Modeling Microgrid Islanding Problems as DCOPs

Saurabh Gupta, William Yeoh, Enrico Pontelli
Department of Computer Science
New Mexico State University
Las Cruces, NM, 88003
Email: {saurabh, wyeoh, epontell}@nmsu.edu

Palak Jain, Satish J. Ranade
Klipsch School of Electrical and Computer Engineering
New Mexico State University
Las Cruces, NM, 88003
Email: {palak, sranade}@nmsu.edu

***Abstract*—In this paper, we formulate the microgrid islanding problem as distributed constraint optimization problem (DCOP) and investigate the feasibility of solving it using off-the-shelf DCOP algorithms. This paper puts forward the potential of distributed constraint reasoning paradigm as a candidate for solving common microgrids problems.**

***Index Terms*—Microgrid, Islanding, Distributed Constraint Optimization.**

I. Introduction

Every year, several major and minor weather calamities either severely damage the power infrastructure or cause power disruptions around the world. The recent power outage due to a severe storm affected more than 200,000 people in the areas of Oklahoma, Missouri, and Illinois [1]. It highlighted the need to modernize the power system in order to avoid such power failures. *Customer-Driven Microgrid (CDMG)* is a new paradigm which offers a reliable option to minimize and localize such power failures and can be operated with or without the main power grid [2]. A customer-driven microgrid consists of a conventional residential feeder serving residential and commercial facilities with controllable loads and distributed generation sources; loads and sources are managed by intelligent software agents on behalf of customers. During major power outages and blackouts, these smart facilities in a microgrid can form *islands* and locally satisfy the power demands of the customers. The literature has explored methodologies to automatically realize the grid reconfiguration operations necessary to achieve optimal islanding; this is, however, not ideal—centralized decision points represent single points of failure and may affect overall security, reliability and robustness. The challenge is how to resolve the islanding problem in a fully distributed manner.

This paper proposes a load-centric formulation of the islanding problem. A load-centric approach is critical because, in case of an emergency, power accessibility is a more important issue than economical reasons. In such scenarios, the goal should be to satisfy at least the critical demands of the largest possible population.

This paper also proposes to solve the microgrid islanding problem in a fully decentralized manner using the *Distributed Constraint Optimization Problem (DCOP)* framework [3], [4], [5]. The DCOP framework represents a relatively recent branch within the general field of distributed Artificial Intelligence. A DCOP is a problem in which a group of agents must distributedly choose values for a set of variables, in such a way to optimize the overall cost of a set of constraints over the variables. The focus of this paper is to show how a general power network optimization problem can be modeled as a DCOP and thus can be solved in a fully decentralized manner to find a globally optimal solution. This paper is an attempt towards introducing a general distributed problem solving framework to solve modern distribution system problems.

The structure of this paper is as follows. We first provide some background on the islanding problem; Section II describes the work done by other researches in this field. Section III presents our formulation of islanding problem. In Section IV, we give a brief introduction of distributed constraint optimization problems, some off-the-shelf algorithms to solve DCOP and their taxonomy. We then show how we can use DCOP to model the islanding problem in Section V. Finally, we present some test cases in Section VI and conclude in Section VII.

II. Background

Islanding in conventional distribution feeder or microgrids is very similar to the reconfiguration and restoration problems in conventional power systems, shipboard systems and isolated power systems [6], [7], [8], [9], [10], [11]. Reconfiguration, in this context, is the restoration of energy service under the conditions of system failure, by isolating faults and balancing the load and generation in order to minimize the zones without power. Researchers have adopted several approaches to address these problems and applied classical as well as evolutionary optimization algorithms with single or multiple objectives. For example, Rao *et al.* [6] propose an artificial bee colony algorithm to determine the operation of switches for a radial distribution network reconfiguration. The authors minimize the network losses while satisfying the voltage and current limits constraints, and argue the superiority of the proposed methods compared to other optimization schemes, such as genetic algorithm and simulated annealing. The authors in [7] develop a genetic algorithm based method for network restoration after clearing of a fault. Vadivoo *et al.* [7] solve a multi-objective function with fundamental network constraints to restore IEEE 16 and 33 bus feeders with and without adding distributed generation; they argue that the addition of distributed generation significantly reduces system losses and voltage drop. In [8], a method based on the integration of genetic algorithm and fuzzy multi objective programming is proposed for network configuration, under both normal and faulty conditions. The state of the switches is determined for the predetermined fault locations, and losses are minimized while satisfying the voltage magnitudes and line flows.

While the above mentioned approaches adopt centralized methods to solve the network reconfiguration, restoration, and islanding problems, some researchers have also explored distributed approaches to tackle these problems [9], [10], [11]. In [9], a decentralized agent based control is applied for reconfiguration of a mesh-structured power system. The proposed market-based reasoning algorithm aims to maximize the critical loads of the system by satisfying the voltage and current limit constraints. The authors in [11] also use DPOP, an algorithm to tackle DCOP problems, to solve the reconfiguration of power networks for power restoration after a fault. However, their objective is to minimize the global cost of the network with capacity constraints in the lines. In the literature, the majority of the researchers have solved the reconfiguration problem using different algorithms by merely switching objectives and constraints. The global aim is to minimize either the network cost or the losses. Our approach is quite unique, in the sense that we are taking a load centric approach, where our goal is to minimize the unserved load of the system—this is more critical after a fault when a system needs restoration.

In our own previous work, we developed a decentralized, rule-based algorithm for islanding a microgrid in case of a fault with only neighbor-to-neighbor communication [10]. A fully distributed solution was demonstrated for the islanding problem with complete ac power flow constraints in [10]. The approach made use of the Gauss-Seidel algorithm for power flow, which is intuitive for distributed use, but unfortunately has poor convergence guarantees. An alternative method to solve the power flow equations is the Newton-Raphson method, which has better convergence properties, but it is less suitable to distributed application.

This paper is a step towards realizing the potential of distributed constraint optimization modeling and algorithms, like DPOP, with guaranteed convergence, applied to the resolution of non-linear problems in power systems.

III. The Islanding Problem Formulation

The islanding problem frames the following question: "*Determine the status of switches in order to serve critical load and then maximize the load served*?" Figure 1 shows a mathematical formulation of the problem, where

- n is the number of nodes in the network;
- Pg_i is the power generation at node i;
- Pl_i is the power consumption at node i;
- Pt_{ij} is the transmission line flow between nodes i and j;
- $\widehat{Pg_i}$ is the maximum generation capacity at node i;
- $\widehat{Pl_i}$ and $\widehat{pl_i}$ is the maximum and minimum power consumption at node i, respectively;
- $\widehat{Pt_{ij}}$ is the maximum transmission line capacity between nodes i and j;
- S_{ij} is the status of switch between nodes i and j.

$$\textbf{Minimize} \sum_{i=1}^{n} W(\widehat{pl_i} - Pl_i) + (\widehat{Pl_i} - Pl_i) \tag{1}$$

subject to the following constraints:

$$Pg_i - Pl_i - \sum_j Pt_{ij} = 0 \qquad \forall i \tag{2}$$

$$Pt_{ij} + Pt_{ji} = 0 \qquad \forall i,j \tag{3}$$

$$Pg_i \in \{0, \ldots, \widehat{Pg_i}\} \qquad \forall i \tag{4}$$

$$Pl_i \in \{0, \widehat{pl_i}, \ldots, \widehat{Pl_i}\} \qquad \forall i \tag{5}$$

$$Pt_{ij} \in \{-\widehat{Pt_{ij}}, \ldots, \widehat{Pt_{ij}}\} \qquad \forall i \tag{6}$$

$$-S_{ij} \times \widehat{Pt_{ij}} \leq Pt_{ij} \leq S_{ij} \times \widehat{Pt_{ij}} \qquad \forall i,j \tag{7}$$

$$S_{ij} \leq |Pt_{ij}| \qquad \forall i,j \tag{8}$$

$$S_{ij} \in \{0,1\} \qquad \forall i,j \tag{9}$$

Fig. 1: Mathematical Model of the Islanding Problem

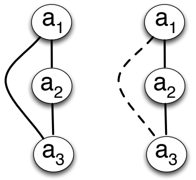
This formulation restricts all the variables to take only integer values. The objective function in Equation (1) is to minimize the unserved load in the network. The objective function is the sum of two parts. The first part of the objective function is the difference between the critical load ($\widehat{pl_i}$) and the served load (Pl_i) at node i. This part of the objective function is multiplied by a weight (W), which should be bigger than the maximum load at any node. This ensures that the critical loads of all the nodes should be served first. The second part of the objective function is the difference between the maximum load at node i ($\widehat{Pl_i}$) and served load (Pl_i). This part ensures that the maximum possible load of node i is served.

This optimization problem is subject to the following constraints. Equation (2) assumes that there are no losses in the system. Thus, total power generated in the network is equal to the total power consumed by the network. Equation (3) ensures that the power transferred from node i to node j is equal to the power received by node j. The constraints in Equations (4) and (5) represent the generation and load limits at node i; the constraint in Equation (6) represents the capacity of the line between node i and node j. The constraint in Equation (7) ensures that power is flowing between nodes i and j only if the switch between the two nodes is closed. Equation (8) makes sure that the switch is closed only when the the power is flowing between the node i and node j. The constraint in Equation (9) restricts the possible values of S_{ij} to 0 and 1, where a value of 0 (1) indicates that the switch between nodes i and j is open (closed).

IV. Distributed Constraint Optimization Problems

A. Basic Definitions

Let us review the *distributed constraint optimization problem* (DCOP) framework, which we will later use to model the islanding problem described in Section III. A DCOP [3], [4], [5] is defined by a tuple $\langle \mathcal{A}, \mathcal{X}, \alpha, \mathcal{D}, \mathcal{F} \rangle$, where:



(a) (b)

for $i < j$

a_i	a_j	Costs
0	0	5
0	1	8
1	0	20
1	1	3

(c)

Fig. 2: Example DCOP

- $\mathcal{A} = \{a_1, \ldots, a_p\}$ is a set of agents.
- $\mathcal{X} = \{x_1, \ldots, x_m\}$ is a set of variables.
- $\alpha : \mathcal{X} \to \mathcal{A}$ maps each variable to an agent.
- $\mathcal{D} = \{D_1, \ldots, D_m\}$ is a set of finite domains, where D_i is the domain assigned to the variable x_i.
- $\mathcal{F} = \{F_1, \ldots, F_k\}$ is a set of n-ary constraints, where each constraint $F_i : D_{i_1} \times \ldots \times D_{i_n} \mapsto \mathbb{N} \cup \{\infty\}$ specifies the cost of each combination of values of variables x_{i_1} to x_{i_n}.

A *solution* is a value assignment for a subset of variables. The cost of the solution is the evaluation of all constraints on that solution. A solution is *complete* iff it is a value assignment for all variables. The goal is to find a cost-minimal complete solution X^*:

$$X^* = \arg\min_X \sum_{F_i \in \mathcal{F}} F_i(x_{i_1}, \ldots, x_{i_n} \mid x_{i_1}, \ldots, x_{i_n} \in X) \tag{10}$$

A *constraint graph* visualizes a DCOP instance, where nodes in the graph correspond to variables in the DCOP and edges connect pairs of variables appearing in the same constraint.

Many complete algorithms (see also Figure 3) operate by ordering the variables in the DCOP by constructing a DFS pseudo-tree during a pre-processing step. A *DFS pseudo-tree* arrangement has the same nodes and edges as the constraint graph, and: **(i)** there is a subset of edges, called *tree edges*, that form a rooted tree, and **(ii)** two variables appearing in the same constraint appear in the same branch of that tree. The other edges are called *backedges*. A tree edge connects two nodes that are in a parent-child relationship, while a backedge connects two nodes that are in a pseudo-parent and pseudo-child relationship. A backedge between the two nodes thus represents a constraint shared by these nodes, which are not connected by a tree-edge. A DFS pseudo-tree arrangement can be constructed using distributed DFS algorithms [12].

Figure 2(a) shows the constraint graph of a sample DCOP, with three agents controlling variables with domain $\{0, 1\}$. The three edges of the constraint graph represent three constraints shared by variables a_1, a_2, and a_3. Figure 2(b) shows one possible pseudo-tree (the dotted line is a *backedge*). If two variables a_i and a_j share a soft-constraint then the third column in Figure 2(c) shows the cost of the three constraints when a_i and a_j assume the values given in first and second columns respectively. A DCOP solution minimizes or maximizes the sum of the constraint costs.

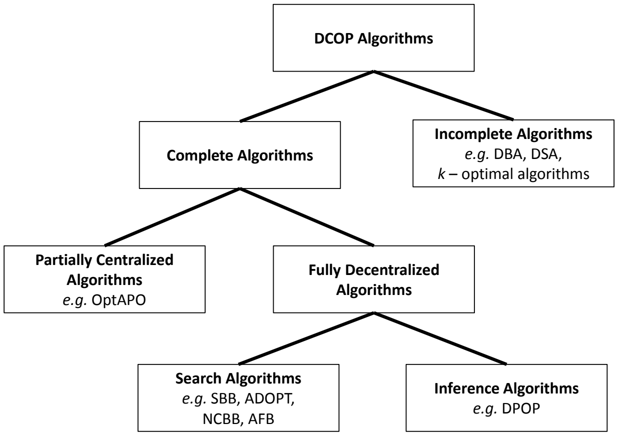


Fig. 3: Taxonomy of DCOP Algorithms

B. Distributed Constraint Optimization Algorithms

Each agent in a DCOP algorithm is responsible for assigning values to its variables and coordinating with other agents, via messages, to find a complete assignment to the DCOP. Typically, DCOP algorithms assume that the messages sent between agents can be delayed by a finite amount of time but are never lost. Additionally, all DCOP algorithms are designed such that each agent requires only *local information*, that is, it only knows about its constraints and its neighboring agents that it shares constraints with.

Figure 3 shows a general taxonomy of DCOP algorithms. They generally fall into two categories: *complete algorithms*, that are guaranteed to find an optimal solution, and *incomplete algorithms* that may not lead to an optimal solution. *Incomplete algorithms* typically employ local search strategies to find locally optimal solutions, and thus can potentially get trapped in local optima. Nevertheless, since solving DCOP problems optimally is NP-hard [3], such algorithms are convenient in large problems, where finding the cost-minimal solution might incur high computation costs. Examples of incomplete DCOP algorithms include DBA [13], MGM [14], DSA [15], Max-Sum [16] and the more recent class of region-optimal algorithms [17], [18], [19], [20].

Complete algorithms can be divided into the following two groups:

Partially centralized algorithms allow some agents to transfer their constraint information to a central agent for processing. OptAPO [21] is an example of a partially centralized algorithm that uses cooperative mediation, where certain agents act as mediators to centrally solve overlapping subproblems.

Fully decentralized algorithms do not have central agents that collect constraint information of other agents. Rather, every agent has access to only its own constraint information. Fully decentralized algorithms are further divided into two groups: inference and search algorithms.

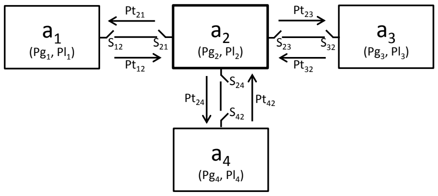


Fig. 4: Example 4-Node Network for the Islanding Problem

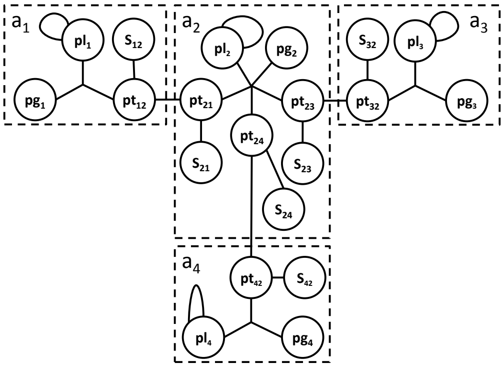


Fig. 5: Constraint Graph for the Network in Figure 4

- *Inference algorithms*, like DPOP [4], use dynamic programming to propagate aggregated constraint utilities from one agent to another agent. The number of messages sent between agents is linear in the number of agents. However, its memory requirements are exponential in the induced width of the problem which, in the worst case, can be exponential in the number of agents.
- *Search algorithms*, like SBB [22], ADOPT [3], NCBB [23] and AFB [24], search through the solution space to find an optimal solution. Their memory requirements are polynomial in the number of agents. However, the number of messages sent between agents can be exponential in the number of agents.

Therefore, both groups are desirable under different conditions as there is a tradeoff between space (memory requirements) and time (number of messages sent).

V. Islanding Problem as DCOP

We now show how one can model the Islanding problem described in Sections III as a DCOP, and use existing DCOP algorithms to solve it. We will illustrate this model using a 4-node network as an example. However, the techniques proposed here are easily generalizable to arbitrary networks. Figure 4 shows our example, where each node is assumed to have an autonomous agent that is capable of controlling the power generation and consumption at its node.

Using the definition of DCOPs described in Section IV, we will now define the tuple $\langle \mathcal{A}, \mathcal{X}, \alpha, \mathcal{D}, \mathcal{F} \rangle$ for our example:

- $\mathcal{A}$ consists of four agents $\{a_1, a_2, a_3, a_4\}$, one for each node of the network.
- $\mathcal{X}$ consists of variables Pg_i, Pl_i, Pt_{ij} and S_{ij} for each agent a_i and its neighboring agent a_j.
- α maps each variable Pg_i, Pl_i, Pt_{ij} and S_{ij} to agent a_i.
- $\mathcal{D}$ consists of the domains D_{Pg_i}, D_{Pl_i}, $D_{Pt_{ij}}$ and $D_{S_{ij}}$ for each agent a_i and its neighboring agent a_j, where:
  - $D_{Pg_i} = \{0, \dots, \widehat{Pg_i}\}$ is the domain of variable Pg_i;
  - $D_{Pl_i} = \{\widehat{pl_i}, \dots, \widehat{Pl_i}\}$ is the domain of variable Pl_i;
  - $D_{Pt_{ij}} = \{-\widehat{Pt_{ij}}, \dots, \widehat{Pt_{ij}}\}$ is the domain of variable Pt_{ij};
  - $D_{S_{ij}} = \{0, 1\}$ is the domain of variable S_{ij};
- $\mathcal{F}$ consists of the following n-ary constraints for each agent a_i:
  - Hard constraint: $Pg_i - Pl_i - \sum_{a_j \in N_i} Pt_{ij} = 0$, where N_i is the set of neighboring agents of agent a_i.
  - Hard constraint: $Pt_{ij} + Pt_{ji} = 0$ for each neighboring agent a_j.
  - Hard constraint: $-S_{ij} \times \widehat{Pt_{ij}} \leq Pt_{ij} \leq S_{ij} \times \widehat{Pt_{ij}}$ for each neighboring agent a_j.
  - Hard constraint: $S_{ij} \leq |Pt_{ij}|$ for each neighboring agent a_j.
  - Soft constraint: $W(\widehat{pl_i} - Pl_i) + (\widehat{Pl_i} - Pl_i)$.

  The cost of a hard constraint is 0 if the constraint is satisfied and ∞ otherwise. The cost of a soft constraint is exactly the evaluation of the constraint.

Given this model, the constraint graph for our example network is shown in Figure 5. The solution to such DCOP maps to a solution of the islanding problem because of the following:

- The objective function of the islanding problem in Equation (1) can be decomposed into the sum of individual functions for each agent, and each such function is represented as a soft constraint. A DCOP solution minimizes the sum of constraint costs, and thus minimizes the objective function as well.
- The constraints in Equations (2), (3), (7) and (8) are satisfied, because the DCOP solution will have a cost of ∞ otherwise, due to the hard constraints.
- The constraints in Equations (4), (5), (6) and (9) are satisfied, because the possible values of the variables Pg_i, Pt_i, Pt_{ij} and S_{ij} are bounded from above by $\widehat{Pg_i}$, $\widehat{Pl_i}$, $\widehat{Pt_{ij}}$, and 1 respectively, and from below by 0, $\widehat{pl_i}$, $-\widehat{Pt_{ij}}$, and 0, respectively.

VI. Test Cases

Using the above formulation, we solved a 7-node test feeder, as shown in Figure 6, using DPOP [4], one of the state-of-the-art DCOP algorithms, whose implementation is publicly available as part of the FRODO simulator [25].

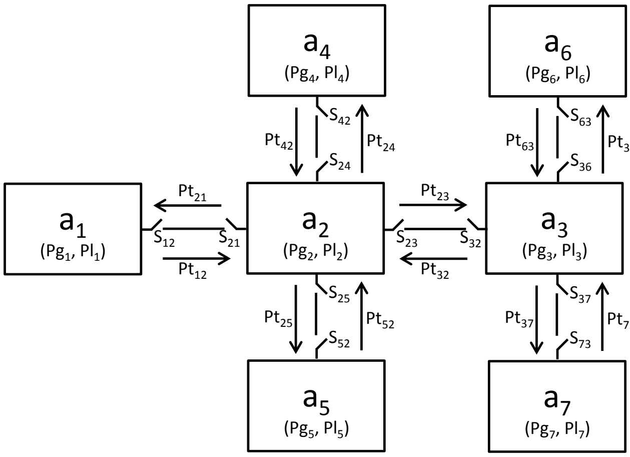


Fig. 6: 7 Node Test Feeder

We ran the following three test cases and the results are shown in Table I.

A. Test Case I

For the first test case, we assigned the following values to the parameters:

$$\begin{array}{llll}
\widehat{Pg_1} = 1 & \widehat{Pg_2} = 0 & \widehat{Pg_3} = 2 & \widehat{Pg_4} = 7 \\
\widehat{Pg_5} = 7 & \widehat{Pg_6} = 7 & \widehat{Pg_7} = 7 & \\
\widehat{Pl_1} = 5 & \widehat{Pl_2} = 5 & \widehat{Pl_3} = 5 & \widehat{Pl_4} = 5 \\
\widehat{Pl_5} = 5 & \widehat{Pl_6} = 5 & \widehat{Pl_7} = 5 & \\
\widehat{pl_1} = 3 & \widehat{pl_2} = 2 & \widehat{pl_3} = 2 & \widehat{pl_4} = 2 \\
\widehat{pl_5} = 2 & \widehat{pl_6} = 2 & \widehat{pl_7} = 2 & \\
\widehat{Pt_{12}} = 2 & \widehat{Pt_{32}} = 2 & \widehat{Pt_{42}} = 2 & \\
\widehat{Pt_{52}} = 2 & \widehat{Pt_{36}} = 2 & \widehat{Pt_{37}} = 2 & \\
W = 100 & & &
\end{array}$$

The results (Table I) show that all the nodes are trying to satisfy their maximum possible loads. However, in order to make the biggest possible island, node 2 transfers its 2 units of power to node 1 instead of satisfying its full load, so that node 1 can satisfy at least its critical load. Thus, all the nodes make one single island.

B. Test Case II

For the second test case, we used the same parameters as in case 1 except that we increased the critical load value of node 1 from 3 to 4. As the results show in the Table I, now node 1 has no way to satisfy its critical load, so it isolates itself from the island and the switches S_{12} and S_{21} assume the value 0. Thus, nodes 2 to 7 make one self-sustained island.

C. Test Case III

For the third test case we assigned the following values to the the parameters:

TABLE I: Experimental Results

Variables	Test Case I	Test Case II	Test Case III
Pg_1	1	0	0
Pg_2	0	0	0
Pg_3	2	2	1
Pg_4	7	7	5
Pg_5	7	7	5
Pg_6	7	7	5
Pg_7	7	7	5
Pl_1	3	0	0
Pl_2	3	5	0
Pl_3	5	5	5
Pl_4	5	5	5
Pl_5	5	5	5
Pl_6	5	5	3
Pl_7	5	5	3
Pt_{12}	-2	0	0
Pt_{21}	2	0	0
Pt_{32}	1	1	0
Pt_{23}	-1	-1	0
Pt_{42}	2	2	0
Pt_{24}	-2	-2	0
Pt_{52}	2	2	0
Pt_{25}	-2	-2	0
Pt_{36}	-2	-2	-2
Pt_{63}	2	2	2
Pt_{37}	-2	-2	-2
Pt_{73}	2	2	2
S_{12}	1	0	0
S_{21}	1	0	0
S_{32}	1	1	0
S_{23}	1	1	0
S_{42}	1	1	0
S_{24}	1	1	0
S_{52}	1	1	0
S_{25}	1	1	0
S_{36}	1	1	1
S_{63}	1	1	1
S_{37}	1	1	1
S_{73}	1	1	1

$$\begin{array}{llll}
\widehat{Pg_1} = 0 & \widehat{Pg_2} = 0 & \widehat{Pg_3} = 1 & \widehat{Pg_4} = 5 \\
\widehat{Pg_5} = 5 & \widehat{Pg_6} = 5 & \widehat{Pg_7} = 5 & \\
\widehat{Pl_1} = 5 & \widehat{Pl_2} = 5 & \widehat{Pl_3} = 5 & \widehat{Pl_4} = 5 \\
\widehat{Pl_5} = 5 & \widehat{Pl_6} = 5 & \widehat{Pl_7} = 5 & \\
\widehat{pl_1} = 5 & \widehat{pl_2} = 5 & \widehat{pl_3} = 5 & \widehat{pl_4} = 5 \\
\widehat{pl_5} = 5 & \widehat{pl_6} = 3 & \widehat{pl_7} = 3 & \\
\widehat{Pt_{12}} = 2 & \widehat{Pt_{32}} = 2 & \widehat{Pt_{42}} = 2 & \\
\widehat{Pt_{52}} = 2 & \widehat{Pt_{36}} = 2 & \widehat{Pt_{37}} = 2 & \\
W = 100 & & &
\end{array}$$

In this scenario, node 1 and node 2 do not have enough generation to satisfy their own critical loads and the lines connecting node 1 and node 2 with the rest of the network do not have enough capacity to transmit power for satisfying at least their critical loads. So, these two nodes isolate themselves from the network. This results in three islands. Node 4 and node 5 make two separate single node islands and node 3, 6, and 7 form one island together.

VII. Conclusions and Discussions

In this paper, we presented a load centric formulation of the microgrid islanding problem. We also demonstrated how such a formulation can be modeled and solved using DCOPs in a fully distributed manner, with a guarantee to find globally optimal solution using off-the-shelf DCOP algorithms. This decentralized formulation is appealing as it allows the problem to be solved in parallel and maintains the privacy of users by providing only local information to each agent instead of the information of the entire problem.

This formulation is a work in progress and our future work includes the expansion of this formulation by incorporating the non-linear power flow constraints and solve it for standard IEEE test feeders. Another challenge for our future work is to exploit domain properties to increase the scalability of these algorithms in order to deploy them in the real world.

Acknowlegement

This research is partially supported by Department of Energy's Microgrid Education and Research grant DE-OE0000098

References

[1] H. Brandes. (2013) Update 3-death toll rises to nine from oklahoma tornadoes. [Online]. Available: http://www.reuters.com/article/2013/06/02/usa-tornadoes-idUSL2N0ED09T20130602

[2] S. Ranade, J. Mitra, S. Suryanarayanan, and P. Riberio, "A holistic approach to customer-driven microgrids, Tech. Rep. NSF Grants ECCS#0702208 and #0757956, 2011.

[3] P. Modi, W.-M. Shen, M. Tambe, and M. Yokoo, "ADOPT: Asynchronous distributed constraint optimization with quality guarantees," *Artificial Intelligence*, vol. 161, no. 1–2, pp. 149–180, 2005.

[4] A. Petcu and B. Faltings, "A scalable method for multiagent constraint optimization," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2005, pp. 1413–1420.

[5] W. Yeoh and M. Yokoo, "Distributed problem solving," *AI Magazine*, vol. 33, no. 3, pp. 53–65, 2012.

[6] R. S. Rao, S. Narasimham, and M. Ramalingaraju, "Optimization of distribution network configuration for loss reduction using artificial bee colony algorithm," *International Journal of Electrical Power and Energy Systems Engineering*, vol. 1, no. 2, pp. 116–122, 2008.

[7] N. S. Vadivoo and S. M. R. Slochanal, "Distribution system restoration using genetic algorithm with distributed generation," *Modern Applied Science*, vol. 3, no. 4, p. 98, 2009.

[8] Y.-Y. Hong and S.-Y. Ho, "Determination of network configuration considering multiobjective in distribution systems using genetic algorithms," *IEEE Transactions on Power Systems*, vol. 20, no. 2, pp. 1062–1069, 2005.

[9] K. Huang, S. K. Srivastava, D. A. Cartes, and M. Sloderbeck, "Intelligent agents applied to reconfiguration of mesh structured power systems," in *International Conference on Intelligent Systems Applications to Power Systems, 2007*, 2007, pp. 1–7.

[10] P. Jain, S. Ranade, and S. Srivastava, "Island identification in customer-driven micro-grids," in *Proceedings of the IEEE Power and Energy Society (PES) Transmission and Distribution Conference and Exposition*, 2010, pp. 1–7.

[11] A. Kumar, B. Faltings, and A. Petcu, "Distributed constraint optimization with structured resource constraints," in *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2009, pp. 923–930.

[12] Y. Hamadi, C. Bessière, and J. Quinqueton, "Distributed intelligent backtracking," in *Proceedings of the European Conference on Artificial Intelligence (ECAI)*, 1998, pp. 219–223.

[13] K. Hirayama and M. Yokoo, "The distributed breakout algorithms," *Artificial Intelligence*, vol. 161, no. 1–2, pp. 89–115, 2005.

[14] R. Maheswaran, J. Pearce, and M. Tambe, "Distributed algorithms for DCOP: A graphical game-based approach," in *Proceedings of the International Conference on Parallel and Distributed Computing Systems (PDCS)*, 2004, pp. 432–439.

[15] W. Zhang, G. Wang, Z. Xing, and L. Wittenberg, "Distributed stochastic search and distributed breakout: Properties, comparison and applications to constraint optimization problems in sensor networks," *Artificial Intelligence*, vol. 161, no. 1–2, pp. 55–87, 2005.

[16] A. Farinelli, A. Rogers, A. Petcu, and N. Jennings, "Decentralised coordination of low-power embedded devices using the Max-Sum algorithm," in *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2008, pp. 639–646.

[17] J. Pearce and M. Tambe, "Quality guarantees on k-optimal solutions for distributed constraint optimization problems," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2007, pp. 1446–1451.

[18] E. Bowring, J. Pearce, C. Portway, M. Jain, and M. Tambe, "On k-optimal distributed constraint optimization algorithms: New bounds and algorithms," in *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2008, pp. 607–614.

[19] C. Kiekintveld, Z. Yin, A. Kumar, and M. Tambe, "Asynchronous algorithms for approximate distributed constraint optimization with quality bounds," in *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2010, pp. 133–140.

[20] M. Vinyals, E. Shieh, J. Cerquides, J. Rodriguez-Aguilar, Z. Yin, M. Tambe, and E. Bowring, "Quality guarantees for region optimal DCOP algorithms," in *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2011, pp. 133–140.

[21] R. Mailler and V. Lesser, "Solving distributed constraint optimization problems using cooperative mediation," in *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2004, pp. 438–445.

[22] K. Hirayama and M. Yokoo, "Distributed partial constraint satisfaction problem," in *Proceedings of the International Conference on Principles and Practice of Constraint Programming (CP)*, 1997, pp. 222–236.

[23] A. Chechetka and K. Sycara, "No-commitment branch and bound search for distributed constraint optimization," in *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2006, pp. 1427–1429.

[24] A. Gershman, A. Meisels, and R. Zivan, "Asynchronous Forward-Bounding for distributed COPs," *Journal of Artificial Intelligence Research*, vol. 34, pp. 61–88, 2009.

[25] T. Léauté, B. Ottens, and R. Szymanek, "FRODO 2.0: An open-source framework for distributed constraint optimization," in *Proceedings of the Distributed Constraint Reasoning Workshop*, 2009, pp. 160–164.