para 0 e o valor da função objetivo é avaliado. Caso haja melhoria da função objetivo, mantém-se a troca, e caso não, o valor original é assumido. Isso é realizado para todos os *shunts* do sistema.

As variáveis referentes à potência ativa e potência reativa nas barras geradoras são obtidas no cálculo da função objetivo, mais precisamente no cálculo das restrições do sistema referentes ao balanço de potência ativa e reativa.

Com base nas leis de *Kirchhoff*, expressões (2) e (3), as seguintes igualdades têm que ser verdadeiras para que haja o balanço de potência.

$$P_i^g = P_i^d + \sum_{kji \in \Omega_l} P_{kji}^{para} + \sum_{kij \in \Omega_l} P_{kij}^{de} + V_i^2 G_i^{sh} \qquad \forall i \in \Omega_b \quad (26)$$

$$Q_i^g = Q_i^d + \sum_{kji \in \Omega_l} Q_{kji}^{para} + \sum_{kij \in \Omega_l} Q_{ij}^{de} - V_i^2 B_i^{sh} \omega_i \qquad \forall i \in \Omega_b \quad (27)$$

Nas barras de carga, P_i^g e Q_i^g assumem o valor zero. Já nas barras de geração são verificados os limites das variáveis P_i^g e Q_i^g. Caso o termo à direita da igualdade das expressões (26) e (27) fiquem fora dos limites das respectivas variáveis P_i^g e Q_i^g, são assumidos os limites inferior ou superior, dependendo do caso. Caso o termo à direita da igualdade fique dentro dos limites das respectivas variáveis, esse valor é assumido.

A função objetivo dessa abordagem é penalizada com as restrições de acordo com (28) como pode ser visto em Garcia (2019).

$$\min Pf = f + \sum_{i=1}^{l} \rho_i |h_i|^q + \sum_{i=1}^{m} \xi_i (\max\{0, g_i\})^q \quad (28)$$

Em que:

- f : Corresponde ao valor da função objetivo, expressão (1);
- Pf : Valor da função objetivo penalizada;
- ρ_i : Coeficiente de penalização para a *i*-ésima restrição de igualdade;
- ξ_i : Coeficiente de penalização para a *i*-ésima restrição de desigualdade;
- q : Expoente de penalização, sendo $q = 1$;
- h_i : Restrições de igualdade correspondentes às leis de Kirchhoff, das expressões (2) e (3);
- g_i : Restrições de desigualdade, correspondentes aos limites de fluxo de potência aparente nas linhas de transmissão, das expressões (11) e (12);
- l : Número de restrições de igualdade;
- m : Número de restrições de desigualdade.

Nesse caso optou-se em utilizar o expoente de penalização $q = 1$ para explorar melhor os valores de violação das restrições entre zero e um. Com a realização dos testes foi

possível observar que com um coeficiente de penalização maior para as restrições de desigualdade, o algoritmo teve um comportamento melhor.

Utilizou-se também coeficientes de penalização distintos para as restrições de igualdades (ρ_i) e desigualdades (ξ_i). Isso porque, também com a realização dos testes, verificou-se que escolhendo coeficientes de penalização maiores para restrições com valores maiores e coeficientes menores para restrições com valores menores, o método conseguia convergência mais rápida para uma solução factível.

Os limites das variáveis (i) posição dos *taps*, (ii) magnitudes e (iii) ângulos de tensão foram gerenciados na meta-heurística, ou seja, na etapa da construção do conjunto P, e na etapa de combinação das soluções para a geração do conjunto $Pool$.

4 RESULTADOS E DISCUSSÕES

Nesta Seção são apresentados os testes e resultados dos dois problemas tratados neste trabalho. Primeiramente serão apresentados os resultados do problema do caixeiro viajante e em seguida os resultados do problema de fluxo de potência ótimo.

4.1 Caixeiro Viajante

Foram utilizadas duas instâncias do problema de caixeiro viajante para avaliar o potencial da meta-heurística implementada. Uma instância contendo 75 cidades e a outra com 150 cidades. Com a finalidade de comparação de resultados, estas mesmas instâncias foram resolvidas através da modelagem matemática tradicional do problema utilizando o *solver* CPLEX. O algoritmo BD foi implementado na linguagem C/C++ e a execução do algoritmo foi realizada em um microcomputador com processador Intel® Core™ i5-7400 de 3,00 GHz e 8 GB de RAM

A Tabela 2 mostra os resultados das funções objetivo da resolução do problema com 75 e 150 cidades pelo CPLEX e pela busca dispersa. Além disso, os tempos gasto para a solução dos problemas também são mostrados. Como pode ser observado, os valores das funções objetivos de ambas as instâncias do problema, praticamente são os mesmos entre o CPLEX e a BD. Entretanto, a metodologia que utiliza a BD apresentou melhor desempenho em relação ao tempo gasto para a solução dos problemas.

Tabela 2 - Resultados dos testes do problema do caixeiro viajante

Instância	CPLEX		BUSCA DISPERSA	
	Função Objetivo (km)	Tempo (seg.)	Função Objetivo (km)	Tempo (seg.)
75 cidades	156,37	75	157,54	0,62
150 cidades	240,10	1048	241,57	204,51

Fonte: Elaboração do autor.

Considerando que, para instâncias maiores o CPLEX pode não convergir, ou demorar muito tempo para a convergência da solução, a metodologia que utiliza a BD pode ser uma alternativa viável para a solução deste tipo de problema.

4.1 Fluxo de Potência Ótimo

Os resultados da solução do problema de FPO pela BD são apresentados nesta seção. O algoritmo foi implementado na linguagem C/C++ e a execução do algoritmo foi realizada em um microcomputador com processador Intel® Core™ i5-7400 de 3,00 GHz e 8 GB de RAM (GARCIA, 2019).

São utilizados três sistemas teste, de 6, 14 e 57 barras, disponíveis em Coffrin, Gordon e Scott (2016). A Tabela 3 e A fim de comparar os resultados, esses mesmos sistemas foram resolvidos com o *solver* KNITRO, o qual utiliza o método de pontos interiores, considerando a modelagem matemática do problema de FPO apresentada neste trabalho. A implementação utilizou a linguagem AMPL.

Tabela 4 mostram algumas características desses sistemas.

Tabela 3 – Características gerais dos sistemas teste utilizados pela BD-FPO – Parte 1

Sistema	Nº de Barras	Nº de Linhas	Barras de Geração	Barras de Carga	Nº de Transformadores	Nº de *Shunts*
nesta_case6_ww	6	11	3	3	1	0
nesta_case14_ieee	14	20	5	9	3	1
nesta_case57_ieee	57	80	7	50	17	3

Fonte: (GARCIA, 2019)

A fim de comparar os resultados, esses mesmos sistemas foram resolvidos com o *solver* KNITRO, o qual utiliza o método de pontos interiores, considerando a modelagem matemática do problema de FPO apresentada neste trabalho. A implementação utilizou a linguagem AMPL.

Tabela 4 – Características gerais dos sistemas teste utilizados pela BD-FPO – Parte 2

Sistema	Carga Ativa	Carga Reativa Total (MVAr)	Capacidade de Geração Ativa (MW)	Capacidade de Geração Reativa (MVAr)

	Total (MW)			
nesta_case6_ww	210,000	210,500	530,000	300,000
nesta_case14_ieee	259,000	73,500	425,000	130,000
nesta_case57_ieee	1250,800	336,400	1377,000	611,000

Fonte: (GARCIA, 2019)

A Tabela 5 mostra os valores das funções objetivos em (US$/hora) obtidos a partir das soluções do problema de FPO para os três sistemas teste utilizando a BD e o *solver* KNITRO.

Tabela 5 – Comparação das funções objetivo referentes aos resultados da solução do problema de FPO por AMPL+KNITRO e BD

Sistema	AMPL+KNITRO		BD	
	Função Objetivo (US$/hora)	Factibilidade	Função Objetivo (US$/hora)	Factibilidade
nesta_case6_ww	3127,5704	Factível	3137,0032	Factível
nesta_case14_ieee	243,9903	Factível	207,4335	Infactível
nesta_case57_ieee	1173,0094	Factível	993,5857	Infactível

Fonte: (GARCIA, 2019)

Como pode ser observado na Tabela 5, a BD não conseguiu soluções factíveis para o problema em dois casos, nos sistemas de 14 e 57 barras, e o modelo utilizando o *solver* conseguiu soluções factíveis para os três casos. A infactibilidade se deu pelo balanço de potência nas barras, sendo que todos os outros limites, incluindo os limites de fluxo de potência aparente nas linhas de transmissão foram respeitados.

5 CONCLUSÃO

Com a realização deste trabalho, conclui-se que a meta-heurística busca dispersa é uma técnica de otimização promissora e que pode ser aplicada em problemas de qualquer natureza.

A BD especializada para resolver o problema do caixeiro viajante mostrou ser uma ferramenta robusta e eficiente, pois em comparação com os resultados obtidos do *solver* CPLEX, o qual utilizou a formulação matemática do problema, a BD apresentou resultados de funções objetivo muito próximos e com um tempo computacional muito menor.

Embora a implementação da BD para a solução do problema de FPO não conseguiu soluções factíveis em dois problemas teste, considera-se um passo dado na resolução deste tipo de problema utilizando esta meta-heurística, uma vez que por ser um problema de PNLIM é de difícil solução, e, ainda um desafio na área de engenharia elétrica. Em trabalhos futuros, pode-se utilizar *solvers* em conjunto com esta implementação para chegar a soluções factíveis de boa qualidade.

Por fim, conclui-se que o objetivo do trabalho foi alcançado, pois apresentou duas metodologias de implementação da BD para a resolução de problemas de qualquer natureza.

REFERÊNCIAS

ATTIA, A.-F.; EL SEHIEMY, R. A.; HASANIEN, H. M. Optimal power flow solution in power systems using a novel Sine-Cosine algorithm. **International Journal of Electrical Power & Energy Systems**, London, v. 99, p. 331–343, jul. 2018.

BAKIRTZIS, A. G.; BISKAS, P. N.; ZOUMAS, C. E.; PETRIDIS, V. Optimal power flow by enhanced genetic algorithm. **IEEE Transactions on Power Systems**, Piscataway, v. 17, n. 2, p. 229–236, mai. 2002.

BENEVIDES, P. **Aplicação de heurísticas e metaheurísticas para o problema do caixeiro viajante em um problema real de roterização de veículos**. 2011. Universidade Federal do Paraná, [s. l.], 2011. Disponível em: <http://dspace.c3sl.ufpr.br:8080/dspace/handle/1884/26793>

CAPITANESCU, F.; WEHENKEL, L. An AC OPF-based heuristic algorithm for optimal transmission switching. In: 2014 POWER SYSTEMS COMPUTATION CONFERENCE 2014, Wroclaw. **Anais**... Wroclaw: IEEE,ago. 2014. Disponível em: <http://ieeexplore.ieee.org/document/7038445/>

CARPENTIER, J. Contribution a l'etude du dispatching economique. **Bulletin de la Societe Francaise des Electriciens**, London, v. 3, n. 1, p. 431–447, ago. 1962.

CARPENTIER, J. Optimal power flows. **International Journal of Electrical Power & Energy Systems**, [s. l.], v. 1, n. 1, p. 3–15, 1979.

COFFRIN, C.; GORDON, D.; SCOTT, P. NESTA, The NICTA Energy System Test Case Archive. **arXiv preprint**, Ithaca, p. 1–26, ago. 2016.

GARCIA, A. M. **Algoritmo de Busca Dispersa aplicado ao problema de Fluxo de Potência Ótimo considerando o Desligamento de Linhas de Transmissão**. 2019. Universidade Estadual Paulista - Ilha Solteira, [s. l.], 2019.

GARCIA, A. M.; MACEDO, L. H.; ROMERO, R. Algoritmo de busca dispersa para otimização de funções contínuas multimodais com restrições. **L SBPO – Simpósio Brasileiro de Pesquisa Operacional**, Rio de Janeiro, ago. 2018.

GAVISH, B. . G. S. C. Travelling Salesman Problem and Related Problems. **The Office of Naval Research**, [s. l.], v. 1, n. 1, p. 32, 1978.

GLOVER, F. Heuristics for integer programming using surrogate constraints. **Decision Sciences**, Hoboken, v. 8, n. 1, p. 156–166, jan. 1977.

GRANVILLE, S. Optimal reative dispach through interior point methods. **IEEE Transactions on Power Systems**, Piscataway, v. 9, n. 4, p. 136–146, fev. 1994.

HUA WEI; SASAKI, H.; KUBOKAWA, J.; YOKOYAMA, R. An interior point nonlinear programming for optimal power flow problems with a novel data structure. **IEEE Transactions on Power Systems**, Piscataway, v. 13, n. 3, p. 870–877, ago. 1998.

IBA, K. Reactive power optimization by genetic algorithm. **IEEE Transactions on Power**

Systems, Piscataway, v. 9, n. 2, p. 685–692, mai. 1994.

JIANG, J.; HAN, X.; WANG, J.; ZHU, X.; SUN, D.; MA, Y. Optimal power flow with transmission switching for power system with wind/photovoltaic generation. In: 2017 CHINESE AUTOMATION CONGRESS (CAC) 2017, Jinan. **Anais**... Jinan: IEEE,out. 2017. Disponível em: <http://ieeexplore.ieee.org/document/8243820/>

KHANABADI, M.; GHASEMI, H.; DOOSTIZADEH, M. Optimal transmission switching considering voltage security and N-1 contingency analysis. **IEEE Transactions on Power Systems**, Piscataway, v. 28, n. 1, p. 542–550, fev. 2013.

LAGUNA, M.; MARTÍ, R. **Scatter Search:Methodology and Implementations in C**. New York: Springer Science, 2003.

LEE, K. Y.; VLACHOGIANNIS, J. G. Optimization of power systems based on ant colony system algorithms: An overview. In: PROCEEDINGS OF THE 13TH INTERNATIONAL CONFERENCE ON, INTELLIGENT SYSTEMS APPLICATION TO POWER SYSTEMS 2005, Arlington. **Anais**... Arlington: IEEE,nov. 2005. Disponível em: <http://ieeexplore.ieee.org/document/1599237/>

LOW, S. H. Convex Relaxation of Optimal Power Flow—Part I: Formulations and Equivalence. **IEEE Transactions on Control of Network Systems**, [s. l.], v. 1, n. 1, p. 15–27, 2014.

MOHAMED, A.-A. A.; MOHAMED, Y. S.; EL-GAAFARY, A. A. M.; HEMEIDA, A. M. Optimal power flow using moth swarm algorithm. **Electric Power Systems Research**, Amsterdam, v. 142, p. 190–206, jan. 2017.

ORMAN, A. J.; WILLIAMS, H. P. A Survey of Different Integer Programming Formulations of the Travelling Salesman Problem. **Power International**, [s. l.], n. 1991, p. 93–104, 2007.

POTLURI, T.; HEDMAN, K. W. Impacts of topology control on the ACOPF. In: 2012 IEEE POWER AND ENERGY SOCIETY GENERAL MEETING 2012, San Diego. **Anais**... San Diego: IEEE,jul. 2012. Disponível em: <http://ieeexplore.ieee.org/document/6345676/>

SALKUTI, S. R. Congestion management using optimal transmission switching. **IEEE Systems Journal**, Piscataway, v. 12, n. 4, p. 3555–3564, dez. 2018.

SUN, D.; ASHLEY, B.; BREWER, B.; HUGHES, A.; TINNEY, W. Optimal power flow by newton approach. **IEEE Transactions on Power Apparatus and Systems**, Piscataway, v. PAS-103, n. 10, p. 2864–2880, out. 1984.

SURENDER REDDY, S.; BIJWE, P. R. Efficiency improvements in meta-heuristic algorithms to solve the optimal power flow problem. **International Journal of Electrical Power & Energy Systems**, London, v. 82, p. 288–302, nov. 2016.

TORRES, G.; QUINTANA, V. An interior-point method for nonlinear optimal power flow using voltage rectangular coordinates. **IEEE Transactions on Power Systems**, Piscataway, v. 13, n. 4, p. 1211–1218, nov. 1998.

YURYEVICH, J.; KIT PO WONG. Evolutionary programming based optimal power flow algorithm. **IEEE Transactions on Power Systems**, Piscataway, v. 14, n. 4, p. 1245–1250, nov. 1999.