

SISTEMA MULTIAGENTE APLICADO AO GERENCIAMENTO TOPOLÓGICO DE SISTEMAS ELÉTRICOS DE DISTRIBUIÇÃO INTELIGENTES

FILIFE DE OLIVEIRA SARAIVA, EDUARDO NOBUHIRO ASADA

*Laboratório de Análise de Sistemas de Energia Elétrica
Escola de Engenharia de São Carlos
Universidade de São Paulo
São Carlos, São Paulo, Brasil*

Emails: filipe.saraiva@usp.br, eeasada@usp.br

Abstract— Smart grids are considered as the new generation of electric power systems, which combine advances in computing, distributed systems and artificial intelligence to provide more functionality to the power grid. This approach changes significantly the way the planning and operation of distribution systems are carried out. With that environment in mind, this paper proposes an approach with the use of multi-agent systems to investigate the operation of the distribution system, by means of the reconfiguration of grid topology to satisfy the operation constraints according to the data processed by agents dispersed in the grid.

Keywords— Smart Grids, Smart Distribution Systems, Multi-agent Systems

Resumo— Os *smart grids* são tidos como a nova geração dos sistemas elétricos de potência, combinando avanços em computação, sistemas distribuídos e inteligência artificial para prover maiores funcionalidades a rede elétrica. Esta abordagem altera fundamentalmente a maneira como se dá o planejamento e a operação de sistemas de distribuição. Com esse cenário em vista, o presente trabalho trata da implementação de um sistema multiagente para simulação computacional de sistemas elétricos de distribuição, buscando executar a reconfiguração topológica do sistema para satisfazer as condições de operação a partir de processamento de dados capturados de forma distribuída pelos agentes dispersos na rede elétrica.

Palavras-chave— Smart Grids, Sistemas de Distribuição Inteligentes, Sistemas Multiagentes

1 Introdução

Há uma grande comoção e expectativa diante das promessas de funcionalidades a serem implementadas nos sistemas elétricos de potência do futuro: auto-recuperação do sistema, utilização em larga escala de geração distribuída, controle e acompanhamento pormenorizado do consumo do usuário, redução de perdas, e muitas outras (Wei et al., 2009). Entretanto, em termos de desenvolvimento tecnológico, estes sistemas conhecidos mais comumente por *smart grids* representam uma série de desafios a serem superados até seu uso em ambiente de produção.

A principal característica de um *smart grid* é a capacidade que os equipamentos alocados ao sistema tem de comunicarem-se entre si (Brown, 2008). As informações daí obtidas permitirão que estes equipamentos iniciem um processo de tomada de decisão afim de atingir algum objetivo - que seriam uma ou mais funcionalidades do sistema elétrico, como as citadas anteriormente.

Logo, a utilização de ferramentas distribuídas para simulação computacional de algoritmos próprios para as funcionalidades das redes elétricas inteligentes é fundamental para garantir um maior nível de fidelidade do ambiente alvo. Nesse quesito, o uso de sistemas multiagentes torna-se uma abordagem interessante para trabalhos desse tipo.

Este artigo trata do desenvolvimento e aplicação de um método computacional distribuído, utilizando sistemas multiagentes, para o gerencia-

mento topológico de sistemas de distribuição inteligentes - que seriam os *smart grids* aplicados a sistemas elétricos de distribuição. O objetivo desta aplicação é gerenciar a topologia do sistema, encontrando uma configuração de chaveamentos que garanta o ponto de operação do sistema elétrico dado um conjunto de restrições.

Este trabalho está estruturado da seguinte maneira: a Seção 2 conceitua o que seria o gerenciamento topológico no contexto de *smart grids* para distribuição de energia. A Seção 3 descreve o que são sistemas multiagentes e sua característica distribuída. A Seção 4 apresenta a proposta de algoritmo distribuído e a modelagem do sistema multiagente para o gerenciamento topológico. Por fim, a Seção 5 comenta sobre a implementação computacional e os testes executados e, em seguida, a Seção 6 conclui o artigo com algumas observações e perspectivas para trabalhos futuros no tema.

2 Gerenciamento Topológico de Sistemas Elétricos de Distribuição Inteligentes

A partir do momento em que, na etapa de geração, um tipo de energia é transformada em energia elétrica, a corrente flui pelos equipamentos e cabos de transmissão ou distribuição, em direção aos centros consumidores ou às cargas do sistema elétrico.

As redes elétricas do sistema de distribuição são, em sua maioria, configuradas para operarem

em uma topologia radial, onde cada carga alocada recebe energia de apenas um ramo do sistema. Esse tipo de topologia facilita fatores relacionados à proteção, coordenação e atenuação de correntes de curto-circuito além de diminuir custos com equipamentos (Mantovani et al., 2000).

Entretanto, apesar das facilidades, a topologia radial também pode ocasionar problemas em situações de contingência ou sobrecarga, nas quais as vezes se faz necessário o corte de cargas, ilhamentos de algumas partes do sistema elétrico, entre outras (Guerra et al., 2011).

O gerenciamento topológico do sistema significa a busca por novas topologias radiais no sistema de distribuição, visando atender algum objetivo específico, normalmente relacionado a aspectos sobre otimização, controle ou gerenciamento do estado das redes elétricas.

No contexto dos sistemas elétricos de distribuição inteligentes, imagina-se que a possibilidade de aferição do estado do sistema, de forma distribuída, poderá possibilitar subsídios necessários de informação para a execução de ações específicas a alguma funcionalidade do sistema elétrico. Por exemplo, o sistema elétrico poderia decidir qual o melhor momento para usar uma fonte de geração distribuída; a topologia do sistema poderia ser chaveada para rotear o fluxo de corrente afim de não utilizar um trecho defeituoso ou adequar-se a um novo perfil de carga em dado momento, reduzindo as perdas para aquele estado; entre outros. Todas estas funcionalidades são executadas a partir de ações de gerenciamento topológico do sistema.

A próxima seção descreve a ferramenta computacional utilizada no algoritmo que propomos para o gerenciamento topológico de sistemas elétricos de distribuição inteligentes, os sistemas multiagentes.

3 Sistemas Multiagentes

Sistemas multiagentes são trabalhados como ferramentas de inteligência artificial distribuída que, diferentemente da inteligência artificial “clássica”, embasa suas pesquisas na possibilidade de aprendizado e resolução de problemas a partir de um fenômeno social (Wooldridge, 2009).

Agentes são entidades computacionais autônomas, que percebem o ambiente em que estão inseridos e agem sobre este mesmo ambiente tentando atingir seus objetivos ou reproduzir comportamentos (Shoham and Leyton-Brown, 2009). Sistemas multiagentes são formados por um conjunto de agentes, que devem ser modelados de forma que o encadeamento de suas ações alcance o objetivo previamente modelado.

Entre outras características, agentes de um sistemas multiagentes são restritos em suas capacidades individuais, tanto em nível de percepção

quanto de atuação sobre o ambiente, por conta de sua característica distribuída. O controle em sistemas multiagentes é distribuído, os dados são descentralizados e a computação das informações é assíncrona.

Os principais desafios do desenvolvimento de sistemas multiagentes reside na dificuldade em se modelar de que maneira a dinâmica da sociedade de agentes resultará no cumprimento do objetivo proposto. Para isso, é necessário a determinação de quais tipos de ações e quais agentes irão fazê-las, como será realizada a comunicação entre os agentes, quais protocolos serão utilizados, entre outras (Saraiva et al., 2011).

A característica inerentemente distribuída e o uso de inteligência artificial tornam os sistemas multiagentes interessantes candidatos para o uso em simulações de sistemas elétricos de potência inteligentes, os *smart grids*. A literatura específica desta tecnologia aplicada a área de sistemas de potência já conta com interessantes estudos. Em (McArthur et al., 2007a) e (McArthur et al., 2007b) encontra-se uma revisão sobre o tema, e em (Rumley et al., 2008) tem-se um dos primeiros usos em simulações relacionadas com *smart grids*.

A próxima sessão descreverá o algoritmo proposto e a modelagem do sistema multiagente aplicado ao gerenciamento topológico de sistemas de distribuição inteligentes.

4 Algoritmo e Modelagem do Sistema Multiagente

O sistema proposto define 3 tipos de agentes: os *feederAgents*, responsáveis por monitorarem as subestações do sistema; os *switcherAgents*, que verificam as chaves do sistema e seus estados; e os *loaderAgents*, responsáveis por verificar as demandas das cargas do sistema. Esta seção descreve os agentes, seus comportamentos, e alguns conceitos adicionais utilizados.

Foram também definidos dois conceitos que auxiliam no processo de tomada de decisão e cálculo do sistema. O primeiro, Coalizão, refere-se ao conjunto de agentes em cargas delimitados pelo mesmo conjunto de chaves. A a Figura 1 apresenta apenas o subgrafo que representa a coalizão composta pelos agentes nas cargas 3, 4, 5 e 6.

Caminho Elétrico representa o menor caminho, em termos apenas da resistência elétrica, entre uma determinada subestação e uma coalizão. Para ilustrar, a Figura 2 apresenta o caminho elétrico entre a subestação 1 e a coalizão apresentada na Figura 1.

Agora, será detalhado o *feederAgent*, responsável por monitorar as subestações do sistema. No momento de sua execução, um *feederAgent*, primeiramente, envia mensagens aos *switcherAgents* perguntando qual o estado em que as chaves se

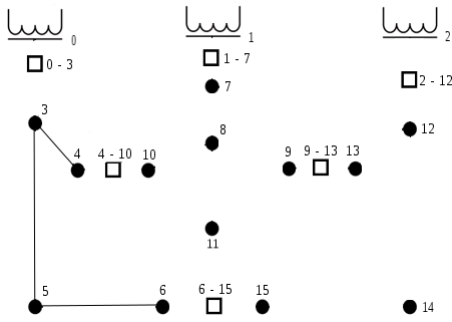


Figura 1: Subgrafo que forma a coalizão dos agentes nas cargas 3, 4, 5 e 6

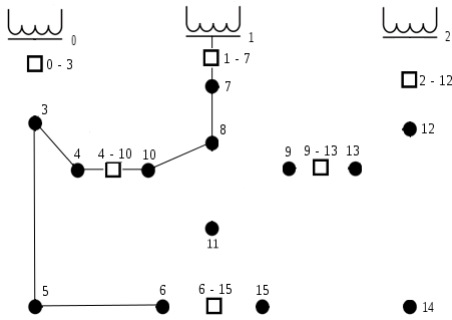


Figura 2: Subgrafo que forma o caminho elétrico para coalizão da Figura 1

encontram, colhendo a resposta na sequência. Em seguida, o *feederAgent* fica na espera do envio de mensagens dos *loaderAgents* líderes de coalizões. Estas mensagens dizem respeito à coalizão liderada e ao caminho elétrico entre o *feederAgent* e a coalizão.

Enquanto o *feederAgent* recebe mensagens sobre coalizões e caminhos elétricos, ele vai “empilhando” os respectivos subgrafos que estão contidos nas mensagens em uma representação de grafo que combina todas estas informações. Ao final, tem-se tanto uma representação radial do sistema elétrico quanto informações sobre demanda dos agentes. Estas informações permitirão ao *feederAgent* ter uma ideia do estado do sistema de distribuição naquele momento e, a partir delas, realizar suas operações.

Com estas informações obtidas, o *feederAgent* procede a execução de um algoritmo de fluxo de carga, utilizando neste trabalho o método da varredura (Kersting, 2006). Isto permitirá ao *feederAgent* verificar se a topologia construída é viável e, também, definir quais as perdas elétricas para cada coalizão do sistema.

Caso a topologia seja inviável, o *feederAgent* faz um procedimento de retirar do cálculo a coalizão com maior caminho elétrico, em termos de somatória da resistência. Isso torna o sistema a ser trabalhado menor, tanto em tamanho quanto

em demanda total, e permite ao *feederAgent* re-executar o fluxo de carga.

Após o método convergir, o *feederAgent* calcula as perdas elétricas para cada coalizão, medidas por $\sum_{i \in C} r_i I_i^2$, onde i é um trecho qualquer que pertence ao conjunto de trechos da coalizão e seu respectivo caminho elétrico, identificado por C ; r_i e I_i são respectivamente a resistência e a corrente encontrada no trecho i .

Em seguida, o *feederAgent* envia mensagens para os líderes das coalizões com o valor calculado de suas perdas elétricas. Após a tomada de decisão dos *loaderAgents*, os *feederAgents* recebem mensagens avisando qual deles foi escolhido para prover energia àquela coalizão. Com esta informação, o *feederAgent* verifica quais *switcherAgents* devem estar fechados para que a energia que sai da subestação por ele monitorada chegue às coalizões ativas. Então, o *feederAgent* envia mensagens para estes *switcherAgents* pedindo que fechem o circuito. Com todas estas ações terminadas, o *feederAgent* termina sua execução.

Tratar-se-á agora do processamento realizado pelos agentes localizados nas cargas, os *loaderAgents*. Quando os *loaderAgents* são inicializados, eles recebem o valor da carga a qual estão monitorando. Em seguida, executam um algoritmo de busca em profundidade (Cormen et al., 2009) modificado, a qual coalizão eles pertencem e qual o líder desta coalizão. O Algoritmo 1 apresenta a busca empregada.

Algoritmo 1: Busca em profundidade - en-
contra coalizão

Entrada: G, v, Sw, Z

```

1: para todo  $z \in \text{Vizinhos}(v)$  faça
2:    $z = \text{Seleciona-Vizinho}(v)$ 
3:   if ( $z \notin Z$ ) and ( $(v, z) \notin Sw$ ) then
4:      $Z = Z \cup \{z\}$ 
5:      $Z = \text{Busca-Profundidade}(G, z, Sw, Z)$ 
6:   fim if
7: fim para
8: retorne  $Z$ 
    
```

Dado um grafo G , um nó v que representa a posição do *loaderAgent*, a definição das arestas que contém chaves representadas por Sw e o conjunto com os nós da coalizão inicialmente vazio Z , o Algoritmo 1 inicia varrendo os vizinhos de v , conforme a linha 1.

Com um vizinho selecionado, a condição *se* da linha 3 verifica, primeiramente, se z já foi avaliado. Caso não, verifica se a aresta (v, z) contém uma chave. Caso não, o vértice z é marcado como fazendo parte da coalizão, e agora ele terá a sua vizinhança avaliada, sendo passado recursivamente para a mesma função.

Ao final da execução do Algoritmo 1, Z será o conjunto de vértices do subgrafo do sistema de distribuição, representado pelo grafo completo, que

contém a coalizão liderada pelo agente *loaderAgent*.

A seleção do líder da coalizão é bem simples: cada *loaderAgent* recebe um nome que inicia com *loaderAgent* seguido de um número de identificação. Após o algoritmo de busca em profundidade listar todos os agentes de uma coalizão, o agente que possuir o maior número de identificação será o líder do grupo.

Após a busca em profundidade, os agentes já conseguiram construir o subgrafo que indica a localização da coalizão. Os *loaderAgents* da coalizão que não são o líder, neste momento, enviam para o líder suas respectivas demandas. Após receber todas as demandas, o agente líder envia os dados da coalizão para todos os *feedergents* do sistema.

Em seguida, o *loaderAgent* líder executa um algoritmo de busca no grafo para encontrar o menor caminho, em termos de resistência elétrica, entre cada *feederAgent* do sistema e algum agente que pertença a sua coalizão. Para tanto, é utilizado um algoritmo de Dijkstra modificado (Saraiva et al., 2011), onde existem vários pontos finais e o objetivo é que o algoritmo encontre o primeiro destes pontos. O Algoritmo 2 apresenta esta versão do algoritmo de Dijkstra.

Algoritmo 2: Algoritmo de Dijkstra com múltiplos pontos finais

Entrada: G, v_i

- 1: $Q_i = \text{Inicializar-Custos-Relativos}(G), \forall i$
- 2: $A_i = \text{Inicializar-no-Anterior}(), \forall i$
- 3: $F = \text{Define-Pontos-Finais}()$
- 4: $S = \emptyset$
- 5: $R = \emptyset$
- 6: $D = \emptyset$
- 7: $temp = 0$
- 8: $noAvaliado = \text{No-Agent-Feeder-Avaliado}()$
- 9: **enquanto** $noAvaliado \notin F$ **faça**
- 10: $Q_i = \text{Calc-Custo-Relativo}(G, noAvaliado), \forall i$
- 11: $S = S \cup noAvaliado$
- 12: $temp = noAvaliado$
- 13: $noAvaliado = \text{Minimo}((V - S), Q)$
- 14: $A_{noAvaliado} = temp$
- 15: **fim enquanto**
- 16: **para todo** $v \in V$ **faça**
- 17: $D = D \cup (v, A_v)$
- 18: **fim para**
- 19: $R = (V, D)$
- 20: **retorne** R

No Algoritmo 2, dado o grafo $G = \{V, E\}$ onde V é o conjunto dos vértices e E o conjunto das arestas; v_i será o vértice-fonte, que será a barra onde se encontra o *loaderAgent* líder. Na linha 3, a chamada para a função *Define-Pontos-Finais()* retorna o conjunto de pontos finais do algoritmo, ao qual, quando encontrar qualquer de-

les, terá seu objetivo atingido - representando a condição do laço *enquanto* da linha 9.

Com os caminhos elétricos calculados, o líder envia os dados de cada caminho ao *feederAgent* relacionado. Este é o momento em que os *feederAgents* executam o algoritmo de fluxo de carga, como descrito anteriormente.

A próxima ação executada pelos *loaderAgents* é receber o valor das perdas elétricas enviadas por cada *feederAgent* e selecionar qual deles será o *feederAgent* ativo. Essa seleção é feita a partir do *feederAgent* que indica a menor perda para a coalizão. Apesar do critério de escolha, isso não garante que a configuração do sistema elétrico, globalmente, terá a menor perda possível. Esse critério apenas tornará a topologia final radial.

O último dos agentes que compõem a modelagem aqui apresentada, chama-se *switcherAgent*, que é puramente reativo (Wooldridge, 2009) e também o mais simples.

Basicamente, o *switcherAgent* apenas “reage” a mensagens que são enviadas pelos *feederAgents*: ele informa seu estado atual da chave que monitora, ou a altera.

Do exposto, o próximo passo da pesquisa foi implementar a modelagem e aplicá-la a alguns exemplos teóricos de sistemas de distribuição para avaliar se os agentes conseguem gerenciar a topologia da rede.

5 Implementação e Simulações Computacionais

A implementação foi realizada utilizando a linguagem de programação Java, com o *framework* de programação multiagente JADE (Bellifemine et al., 2007). A solução computacional foi executada em um Linux 32bits com máquina virtual Java OpenJDK, em um computador Intel Core i5 com 4 GB de RAM.

Foram selecionados alguns exemplos teóricos de sistemas de distribuição para verificar a aplicabilidade e comportamento do algoritmo proposto. Em todos os sistemas a tensão nominal dos alimentadores é igual a 23 kV, a tolerância utilizada no método da varredura é igual ao produto da tensão nominal por 10^{-3} e o número máximo de iterações executadas é igual a 100. Nas tabelas subsequentes, P significa a potência ativa, medida em kW; Q a potência reativa, medida em kvar; as tensões são dadas em kV e as correntes em A.

Sistema com 10 barras

O primeiro exemplo, apresentado graficamente na Figura 3 e cujos dados se encontram na Tabela 1, testa a comunicação entre os agentes e a respectiva tomada de decisão da forma mais simples possível. Trata-se de um sistema com apenas um ramo e sem chaves, com apenas uma subestação.

O objetivo é verificar se a comunicação entre os agentes, de fato, funcionará. Que esta comunicação permitirá que apareça uma liderança nos *loaderAgents* e, por consequência, uma coalizão; que essa liderança consiga executar os cálculos de caminho mínimo e reunir as mensagens sobre as demandas nos demais agentes; que o agente *feederAgent* fará o cálculo de fluxo de carga a partir das informações enviadas pelo líder da coalizão; que o *feederAgent* realize a proposta para o líder e, ao final, que o líder aceite esta proposta e eleja o *feederAgent* como o alimentador ativo para aquela coalizão.

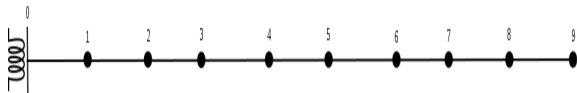


Figura 3: Sistema de testes - 10 barras

Tabela 1: Sistema de 10 barras

Sistema 10 barras		
Carga	P (kW)	Q (kvar)
1	1840	460
2	980	340
3	1790	446
4	1598	1840
5	1610	600
6	780	110
7	1150	60
8	980	130
9	1640	200
Trechos	Resistência (Ω)	Reatância (Ω)
0-1	0,1233	0,4127
1-2	0,0140	0,6051
2-3	0,7463	1,2050
3-4	0,6984	0,6084
4-5	1,9831	1,7276
5-6	0,9053	0,7886
6-7	2,0552	1,1640
7-8	4,7953	2,7160
8-9	5,3434	3,0264

Este sistema tem apenas uma coalizão, que será liderada pelo *loaderAgent* 9. Após descobrir-se líder, este agente recebe mensagens com os valores das demandas dos demais *loaderAgents*, formando a coalizão. Ele então envia estes dados para o único *feederAgent* do sistema. Em seguida, o líder executa o algoritmo de Dijkstra modificado para encontrar o menor caminho entre o *feederAgent* e a coalizão. Encontrando (este caminho é apenas o trecho 0 - 1), ele envia esta informação ao *feederAgent*.

Com todos estes dados obtidos, o *feederAgent* executa o algoritmo de fluxo de carga (método da varredura) e encontra uma solução em 12 iterações, indicando que a topologia é viável. Quando o *feederAgent* envia a resposta para o líder da coalizão, este seleciona-o como ativo para seu grupo.

Tabela 2: Resultado final para o sistema de 10 barras

Tensões Finais (kV) - Sistema de 10 barras	
Barra	Tensão
0	23,0000
1	22,8357
2	22,7041
3	22,1400
4	21,7810
5	21,0504
6	20,8105
7	20,3796
8	19,6622
9	19,1574
Número de iterações: 12	

Sistema com 11 barras e 2 chaves

O próximo exemplo, cujo desenho esquemático pode ser visto na Figura 4 e os dados encontram-se na Tabela 3, é o exemplo anterior da Figura 3 com a adição de mais uma subestação e duas chaves, uma entre as barras 0 e 2 e outra entre as barras 1 e 2.

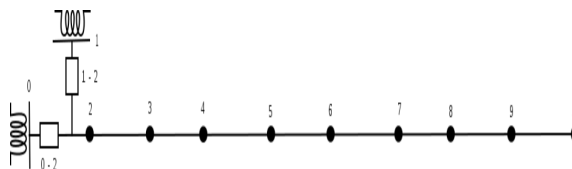


Figura 4: Sistema de testes - 11 barras e 2 chaves

Tabela 3: Sistema de 11 barras e 2 chaves

Sistema 11 barras 2 chaves		
Carga	P (kW)	Q (kvar)
2	1840	460
3	980	340
4	1790	446
5	1598	1840
6	1610	600
7	780	110
8	1150	60
9	980	130
10	1640	200
Trechos	Resistência (Ω)	Reatância (Ω)
0-2	0,1233	0,4127
1-2	0,0123	0,0412
2-3	0,0140	0,6051
3-4	0,7463	1,2050
4-5	0,6984	0,6084
5-6	1,9831	1,7276
6-7	0,9053	0,7886
7-8	2,0552	1,1640
8-9	4,7953	2,7160
9-10	5,3434	3,0264
Chaves: Trechos 0 - 2 e 1 - 2		

A finalidade deste exemplo é, além de atingir os objetivos já listados no primeiro exemplo, verificar se a única coalizão do sistema escolherá apenas um alimentador, e que essa escolha será

pelo alimentador que conduz a uma menor perda - por conta da diferença entre as resistências nos trechos 0 - 2 e 1 - 2. A opção escolhida também implicará em uma configuração específica para os *switcherAgents* do sistema.

Após as trocas de mensagens entre os *loaderAgents* e o envio de mensagens sobre coalizão e caminho elétrico para os *feederAgents*, a execução do fluxo de carga chega a resultados diferentes: o *feederAgent* 0 encontra uma perda de 78,3778 kW e o *feederAgent* 1 uma perda de 72,9378 kW.

Quando os *feederAgents* enviam mensagem para o líder dos *loaderAgents*, ele seleciona o *feederAgent* 1 como alimentador ativo da coalizão. Então, o *feederAgent* 1 envia uma mensagem para o *switcherAgent* em 1 - 2 requisitando que ele assuma o estado fechado, permitindo então que a energia percorra a rede elétrica e alimente as cargas do sistema.

A Tabela 4 apresenta o resultado para os dois *feederAgents*; entretanto, o *feederAgent* escolhido é apenas o 1. Na tabela, A_0 representa a subestação monitorada pelo *feederAgent* 0, e A_1 a subestação monitorada pelo *feederAgent* 1.

Tabela 4: Resultado final para o sistema de 11 barras e 2 chaves

Tensões Finais (kV)		
Barra	A_0	A_1
0	23,0000	-
1	-	23,0000
2	22,8357	22,9839
3	22,7041	22,8559
4	22,1400	22,2998
5	21,7810	21,9447
6	21,0504	21,2235
7	20,8105	20,9869
8	20,3796	20,5612
9	19,6622	19,8523
10	19,1574	19,3536
Iterações	12	12
Perdas	78,3778 kW	72,9378 kW

Sistema com 11 barras e 3 chaves

O terceiro exemplo, representado na Figura 5 e na Tabela 5, é também composto por duas subestações, mas cada uma está em uma extremidade do sistema. Há também uma chave que divide o sistema elétrico entre as barras 5 e 6.



Figura 5: Sistema de testes - 11 barras e 3 chaves

O comportamento a ser verificado neste exemplo, em adição aos anteriores, é observar se, de

Tabela 5: Sistema de 11 barras e 3 chaves

Sistema 11 barras 3 chaves		
Carga	P (kW)	Q (kvar)
2	1840	460
3	980	340
4	1790	446
5	1598	1840
6	1610	600
7	780	110
8	1150	60
9	980	130
10	1640	200
Trechos	Resistência (Ω)	Reatância (Ω)
0-2	0,1233	0,4127
2-3	0,0140	0,6051
3-4	0,7463	1,2050
4-5	0,6984	0,6084
5-6	∞	∞
7-6	1,9831	1,7276
8-7	0,9053	0,7886
9-8	2,0552	1,1640
10-9	4,7953	2,7160
1-10	5,3434	3,0264
Chaves: Trechos 0 - 2, 5 - 6 e 1 - 10		

fato, serão formadas duas coalizões e se estas coalizões ativarão, cada uma, o alimentador que lhe está mais próximo. O trecho 5 - 6, monitorado por um *switcherAgent*, tem uma resistência muito alta. Isso implicará que os *feederAgents* não conseguirão convergir o método de cálculo do fluxo de carga para uma topologia onde cada um tenta alimentar as duas coalizões juntas.

Assim, os *feederAgents* iniciarão o método de retirar as coalizões que tem o caminho elétrico maior e, sem estas, executarão novamente o algoritmo de fluxo de carga tentando verificar se a nova topologia é viável.

De fato, este comportamento pôde ser observado. Na Tabela 6 pode-se perceber que cada *feederAgent* está alimentando a coalizão que lhe é mais próxima. Isso implica uma configuração de chaves com os trechos 0 - 2 e 1 - 10 configuradas como fechadas e a chave no trecho 5 - 6 aberta.

Tabela 6: Resultado final para o sistema de 11 barras e 3 chaves

Tensões Finais (kV)		
Barra	A_0	A_1
0	23,0000	-
1	-	23,0000
2	22,9090	-
3	22,8344	-
4	22,5986	-
5	22,4994	-
6	-	19,0806
7	-	19,3064
8	-	19,8824
9	-	21,1452
10	-	21,7890
Iterações	5	12
Perdas	44,9422 kW	904,7419 kW

Sistema com 16 barras e 6 chaves

O próximo e último exemplo é baseado no clássico sistema elétrico de Cinvalar de 16 barras (Civanlar et al., 1988), mas foi modificado para ter apenas 6 chaves. A Figura 6 e a Tabela 7 trazem, respectivamente, o desenho e o conjunto de dados desta instância. Outra diferença em relação ao sistema original foi a adição de limites máximos de correntes no trechos, cujos valores são dados na Tabela 8. A adição desta restrição visa apresentar a versatilidade do método aqui trabalhado.

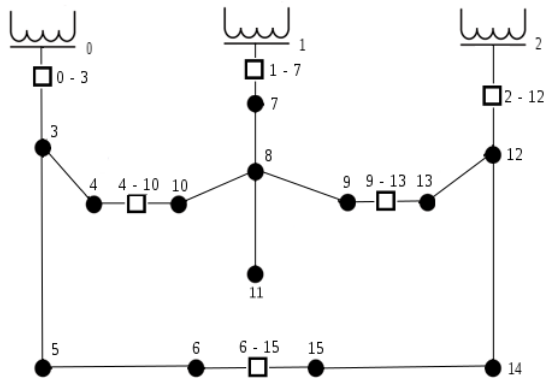


Figura 6: Sistema de testes - 16 barras e 6 chaves

Tabela 7: Sistema de 16 barras e 6 chaves

Sistema 16 barras 6 chaves		
Carga	P (kW)	Q (kvar)
3	2,0	1,6
4	3,0	1,5
5	2,0	0,8
6	1,5	1,2
7	4,0	2,7
8	5,0	3,0
9	1,0	0,9
10	0,6	0,1
11	4,5	2,0
12	1,0	0,9
13	1,0	0,7
14	1,0	0,9
15	2,1	1,0
Trechos	Resistência (Ω)	Reatância (Ω)
0-3	0,3967	0,5290
3-4	0,4232	0,5819
3-5	0,4761	0,9522
5-6	0,2116	0,2116
1-7	0,5819	0,5819
7-8	0,4232	0,5819
7-9	0,5819	0,5819
8-10	0,5819	0,5819
8-11	0,4232	0,5819
2-12	0,5819	0,5819
12-13	0,4761	0,6348
12-14	0,4232	0,5819
14-15	0,2116	0,2116
4-10	0,2116	0,2116
9-13	0,2116	0,2116
6-15	0,6348	0,6348
Chaves: 0-3, 1-7, 2-12, 4-10, 6-15, 9-13		

Tabela 8: Limite máximo de corrente nos trechos

Limites de Correntes	
Trechos	Correntes (A)
0-3	1700
3-4	1500
3-5	1500
5-6	1500
1-7	1200
7-8	1500
7-9	1500
8-10	1000
8-11	1000
2-12	2500
12-13	1500
12-14	1500
14-15	1500
4-10	1500
9-13	1500
6-15	1500

A Tabela 9 apresenta a solução para este sistema.

Tabela 9: Resultado final para o sistema de 16 barras e 6 chaves

Tensões Finais (kV)					
Cargas	P	Q	A_0	A_1	A_2
3	2	1,6	22,7300	-	-
4	3	1,5	22,6359	-	-
5	2	0,8	22,5717	-	-
6	1,5	1,2	22,5463	-	-
7	4	2,7	-	22,1161	-
8	5	3	-	21,5280	-
9	1	0,9	-	21,2357	-
10	0,6	0,1	-	21,5088	-
11	4,5	2	-	21,3826	-
12	1	0,9	-	20,9680	-
13	1	0,7	-	21,1483	-
14	1	0,9	-	20,8507	-
15	2.1	1	-	20,8189	-

É possível perceber que o algoritmo executado encontrou uma configuração possível para o sistema, cumprindo as restrições de radialidade e limite máximo de corrente no trecho. Imagina-se que, com algumas modificações, o método proposto é expansível para tratar outras restrições típicas do sistema elétrico, caso sejam adicionadas ao problema.

Do exposto, o método em termos de simulação cumpriu seu objetivo. A comunicação entre os agentes alocados, bem como a coleta descentralizada dos dados, que subsidia o cálculo do fluxo de potência e a tomada de decisão sobre a configuração da topologia do sistema elétrico, é possível.

6 Conclusões e Trabalhos Futuros

Os sistemas elétricos do futuro, através do uso de técnicas de inteligência artificial, terão um alto nível de automação e também serão dotados de uma camada de comunicação entre os equipamentos alocados a rede. A troca de informações pos-

sibilitada por esta camada será usada como subsídios para tomadas de decisão e funcionalidades dos *smart grids*.

Este artigo tratou da aplicação de sistemas multiagentes para gerenciamento topológico de sistemas elétricos distribuição inteligentes. O método desenvolvido encontrou topologias viáveis do sistema, de forma distribuída, dado um conjunto de restrições como radialidade e fluxo máximo de corrente nos trechos.

Espera-se que mais pesquisas sejam possibilitadas a partir da ideia de uso de métodos distribuídos e inteligência artificial para simulações de funcionalidades de *smart grids*. Uma possibilidade de trabalho futuro dessa pesquisa é a criação de uma versão dinâmica do método aqui apresentado, que aproxima a simulação de um ambiente de produção real.

Agradecimentos

Os autores agradecem os apoios prestados pela CAPES e FAPESP, que subsidiaram a realização e publicação deste trabalho.

Referências

- Bellifemine, F. L., Caire, G. and Greenwood, D. (2007). *Developing Multi-Agent Systems with JADE*, Wiley.
- Brown, R. (2008). Impact of Smart Grid on distribution system design, *Power and Energy Society General Meeting-Conversion and Delivery of Electrical Energy in the 21st Century, 2008 IEEE*, IEEE, pp. 1–4.
- Civanlar, S., Grainger, J., Yin, H. and Lee, S. (1988). Distribution feeder reconfiguration for loss reduction, *IEEE Transactions on Power Delivery* **3**(3): 1217–1223.
- Cormen, T. H., Leiserson, C. E., Rivest, R. L. and Stein, C. (2009). *Introduction to Algorithms, Second Edition*, The MIT Press.
- Guerra, W., Unsihuay-Vila, C. and Kagan, N. (2011). Impacto da Geração Distribuída na Reconfiguração de Sistemas de Distribuição de Energia Elétrica Utilizando Busca em Vizinhança Variável, *XLIII SBPO*, SOBRAPO, São Paulo.
- Kersting, W. (2006). *Distribution system modeling and analysis*, Vol. 15, CRC.
- Mantovani, J., Casari, F. and Romero, R. (2000). Reconfiguração de sistemas de distribuição radiais utilizando o critério de queda de tensão, *Revista Controle e Automação* **11**(03).
- McArthur, S., Davidson, E., Catterson, V., Dimeas, A., Hatziargyriou, N., Ponci, F. and Funabashi, T. (2007a). Multi-agent systems for power engineering applications - Part I: concepts, approaches, and technical challenges, *Power Systems, IEEE Transactions on* **22**(4): 1743–1752.
- McArthur, S., Davidson, E., Catterson, V., Dimeas, A., Hatziargyriou, N., Ponci, F. and Funabashi, T. (2007b). Multi-agent systems for power engineering applications - Part II: technologies, standards, and tools for building multi-agent systems, *Power Systems, IEEE Transactions on* **22**(4): 1753–1759.
- Rumley, S., Kagi, E., Rudnick, H. and Germond, A. (2008). Multi-agent approach to electrical distribution networks control, *Computer Software and Applications, 2008. COMP-SAC'08. 32nd Annual IEEE International*, IEEE, pp. 575–580.
- Saraiva, F. d. O., Sousa, A. S. and Asada, E. N. (2011). Implementação distribuída do algoritmo de dijkstra através de sistemas multiagentes, *X SBAI*, Vol. X, SBA, pp. 51–56.
- Shoham, Y. and Leyton-Brown, K. (2009). *Multi-agent systems: Algorithmic, game-theoretic, and logical foundations*, Cambridge Univ Pr.
- Wei, X., Yu-hui, Z. and Jie-lin, Z. (2009). Energy-efficient distribution in smart grid, *Sustainable Power Generation and Supply, 2009. SUIPERGEN'09. International Conference on*, IEEE, pp. 1–6.
- Wooldridge, M. (2009). *An introduction to multi-agent systems*, Wiley, England.