Universidade Federal de Juiz de Fora
Instituto de Ciências Exatas
Bacharelado em Ciência da Computação

Análise de Aplicação de Técnicas de Otimização Binária por Enxame de Partículas para Alocação de Fontes de Geração Distribuída

Lucas Lima Yoshida

JUIZ DE FORA
MARÇO, 2013

Análise de Aplicação de Técnicas de Otimização Binária por Enxame de Partículas para Alocação de Fontes de Geração Distribuída

Lucas Lima Yoshida

Universidade Federal de Juiz de Fora

Instituto de Ciências Exatas

Departamento de Ciência da Computação

Bacharelado em Ciência da Computação

Orientador: Luciana Conceição Dias Campos

Co-orientador: Bruno Henrique Dias

JUIZ DE FORA

MARÇO, 2013

Análise de Aplicação de Técnicas de Otimização Binária por Enxame de Partículas para Alocação de Fontes de Geração Distribuída

Lucas Lima Yoshida

MONOGRAFIA SUBMETIDA AO CORPO DOCENTE DO INSTITUTO DE CIÊNCIAS EXATAS DA UNIVERSIDADE FEDERAL DE JUIZ DE FORA, COMO PARTE INTEGRANTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE BACHAREL EM CIÊNCIA DA COMPUTAÇÃO.

Aprovada por:

Luciana Conceição Dias Campos
Doutora em Engenharia Elétrica

Bruno Henriques Dias
Doutor em Engenharia Elétrica

Heder Soares Bernardino
Doutor em Modelagem Computacional

JUIZ DE FORA

25 DE MARÇO, 2013

A todos que tornaram esse trabalho possível.

Resumo

A população demanda cada vez mais energia elétrica e a sobrecarga gerada sobre o sistema elétrico o apresenta como um sistema incapaz de suportar a mesma. A instalação de unidades de geração distribuída (GD), tendem a aliviar a carga sobre o sistema e ao mesmo tempo garantir um serviço de qualidade aos clientes finais. Porém, a alocação adequada de centros de geração distribuída não é uma tarefa simples, sendo necessário um estudo para encontrar os pontos de menor perda de potência ao longo do sistema, sendo este um problema de otimização complexo. Frente a esses problemas, a aplicação de técnicas bio-inspiradas como a Otimização por Enxame de Partículas (PSO) se apresenta como uma solução interessante de estudo e pesquisa. Através do comportamento semelhante ao dos pássaros, esse algoritmo realiza uma busca no espaço de soluções, convergindo para os pontos considerados idéias para a instalação e ativação de uma GD dentro do sistema, tendo assim a menor perda de potência. Nesse trabalho são apresentadas duas variantes do Algoritmo PSO chamados de PSO Binário, com o propósito de obter bons pontos de instalação de uma GD. Finalmente os resultados obtidos neste trabalho serão comparados com dados na literatura.

Palavras-chave: Geração Distribuída, Otimização, Otimização por Enxame de Partículas, Otimização Binária por Exame de Partículas.

Abstract

The people demand more and more electricity and the overload on the electrical system presents it as a system inefficient and unstable. The installation of distributed generation units (DG), lighten the load on the system while ensuring a quality service to end customers. However, the allocation of distributed generation centers is not a simple task, requiring a study to find the points with lower voltage loss throughout the system, which is a complex optimization problem. For these problems, the application of bio-inspired techniques such as Particle Swarm Optimization (PSO) presents as an interesting study and research. Through the similar behavior of the birds, this algorithm performs a search in the solution space, converging on the points considered ideas for installing and activating a GD on the system, having a lower loss. In this paper we present two variants of PSO algorithm called Binary PSO, in order to obtain the finer points of installing a GD. And the results of this study will be compared with literature.

Keywords: Distributed Generation,Optimization, Particle Swarm Optimization, Binay-Particle Swarm Optimization .

Agradecimentos

A minha mãe, Katia, pelo apoio e o carinho nos momentos difíceis, fundamental para a conclusão desse trabalho.

A meu pai, Julio, e meu irmão, Matheus, que mesmo longe deram apoio no desenvolvimento desse trabalho.

A minha namorada, Gina, que mesmo com a minhas eventuais ausências, não deixou de me apoiar.

A todos os meus parentes, pelo encorajamento e apoio.

A professora Luciana pela orientação e principalmente, pela paciência ao longo desse projeto.

Ao professor Bruno pela orientação das ferramentas e pelo apoio na análise dos problemas.

Aos meus amigos que me forneceram apoio ao longo de todo o projeto, inclusive me cobrando o andamento do mesmo

Aos professores do Departamento de Ciência da Computação pelos seus ensinamentos.

"Everybody in this country should learn how to program a computer...because it teaches you how to think".

Steve Jobs

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviações

GD	Geração Distribuída
PSO	*Particle Swarm Optimization*
FPO	Fluxo de Potência Ótimo

1 Introdução

1.1 Contextualização

A demanda por energia elétrica cresce a cada momento. A população liga cada vez mais aparelhos elétricos e passa a depender mais destes. Com o aumento gradativo da carga sobre o sistema elétrico, espera-se que os sistemas de distribuição sejam capazes de suprir essa necessidade e sejam robustos o suficiente para tolerar falhas. Com os constantes problemas nos sistemas elétricos, recentemente apresentados na mídia, fica claro que o sistema elétrico não está preparado para a sobrecarga que é gerada, se apresentando cada vez mais deficiente e instável (Liu et al (2007)).

Além dos problemas de sobrecarga, existem outros fatores que limitam a expansão do sistema para garantir o fornecimento contínuo e de qualidade a toda população. Dentre os principais podemos citar os fatores ambientais, governamentais e a falta de padronização entre a parte estatal e a privatizada do sistema elétrico (Dias et al (2012)).

Avanços em pesquisas relacionadas ao fornecimento de energia, propõe dentre algumas soluções para o problema descrito o uso de Geração Distribuída (GD), como uma alternativa para a manutenção do abastecimento. Esses são conectados diretamente nos sistema de distribuição ou clientes finais, com o intuito de ajudar a sustentar o sistema.

A instalação de centros de geração distribuída apresenta uma série de vantagens ao sistema elétrico atual, destacando-se o aumento da qualidade da energia, ou seja o fornecimento em conformidade com parâmetros seguros de tensão e corrente, redução da sobrecarga nos sistemas de distribuição e transmissão, redução do volume de investimentos e perdas do sistema. Em contrapartida, uma definição mal planejada pode gerar mais desvantagens que vantagens sobre o sistema, como perda de confiabilidade do sistema e violação de limites de tensão em relação ao permitido (Dias et al (2012)).

Definir os pontos de instalação de um centro de uma GD não é uma tarefa trivial, sendo esta caracterizada como um problema de otimização complexo. A viabilidade é feita através de técnicas de otimização, dentre elas as bio-inspiradas como a Otimização

por Colônia de Formigas (ACO, do inglês *Ant Colony Optimization*) e a Otimização por Exame de Partículas (PSO, do inglês *Particle Swarm Optimization*) (Krueasuk et al (2006)) (Dias et al (2012)).

Nesse trabalho, serão avaliadas a aplicação de duas versões binárias do PSO, como alternativas para resolver o problema de alocação de GD. Estes algoritmos percorrem o espaço de soluções, que, neste caso, são os pontos possíveis de instalação de uma fonte de GD. Faz parte do objetivo, além da instalação de uma GD, obter um sistema com pouca perda. Os pontos obtidos pela execução do PSO, serão avaliados por outro algoritmo, o FPOLINGO, usado para calcular o fluxo de potência ótimo de uma rede elétrica. Esse irá avaliar as perdas do sistema, a fim de eleger, dentro do espaço de soluções obtidas no PSO, os pontos ideais de instalação de uma fonte de GD.

1.2 Motivação

A energia elétrica pode ser considerada um item básico importante no dia a dia das pessoas, pois há uso de muitos equipamentos eletrônicos. A fim de garantir a qualidade do serviço de distribuição de energia, o sistema de fornecimento deve ser robusto e tolerante a falhas. A definição de técnicas que permitem chegar a esse ponto de estabilidade são importantes para o crescimento de uma nação.

Dentre as técnicas disponíveis, que fornecem suporte a um sistema contínuo de fornecimento de energia, destacamos a definição de pontos de geração distribuída sendo esses instalados de forma estratégica. O problema de alocação ótima de Geração Distribuída se caracteriza como um problema de otimização complexo, sendo necessário o uso de técnicas para solucioná-lo (Dias et al (2012)).

Uma das técnicas de otimização bio-inspiradas, a Otimização por Enxame de Partículas, apresenta características que são interessantes para a análise do problema em questão, como a sua fácil implementação. Na literatura existem aplicações do PSO original ao problema de alocação de GD. A análise binária representa uma alternativa interessante para avaliar soluções para o problema, visto que o mesmo é uma aplicação tipicamente binária. No cas,o a alocação de uma fonte, definindo a mesma como ativa (1) ou não ativa (0). De forma geral, a escolha da técnica PSO apresenta vantagens para o estudo, pois

se trata de um algoritmo de fácil implementação, com baixo custo computacional e que apresenta resultados com bom desempenho.

1.3 Objetivos

O objetivo desse estudo é, a partir da execução para diferentes variações da versão binária do PSO, definir bons pontos de alocação de GD. Esses possíveis pontos, que repesentam os possíveis pontos ótimos de instalação de uma GD, serão posteriormente analisados em outro algoritmo, o FPOLINGO, que irá realizar o dimensionamento dos pontos de GD, definindo a potência de cada um, a partir dessas informações definir as perdas no sistema, a fim de identificar as soluções ótimas.

Os resultados obtidos com as execuções dos algoritmos implementados serão comparados a outros dados presentes na literatura. Dessa forma, permiti-se identificar se a solução proposta neste trabalho obtém os melhores resultados. O objetivo com a versão binária do PSO é atingir os pontos ótimos para alocação de centros de GD com baixo custo computacional.

1.4 Organização do Trabalho

O restante desse trabalho é organizado da seguinte forma: o capítulo 2 apresenta toda a fundamentação teórica dos estudos realizados. O capítulo 3 aborda as versões binárias do PSO abordadas nesse trabalho, juntamente com a descrição da implementação da solução proposta para o sistema. O capítulo 4 apresenta o estudo de caso e a comparação dos resultados obtidos com dados da literatura e por fim no capítulo 5 são feitas as conclusões finais e indicação de trabalhos futuros dentro do problema de alocação de GDs, os quais não foram cobertos neste trabalho.

2 Fundamentação Teórica

2.1 Geração Distribuída

A Geração Distribuída (GD) é um conceito importante, relativamente recente e relacionado à geração e fornecimento de energia.

Das diversas definições propostas, uma considerada ampla e bem objetiva foi descrita em (Ackermann et al (2001)), onde ele define uma GD como uma fonte de energia elétrica que se conecta diretamente aos centros de distribuição ou até mesmo aos clientes finais. Pode ser classificada segundo a sua capacidade de distribuição, podendo ser definida como mínima (até 5kW), pequena (de 5 kW a 5 MW), média (5MW a 50MW) e grande (Acima de 50MW).

Suas fontes podem ser renováveis, como a solar e a Eólica, ou não renováveis, como as termoelétricas.

Uma GD pode fornecer uma redução da carga existente sobre o sistema de distribuição de energia, reduzindo as quedas do mesmo e, consequentemente, melhorando a qualidade do serviço prestado. Entretanto, uma aplicação mal planejada de uma GD pode estimular ainda mais as quedas, diminuindo a confiabilidade no sistema. A instalação de uma GD deve ser bem planejada e mensurada. Espera-se que uma GD seja capaz de fornecer um maior suporte à tensão, suficiente para a manutenção do funcionamento do sistema elétrico, além de melhorar a qualidade da energia fornecida e reduzir as perdas. Outro ponto interessante sobre as instalações de uma GD é o fato que ela alivia a carga sobre os sistema de distribuição, assim como reduz o volume de investimento nesses, por aumentar a expectativa de duração das instalações existentes (Krueasuk et al (2006)) (Dias et al (2012)).

Em contrapartida, uma instalação ineficiente de uma GD pode trazer sérios problemas ao sistema elétrico no qual está conectada. Tal sistema passa a ser classificado como não confiável, por apresentar altas taxas de perdas de energia e ser de difícil administração, por exigir um maior controle da geração dentro do sistema. Outro problema

decorrente da má definição durante a instalação de uma GD é a possibilidade de uma violação dos limites pré-estabelecidos de tensão e corrente, podendo gerar danos ao sistema de transmissão e/ou aos clientes finais (Dias et al (2012)).

2.2 Otimização por Exame de Partículas - PSO

2.2.1 PSO Original

A Otimização por Exame de Partículas (PSO, do inglês *Particle Swarm Optimization*) é um algoritmo adaptativo de busca, baseado no comportamento dos bandos de passáros. Foi apresentado pela primeira vez em 1995, por Russel C. Eberhart e James Kennedy, sendo amplamente usado para resolução de problemas de otimização(Kennedy et al (1995)).

O comportamento padrão desse algoritmo é feito a partir de um conjunto de partículas distribuídas aleatoriamente pelo espaço de busca, onde cada uma dessas representa uma solução em potencial para o problema. As partículas percorrem um espaço multidimensional em busca da melhor solução para o problema. Cada partícula possui dois atributos, sua posição atual e sua velocidade. A cada iteração a velocidade e a posição das partículas são atualizadas e no final de cada iteração, cada uma das partículas são avaliadas para verificar se alguma delas representa uma solução ótima para o problema (Kennedy et al (1995)).

As partículas compartilham entre si a informação do melhor posicionamento obtido até o momento entre todas as partículas, representada pela variável *Gbest*. De forma análoga, cada partícula possui uma variável local que apresenta sua melhor posição até o momento, conhecida como *Pbest*. Essas informações são atualizadas a cada iteração e são utilizadas para o cálculo da velocidade de cada partícula, como indica a seguinte equação 2.1 (Eberhart et al (2001)):

$$V_i(t+1) = \omega.V_i(t) + c_1 o_1(Pbest_i - x_i) + c_2 o_2(Gbest - x_i) \tag{2.1}$$

onde $V_i(t+1)$ será a nova velocidade da partícula i, $V_i(t)$ é a velocidade atual, ω é definida

como a inércia do movimento, ou seja, indica o quanto a velocidade atual irá afetar no cálculo da nova velocidade e x_i é a atual posição da partícula. As variáveis c_1 e c_2 são constantes positivas definidas pelo usuário e o_1 e o_2 dois valores aleatórios obtidos no intervalo [0.0,1.0]. Um limite superior $Vmax$ é definido de forma que a partícula não atinja em um dado espaço uma velocidade muito alta.

A nova posição da partícula é calculada com base na posição atual e a nova velocidade calculada da partícula, como apresenta a equação 2.2 (Eberhart et al (2001)):

$$x_i(t+1) = x_i(t) + V_i(t+1) \tag{2.2}$$

onde $V_i(t+1)$ é a nova velocidade da partícula i, $x_i(t)$ é a posição atual da partícula e $x_i(t+1)$ é a nova posição obtida.

O algoritmo do PSO pode ser descrito, resumidamente como:

1. Inicializar a colônia de partículas randomicamente pelo espaço multidimensional de busca;
2. Calcular a qualidade da solução correspondente à cada partícula x_i;
3. Comparar a qualidade da solução com as armazenadas nas variáveis de melhor posição global ($Gbest$) e local ($Pbest_i$), atualizando as mesmas, caso o novo valor seja melhor que o valor atual;
4. Calcular e atualizar a velocidade e a posição de cada uma das partículas do conjunto;
5. Retornar ao passo 2, enquanto o critério de parada não é atendido.

2.2.2 PSO Binário

Proposta de Kennedy e Eberhart

A proposta original do PSO é aplicável a vários tipos de problemas de otimização, mas em alguns casos, como problemas de roteamento e agendamento, é necessário trabalhar com variáveis discretas no lugar de contínuas. O PSO original não possui suporte a tal tipo de variável, trabalhando somente com variáveis contínuas. Os pesquisadores que

desejavam aplicar o PSO nas soluções desse tipo de problema usavam arredondamentos de valores de ponto flutuante em valores binários.

Em (Kennedy et al (1997)), Kennedy e Eberhart avaliaram que qualquer problema contínuo ou discreto podia ser apresentado em termos binários. Em 1997 propuseram uma nova versão do PSO, chamada de PSO Binário, para aplicação em problemas que necessitam de valores discretos.

Na definição binária do PSO, a partícula se move no espaço a partir da variação de bits entre [0,1], sendo a velocidade definida como o número de bits alterados por iteração, ou seja, a partícula que alterar todos os bits está se movendo rapidamente pelo espaço, enquanto a que não possui alterações se mantém estática.

Dentro da proposta binária, existem poucas alterações em relação ao modelo original. A atualização das variáveis *Pbest* e *Gbest* se mantém inalterada em relação ao PSO original, assim como o cálculo da velocidade, sendo que no PSO Binário x_i, *Gbest* e $Pbest_i$ passam a ser cadeias de valores binários [0,1]. Nessa versão binária, a velocidade possui outro significado, passando a representar a probabilidade de um bit ter seu valor alterado para 1. Por exemplo, V_i=0,3 significa que o bit tem 30% de probabilidade de ter valor 1 e 70% de ser 0. Baseada nessa probabilidade é feita a atualização do valor x_i.

Para a definição da posição x_i da partícula, a equação 2.2 é substituída pela equação 2.3 (Kennedy et al (1997)):

$$se(rand() < S(vi))\ então\ x_i = 1\ senão\ x_i = 0 \tag{2.3}$$

onde $S()$ representa uma função sigmóide, que limita e ajusta os valores da velocidade no intervalo entre [0.0 , 1.0] e $rand()$ representa um valor numérico aleatório no intervalo de [0.0 , 1.0]. Observe que a trajetória da partícula que antes era ajustada de acordo com a velocidade na qual estava se deslocando, passa a ser feita a partir de uma validação da probabilidade do bit alterar seu valor. E o parâmetro $Vmax$ que antes limitava a velocidade, passa a ter uma influência inversa à versão original. No PSO original um valor alto em $Vmax$ definie uma busca de maior alcance, porém, na versão binária, um valor mais baixo em Vmax é que define um maior índice de mutação dos bits, isso ocorre visto que quanto maior o valor da velocidade, maior seu valor normalizado na função

sigmóide, portanto menor chance de alteração do valor do bit.

Os passos do PSO Binário proposto podem ser descritos, resumidamente como:

1. Inicializar a colônia de partículas randomicamente pelo espaço de busca;
2. Avaliar a qualidade da solução correspondente a partícula x_i;
3. Comparar a qualidade da solução com as armazenadas nas variáveis de melhor posição global ($Gbest$) e local ($Pbest$), atualizando as mesmas, caso o novo valor seja melhor que o valor atual;
4. Calcular e atualizar a velocidade e aplicar a função sigmóide para normalizar os dados;
5. Atualizar a posição de cada uma das partículas da colônia;
6. Retornar ao passo 2, enquanto o critério de parada não é atendido.

Proposta de Mojtaba

(Khanesar et al (2007)) avaliaram a versão original do PSO Binário e conseguiram identificar duas inconsistências em relação à versão do PSO original. Não foi somente a interpretação da velocidade e a trajetória que sofreram alterações, mas também a limitação da velocidade ($Vmax$) e a inércia(ω) passaram a ter um propósito opostos ao do PSO original.

Como citado anteriormente, no PSO original os valores altos do parâmetro $Vmax$ incentivam uma maior exploração do espaço de soluções. Já na versão binária, quanto menor o seu valor, maior a exploração, mesmo quando uma solução ótima já tenha sido obtida.

Com as mudanças entre a versão original e a binária, a inércia se torna um parâmetro difícil de definir. Valores inferiores a 1 podem impedir a convergência pois a velocidade $V_i(t)$ irá tender a ser sempre 0. Existem diferentes vertentes para definir o melhor valor para a inércia, dentre elas, destacamos duas: remover a inércia ou passar a utilizar valores randômicos entre [-1, 1]. A inércia é uma informação importante para a definição da velocidade, visto que representa o efeito da velocidade atual sobre a que

será calculada. Como o verificado pelos pesquisadores, a retirada desse parâmetro não gerou nenhuma alteração em desempenho, da mesma forma utilizar valores no intervalo [-1, 1] gerou algumas soluções não satisfatórias. Na sua proposta, estudada nessa sessão (Khanesar et al (2007)), utilizaram valores entre 0 e 1, decrescendo esse valor a cada iteração.

Outro problema identificado é a questão da alteração dos bits que definem a posição da partícula. Na versão original do PSO Binário tal alteração é independente dos valores atuais, ao contrario da versão pura do PSO, onde a posição atual é base para a definição da posição nova. Isso permite que ocorram mudanças aleatórias de posição dentro do espaço, perdendo a relação histórica de posições da partícula. Buscando resolver os problemas listados, foi proposto em 2007 uma versão alternativa do PSO Binário chamado de *Novel Binary Particle Swarm Optimization*. Nesta versão é apresentado como principais diferenças alterações na interpretação da velocidade e na forma de se atualizar a posição da partícula (Khanesar et al (2007)).

Na nova proposta, as variáveis *Pbest* e *Gbest* são atualizadas da mesma forma que na versão pura do PSO. Para o cálculo da velocidade são introduzidos dois novos vetores, não complementares V_0 e V_1, que representam a probabilidade de um bit se tornar 0 ou 1 respectivamente. A definição de qual desses será aplicado sobre a partícula, segue a regra apresentada pela equação 2.4 (Khanesar et al (2007))

$$|Vi| = \begin{cases} V1, & \text{se} \quad xi0, \\ V0, & \text{se} \quad xi1. \end{cases} \tag{2.4}$$

A velocidade pode ser obtida considerando parâmetros que tem seus valores definidos a partir do valor de Pbest e Gbest, sendo definido como nas equações 2.5(Khanesar et al (2007)):

$$\begin{aligned} &Se\ Pbest_i = 1\ então\ d_1i_1 = r_1c_1\ e\ d_0i_1 = -r_1c_1 \\ &Se\ Pbest_i = 0\ então\ d_0i_1 = r_1c_1\ e\ d_1i_1 = -r_1c_1 \\ &Se\ Gbest = 1\ então\ d_1i_2 = r_2c_2\ e\ d_0i_2 = -r_2c_2 \\ &Se\ Gbest = 0\ então\ d_0i_2 = r_2c_2\ e\ d_1i_2 = -r_2c_2 \end{aligned} \tag{2.5}$$

onde d são dois valores temporários usados na fórmula da velocidade, r são valores randômicos entre [0.0,1.0] e c são valores fixos, predefinidos pelo usuário. Sendo assim as velocidades V_0 e V_1 são calculadas como através da equação 2.6 (Khanesar et al (2007)):

$$\begin{aligned} {}_1 &= \omega.V_1 + d_1 i_1 + d_1 i_2 \\ V_0 &= \omega.V_0 + d_1 i_1 + d_1 i_2 \end{aligned} \tag{2.6}$$

Com a obtenção do valor da velocidade, é feito o calculo da nova posição de x. Diferente da versão do PSO binário original não utiliza-se a função sigmóide para a definição de que se um bit é 1 ou 0. Nesta nova proposta o valor de x_i pode ser mantido ou trocado pelo o complemento do valor atual, ou seja seu inverso, como o descrito na regra apresentada na equação 2.7:

$$x_i = \begin{cases} Inverso\ de\ x_i, & \text{se} \quad Rand()V_i, \\ x_i, & \text{se} \quad Rand()V_i. \end{cases} \tag{2.7}$$

onde V_i é a velocidade definida com base no descrito em 2.4 e $rand()$ representa um valor numérico aleatório no intervalo de [0.0 , 1.0].

Essa versão do algoritmo PSO Binário pode ser descrito, resumidamente como:

1. Inicializar o conjunto de partículas randomicamente pelo espaço de busca;
2. Avaliar a qualidade da solução correspondente à partícula x_i;
3. Comparar a qualidade da solução com as armazenadas nas variáveis de melhor posição global e local, atualizando as mesmas, caso o novo valor seja melhor que o valor atual;
4. Calcular e atualizar os valores das velocidades V_0 e V_1 de cada partícula;
5. Definir a velocidade de cada partícula;
6. Definir a nova posição das partículas;
7. Retornar para o passo 2, enquanto o critério de parada não é atendido.

2.2.3 Adaptação do PSO Original ao problema de alocação ótima de GDs

Para resolver do problema de alocação ótima de GDs foi utilizado um algoritmo baseado no PSO original (Dias et al (2012)). A implementação é dividida em dois módulos:

- PSOt (*Particle Swarm Optimization Toolbox*): desenvolvido em MATLAB® com um *toolbox* específico para PSO Original.
- FPOLINGO: uma aplicação desenvolvida no LINGO® , usada para o cálculo do Fluxo de Potência Ótimo (FPO). Com isso é possível determinar as perdas no sistema e identificar a melhor combinação de barras que determina a alocação de GDs ótima.

A aplicação desenvolvida é inicializada com o número de barras elétricas do sistema, unidades de GD a serem instaladas, o número partículas a serem utilizadas na execução e o total de iterações. Com base nos dados fornecidos é definido, de forma aleatória, uma matriz MxN (Número partículas X Número de Barras) como a solução inicial do problema. Em cada linha da matriz são alocadas aleatoriamente as barras ativas, neste caso, somente o número de barras ativas igual ao número de GDs a serem instaladas.

Na *toolbox* foram necessárias modificações, pois a mesma trabalha com variáveis contínuas enquanto a aplicação FPOLINGO, recebe somente valores binários. Essas alterações permitiram respostas em formato binário, através do arredondamento de valores de ponto flutuante.

As alterações propostas estabeleceram um limite para o espaço das partículas entre 0 e 1. A velocidade máxima da partícula ($Vmax$) também é limitada, tendo como seu valor máximo 1. A cada início de iteração, os valores da matriz de posição são arredondados e assim fornecidos para a função que executa a aplicação FPOLINGO. Esta calcula as perdas no sistema definido em cada partícula, e encontra a solução ótima (Dias et al (2012)).

Além das modificações já citadas, duas outras modificações foram implementadas. Definiu-se uma validação, em que, após 50 iterações, caso o melhor valor encontrado se mantenha o mesmo, as partículas são reinicializadas, assim como as suas velocidades.

Essa operação tem como objetivo sair de um possível ótimo local. Outra modificação realizada foi uma penalização para o caso que, se a melhor solução encontrada possui o número de barras ativas superior ao número definido. Nesse caso são adicionadas 10 unidades ao valor da função objetivo, visto que os valores da função objetivo são sempre inferiores a 1. Desta forma, o PSO consegue efetuar uma nova busca por outra solução.

A aplicação FPOLINGO é utilizada para definir, a partir das respostas obtidas pelo PSO no MATLAB®, qual é o fluxo de potência ótima no sistema, através do cálculo da perda obtida. Esta recebe como entrada um arquivo de configuração e a resposta obtida no PSO. O arquivo de configuração é composto pelas configurações das barras do sistema, como tensão, resistência, potência máxima e etc. Baseado nos valores de cada barra e nas lista de barras ativas, definidas pelo PSO, é obtida a configuração do melhor sistema, que possui a menor perda.

3 Estudo de Caso

3.1 Aplicação do PSO Binário ao problema de alocação ótima de GDs

Para o desenvolvimento desse trabalho, foram desenvolvidas duas versões com a aplicação do PSO Binário: uma baseada na aplicação original, obtida em (Dias et al (2012)) e adaptada para a versão Binária de Kenedy e Eberhart, e outra com uma nova implementação do PSO Binário baseada em (Khanesar et al (2007)). Ambas implementações utilizam a aplicação FPOLINGO. Elas são listadas abaixo:

3.1.1 Baseada na aplicação original

Com base no modelo existente, foi desenvolvida uma nova proposta, através de modificações na *toolbox* utilizada, para suportar o PSO Binário original, proposto por Kennedy e Eberhart (Kennedy et al (1997)). No desenvolvimento dessa solução, foram removidas todas as alterações efetuadas para a aplicação original no *toolbox*, exceto a estrutura para reinicializar o processo em caso de ficar preso em um mínimo local.

No modelo Binário cada posição da partícula é representada em valores binários, sendo cada bit o valor da partícula em cada dimensão. No problema em questão cada barra representa uma dimensão do problema.

Seguindo o proposto pelos autores em (Kennedy et al (1997)) foram realizadas alterações na biblioteca, na funcionalidade de atualização da posição da partícula. Esta não sendo mais realizada como a soma da posição atual com a velocidade. Com a alteração proposta, é gerada uma matriz cópia da matriz de velocidades, essa preenchida pelas velocidades originais, normalizadas pela função sigmoide, dessa forma a velocidade pode ser usada de forma a representar uma estatística. A atualização da posição passa a ser definida pela comparação entre a velocidade calculada na função sigmóide e um valor randômico entre [0.0,1.0]. É definindo 1 (Um) se o valor da sigmóide for superior ao valor

randômico, e 0 (Zero) caso contrario, conforme o apresentada na equação 2.3.

Na proposta em questão, somente um módulo da solução, desenvolvido no MATLAB, sofreu alterações, sendo mantido inalterada a aplicação FPOLINGO. Com a alteração proposta a matriz de posição do PSO passa a ser composta efetivamente por valores binários e não valores arredondados.

3.1.2 Nova implementação

Desenvolveu-se Outra versão do PSO Binário, proposta por (Khanesar et al (2007)) foi desenvolvida.

Seguindo os conceitos propostos, foi preparada uma aplicação semelhante a original, são inicialmente definidos os parâmetros a serem utilizados, sendo eles: número de barras, GDs a serem instaladas, o número de partículas, o número máximo de iterações, a velocidade máxima ($Vmax$) e a matriz MxN (Número partículas X Número de Barras) que representa a solução inicial para o problema. Além desses também é definida a velocidade inicial de cada partícula, assim como as respectivas velocidades V_0 e V_1 que indicam a probabilidade de uma partícula ter o bit alterado para 0 ou 1 respectivamente. Inicializa-se também as variáveis de melhor localização global ($Gbest$) e local ($Pbest$) das partículas, a partir da execução da função objetivo para a solução inicial gerada aleatoriamente.

Com base nos valores iniciais, são efetuadas as iterações do programa, sendo que cada iteração é dividida em 6 fases, sendo elas:

1. Calcular a inércia inicial, ou seja a variável que define o impacto do movimento anterior no movimento atual, com valor no intervalo [0.0,1.0], calculada de acordo com o número total de iteraçãos menos o número de iterações ja executadas, dividido pelo total de iterações definido na inicialização da aplicação;

2. Executar a função objetivo para calcular as perdas no sistema, com posterior atualização das variáveis $Pbest$ e $Gbest$;

3. Com base nos valores de Pbest e Gbest obtidos, definir os valores para o cálculo das velocidades V_0 e V_1 , como o descrito na equação 2.5;

4. Calcular as velocidades V_0 e V_1 para cada uma das partículas;

5. De acordo com o valor de x_i, definir qual das velocidades V_0 ou V_1 deve-se utilizar para atualizar a posição da partícula, sendo que se $x_i = 1$ é utilizado V_0, caso contrario, V_1;

6. De acordo como a velocidade definida, assim como no PSO Binário original, gera-se a matriz de velocidades processada na função sigmóide para posterior comparação com um valor randômico dentro do intervalo [0.0, 1.0]. Sendo tal comparação é feita segundo a regra: se o valor aleatório for menor que a velocidade, o bit tem seu valor alterado para o seu complemento (valor oposto), caso contrário mantém seu valor atual.

De forma semelhante ao algoritmo original, a cada iteração a solução proposta passa pelo processamento na aplicação FPOLINGO. Esse calcula a perda de energia dentro do sistema definido, permitindo identificar se a solução proposta pelo PSO gera soluções com qualidade superior à aplicação original

3.1.3 Heurísticas utilizadas

Ambas as propostas de PSO Binário trabalham com a representação de valores reais em binário. Com o intuito de trabalhar somente com os valores binário, foram realizados testes introduzindo heurísticas para garantir que não seja ultrapassado o número máximo de barras ativas, que correspondem ao total de GDs a serem instaladas. Todas as heurísticas aqui listadas foram aplicadas nas duas versões do algoritmo do PSO Binário implementadas.

C.1 - Adicionar à função objetivo (FOB) um valor previamente definido, quando existem mais barras ativas que o número de GDs determinado

A presente heurística, já existente na aplicação original definida em (Dias et al (2012)), foi a penalização da solução quando esta apresentar um número de barras ativas superior o número de GDs definido. Neste caso, é adicionado um número de unidades pré-fixado (10 unidades) ao resultado da função objetivo para essa partícula, de tal forma que penalize essa solução candidata.

C.2 - Adicionar à FOB um valor, esse proveniente da diferença entre o número extra de barras ativas, menos o número de GDs a serem instaladas, no caso em que existem mais barras ativas que o número de GDs

Esta heurística propõe uma penalização de forma semelhante a anterior, porém no lugar da adição de um valor pré-fixado, é adicionado um valor fixo, multiplicado pelo número total de barras excedentes.

C.3 - Remoção aleatória das barras ativas que superam o número de GDs esperados

A abordagem nessa heurística é desativar aleatoriamente todas as barras ativas que ultrapassem o número de GDs esperados.

C.4 - Limitar a ativação de barras, sendo essas limitadas a quantidade de GDs a serem instaladas

A abordagem proposta nessa heurística é limitar a cada iteração o número de barras que ficam ativas por partícula ao mesmo número de GDs a serem instaladas. Como descrito na literatura, a velocidade é usada como definição da probabilidade de um bit ter seu valor alterado para 1. Através desse critério, durante a atualização da posição das partículas, todos os seus bits são alterados para 0. De acordo com a matriz de velocidades para a dada partícula, são encontradas os N valores mais altos, sendo N o número de GDs a serem instaladas. O bit correspondente para cada um desses valores tem seu valor alterado para 1, definindo em quais barras devem ser instaladas as GDs.

4 Análise dos Resultados

4.1 Aplicação do PSO Binário ao problema de alocação ótima de GDs

Para a análise da execução das aplicações propostas neste trabalho, foi utilizado um problema comum da literatura, em que existem 69 barras distribuídas em ramos, com uma perda total de 225kW, (Abu-Mouti et al (2011)). A Figura 4.1, apresenta o problema em questão.

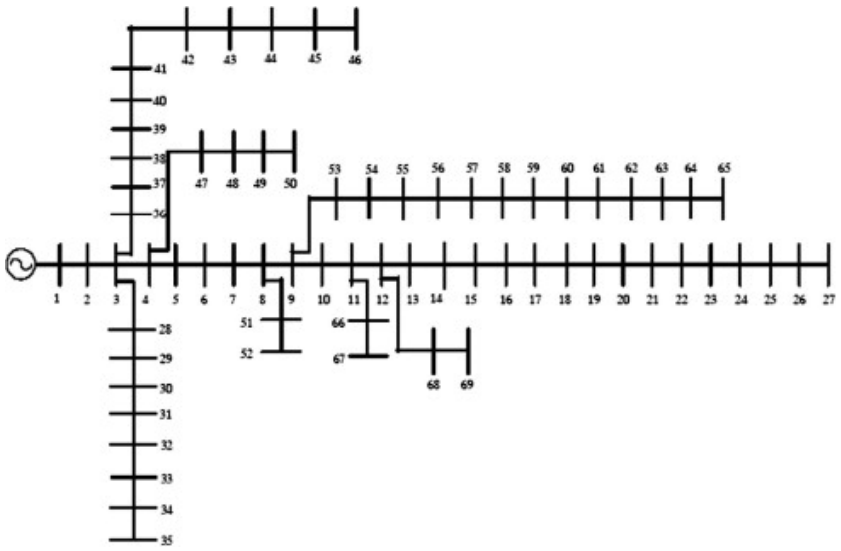

Figura 4.1: Sistema com 69 barras

A execução desse algoritmo foi realizada em um ambiente Intel Core 2 Duo, 2.53Mhz, 4GB de RAM, rodando o sistema operacional Windows, e as aplicações MATLAB R2008b e Lingo V11

4.1.1 PSO Original

A implementação do PSO original proposta em (Dias et al (2012)), teve como parâmetros de entrada os seguintes valores:

- Número de Partículas = 50

- C_1 e $C_2 = 2$
- A inércia variando de forma decrescente, limitada na implementação entre 0.9 e 0.4 (Dias et al (2012))
- Número de iterações = 50

a Tabela 4.1 apresenta os resultados obtidos em (Dias et al (2012)), para cada um dos números de GDs determinados são listadas as barras ativadas dentre as apresentadas na Figura 4.1, e a perda total dentro do sistema obtido:

Tabela 4.1: Resultados no PSO Original

Num. Barras	Barras Ativas	Perda (kW)
1	61	152
2	17/61	146,24
3	11/21/61	145,12

A aplicação apresentada em Abu-Mouti et al (2011)) obtêm os valores apresentados na Tabela 4.2, esses foram usados de base para comparação com os resultados em (Dias et al (2012)):

Tabela 4.2: Resultados no PSO Original

Num. Barras	Barras Ativas	Perda (kW)
1	56	155,29
2	53/56	149,63
3	56/61/33	148,31

4.1.2 Modificação sobre a aplicação original

Para a execução do algoritmo modificado, foram utilizados como base os mesmos parâmetros utilizados para a execução na aplicação original.

Execução baseada na heurística C.1

O algoritmo modificado se assemelha muito com a aplicação do PSO original, sendo alterado somente os trechos de código que estavam relacionados ao arredondamento da matriz de soluções para se comportarem como valores binários e o método de

atualização da posição das partículas. Como mencionado anteriormente, a heurística C.1 está presente na aplicação original sendo mantida para essa execução.

A execução não apresentou resultados satisfatórios, foram obtidos sistemas onde existiam um número de barras ativas superior ao número de GDs estabelecidos. A melhor solução encontrada em todas as tentativas de execução sempre foi a melhor dentre todas as presentes na matriz de soluções inicial, ou seja, após a primera iteração não foi possível obter um sistema que fosse melhor. A partir da segunda iteração o número de barras ativas sempre superava o número de GDs esperados, com isso o algoritmo não convergia para uma nova solução que atendesse a restrição do número minímo de GDs.

Mesmo com modificações nos parâmetros de entradas, aumentando ou diminuindo os valores de correspondentes ao número de partículas ou valor utilizado na penalização, não foi possível obter uma boa resposta, visto que o algoritmo não convergia para uma solução ótima, mantendo sempre com um número de barras ativas superior ao esperado.

Execução baseada na heurística C.2

A heurística C.2 se assemelha muito a C.1, mas no caso a penalização é feita pela multiplicação de um valor pré-definido pelo número de barras que excedem o número de GDs definido no problema. Dessa forma, sendo possível gerar valores diferentes de penalização na função objetivo de cada partícula.

Assim como os resultados obtidos em C.1, a presente heurística não apresentou resultados satisfatórios. Em todas as iterações foram efetuadas penalizações em todas as partículas. A penalização, mesmo quando aplicada, mantinha a geração de soluções não aplicáveis, com um número de barras superior ao número de GDs definido. Alterações de parâmetros, como aumentando ou diminuindo se seus valores, não forneceu ao algoritmo nenhuma alteração de resultado.

Execução baseada na heurística C.3

A execução do algoritmo com a heurística C.3, apresentou resultados que não violassem as restrições existentes. Essa heurística define que, se existir um número de barras ativas superior ao número de GDs definido, as barras que excedem o definido são

removidas de forma aleatória. Essa proposta não é ideal para um algoritmo de otimização, adiciona aleatoriedade na definição de soluções que violam a restrição. Os resultados foram listados na Tabela 4.3:

Tabela 4.3: Resultados da heurística C3 aplicada ao PSO Original Modificado

Num. Barras	Barras Ativas	Perda (kW)
1	32	224,98
2	15/35	182,28
3	12/24/68	279,74

Com a aplicação da heurística C.3, embora tenham sido obtidos resultados viáveis, a penalização foi aplicada para todas as partículas em todas as iterações, dessa forma demonstrando que a movimentação das partículas se mantem aleatória no espaço de soluções, não direcionando para um ponto ótimo. Na simulação com a heurística C.3, dentre quatro execuções, somente uma apresentou barras ativas que se aproximam dos resultados apresentados em (Dias et al (2012)).

Execução baseada na heurística C.4

A heurística C.4, define que será ativada somente as barras que apresentaram maior velocidade, consequentemente maior probabilidade do bit ter seu valor alterado para 1, sendo as demais barras mantidas inativas. Como o esperado, foram obtidos sistemas com o número de barras ativas igual ao número de GDs a serem instaladas. Dentre as simulações do algoritmo efetuadas, foi possível obter soluções próximas da melhor solução descrita em (Dias et al (2012)). Esses resultados são apresentads na Tabela 4.4:

Tabela 4.4: Resultados da heurística C4 aplicada ao PSO Original Modificado

Num. Barras	Barras Ativas	Perda (kW)
1	34	224,96
2	34/62	250,91
3	13/30/65	412,74

A proposta da heurística fornece resultados que atendem as restrições, porém as partículas mantêm uma movimentação aleatória no espaço de soluções, não convergindo para um ponto ótimo.

4.1.3 Nova Implementação

Para iniciar a aplicação do algoritmo na seção "Nova implementação" dentro de "Estudo de Caso" foram feitas execuções sem a aplicação de nenhuma heurística, baseado nos mesmos parâmetros usados na aplicação com o PSO original. De forma análoga a versão modificada, não foi possível obter uma solução viável, visto que o algoritmo não encontra uma resposta que atenda a restrição do problema, na qual o número de barras ativas é igual ao número de GDs definido.

No intuito de obter os resultados válidos, foram aplicadas as heurísticas propostas, sendo descritos os resultados de cada uma:

Execução baseada na heurística C.1

A aplicação da heurística C.1, que define a penalização das partículas com um valor pré-fixado, de forma semelhante a execução da versão modificada do PSO original não gerou resultados viáveis, as soluções obtidas violavam o número de GDs definido, apresentando barras ativas além desse valor. A penalização foi aplicada a todas as partículas, em todas as iterações da aplicação. Alterações nos parâmetros de entrada, aumentando ou diminuindo seus valores não geraram nenhuma alteração nos resultados.

Execução baseada na heurística C.2

A heurística proposta, que aplica a penalização através da multiplicação entre um valor pré-fixado e o número de barras excedentes, assim como a C.1, não gerou resultados satisfatórios. As soluções mantiveram o número de barras ativas superiores ao número de GDs definidas. A penalização proposta na heurística foi aplicada em todas as execuções em todas as partículas, demonstrando que a mesma não gera alterações na execução do algoritmo.

Execução baseada na heurística C.3

A heurística C.3, que propõe a remoção aleatória das barras ativas que ultrapassam o número de GDs definido, foi aplicada sendo possível obter soluções viáveis, como pode ser visto na Tabela 4.5:

Tabela 4.5: Resultados da heurística C3 aplicada a nova implementação

Num. Barras	Barras Ativas	Perda (kW)
1	12	175,99
2	44/63	182,28
3	2/14/55	115,91

Da mesma forma como já descrito, essa heurística mantêm a execução do algoritmo aleatória, de forma que não é possível determinar se em outra execução, com os mesmos parâmetros, serão obtidas boas soluções.

Execução baseada na heurística C.4

A heurística C.4, propõe que são ativadas somente as barras que apresentaram maior probabilidade do bit ter seu valor alterado para 1, sendo as demais inativada. A proposta fornece soluções viáveis, não atingindo o melhor valor encontrado em (Dias et al (2012)). Os valores obtidos são apresentados na Tabela 4.6:

Tabela 4.6: Resultados da heurística C4 aplicada a nova implementação

Num. Barras	Barras Ativas	Perda (kW)
1	58	517,16
2	57/61	219,03
3	23/53/ 62	747,22

Da mesma forma como já descrito, essa heurística mantêm a execução do algoritmo aleatória, de forma que não é possível determinar se em outra execução, com os mesmos parâmetros, serão obtidas boas soluções.

Sobre os resultados obtidos

Verificando os motivos da não obtenção de soluções viáveis, pelo fato do algoritmo não convergir para o ponto de boa solução para o problema, foi constatado que como a execução das aplicações binárias é descontínua, as partículas têm suas posições definidas de forma aleatória. Outro ponto analisado foi que ambos os trabalhos apresentados em (Kennedy et al (1997)) e (Khanesar et al (2007)) os autores trabalham com valores reais em sua representação binária, aplicados a funções hiperbólicas, não sendo aplicáveis para o uso de somente um bit na respresentação em cada dimensão.

5 Conclusão

Como foi apresentado, os algoritmos do PSO Binário propostos não forneceram resultados satisfatórios para o problema da alocação ótima de GDs. Mesmo com a aplicação de heurísticas para tratar as restrições, os resultados obtidos não superam o já proposto com a aplicação do PSO original (Dias et al (2012)). Foi constatado que, as propostas binárias estudadas se baseiam em representações de valores reais em cadeias binárias, não sendo possável trabalhar com um único bit para definir se uma barra está ou não ativa.

Avaliando o motivo dos resultados obtidos não superarem os estudos existentes, foi constatado que o algoritmo não convergia para uma boa solução quando manipulados somente valores binários, não trabalhando com representações. Com uma análise dos problemas encontrados, verificou-se que a descontinuidade das operações binárias envolvidas nos estudos realizados, não permitiram encontrar um método para manipular os dados a fim de obter a solução ótima.

A aplicação de outras váriações do algoritmo do PSO à problemas semelhantes ao de alocação otima de GDs ja foram propostos, como o apresentado em (del Valle et al (2008)). Não foram obtidas referências de trabalhos utilizando PSO binário para a alocação de GDs.

Devido a limitações de tempo para a excução desse trabalho e o tempo despendido na análise do problema de conversão do PSO Binário, não foi possivel avaliar outras variações do PSO aplicado à alocação ótima de GDs, sendo esse trabalho restrito somente a análise da versão Binária.

Como sugestão para trabalhos futuros fica a análise de outras variações do PSO para o estudo relacionado com GDs, destacamos,em especial, o Volitive PSO, uma combinação do PSO com o FSS(*Fish Search Scholl*). O FSS outro algoritmo bio-inspirado, baseado no comportamento do cardume de peixes, que possui um operador que pode ser adequado para evitar que a busca fique limitada a um mínimo local, de acordo com as respostas obtidas, esse operador específico aumenta ou diminui o raio campo de busca a ser aplicado. Outra sugestão é a modificação da aplicação FPOLINGO para permitir

receber como entrada valores reais, para assim os algoritmos binários trabalharem como o proposto em (Kennedy et al (1997)) e (Khanesar et al (2007)), manipulando valores reais em representações binárias, no lugar de receber diretamente valores binários, sendo assim possível a análise para a aplicação das variações binárias do PSO avaliadas neste trabalho.

Referências Bibliográficas

Abu-Mouti, F.; El-Hawary, M. **Optimal dg placement for minimizing power loss in distribution feeder systems using sensory-deprived optimization algorithm**. In: Electrical and Computer Engineering (CCECE), 2011 24th Canadian Conference, p. 000205–000209. IEEE, 2011.

Ackermann, T.; Andersson, G. ; Söder, L. Distributed generation: a definition. **Electric Power Systems Research**, v.57, n.3, p. 195–204, 2001.

Dias, B.; Oliveira, L.; Gomes, F.; Silva, I. ; Oliveira, E. **Hybrid heuristic optimization approach for optimal distributed generation placement and sizing**. In: Power and Energy Society General Meeting, 2012 IEEE, p. 1–6. IEEE, 2012.

Eberhart, R. C.; Shi, Y. ; Kennedy, J. **Swarm intelligence**. Morgan Kaufmann, 2001.

Kennedy, J.; Eberhart, R. **Particle swarm optimization**. In: Neural Networks, 1995. Proceedings., IEEE International Conference, volume 4, p. 1942–1948. IEEE, 1995.

Kennedy, J.; Eberhart, R. C. **A discrete binary version of the particle swarm algorithm**. In: Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference, volume 5, p. 4104–4108. IEEE, 1997.

Mojtaba Ahmadieh, K.; Teshnehlab, M. ; Shoorehdeli, M. A. **A novel binary particle swarm optimization**. In: Control & Automation, 2007. MED'07. Mediterranean Conference, p. 1–6. IEEE, 2007.

Krueasuk, W.; Ongsakul, W. Optimal placement of distributed generation using particle swarm optimization. **M. Tech Thesis, AIT, Thailand**, 2006.

Liu, W.; Liu, L.; Cartes, D. A. ; Venayagamoorthy, G. K. Binary particle swarm optimization based defensive islanding of large scale power systems. **Technomathematics Research Foundation, international Journal of Computer Science & Applications**, v.4, n.3, p. 69–83, 2007.

del Valle, Y.; Venayagamoorthy, G. K.; Mohagheghi, S.; Hernandez, J.-C. ; Harley, R. G. Particle swarm optimization: basic concepts, variants and applications in power systems. **Evolutionary Computation, IEEE Transactions**, v.12, n.2, p. 171–195, 2008.