

A Decentralized Control Architecture for a Microgrid with Power Electronic Interfaces

Niannian Cai, *Student Member, IEEE*, and Joydeep Mitra, *Senior Member, IEEE*

Abstract—This paper presents a decentralized control architecture for the autonomous operation of a microgrid with power electronic interfaces. The control architecture is based on a multi-agent system (MAS) in which all agents are hierarchically equal and there is no central agent. Neighbor to neighbor three-step communication algorithms are used by the MAS to determine real and reactive power mismatches and resolve the dispatch of real and reactive power among the agents within the microgrid. The research described here includes the design of control modes for the power electronic interfaces in the microgrid. The proposed method is demonstrated on a system comprising four microsources, and is implemented using MATLAB/SIMULINK.

Index Terms—Multi-agent system, decentralized architecture, neighbor to neighbor communication and microgrid.

I. INTRODUCTION

AS reported in [1], electricity generation will increase approximately 2.4% each year, and by 2030, total generation in the world will reach 30,364 billion kWh. Considering limited amount and greenhouse emissions of fossil-fuel energy, alternative choices such as photovoltaic and wind sources and fuel cells attract power industry. These alternatives are usually small-scale and serve as distributed generation (DG) located in the vicinity of loads. To integrate this large number of DGs into grid, an effective way is to organize them locally, and interface with grid at the PCC (point of common coupling). A microgrid is born under this environment, which consists of a cluster of interconnected distributed generators, loads and intermediate energy storage units cooperating with each other to be served as a controllable load or generator to the grid. A microgrid, as a systematic organization of DGs, distributed loads and storage devices, has larger capacity and more flexibility to fulfill system reliability and power quality requirement.

To control the DGs within microgrid effectively, multi-agent system (MAS) emerges as a new paradigm that facilitates distributed and cooperative control. An agent is an autonomous entity that can perceive and react to its environment and communicate with others to achieve its local goal. MAS is a system consisting two or more agents. These agents collaborate and compete with each other to optimize its local goal, and therefore overall to achieve global goal. An agent has three characteristics: reactivity (reactive to its environment), proactiveness (exhibit goal-directed behavior), and social ability

(communicate with each other) [2]. Agents in microgrid are expected to implement tasks as sensory, communication and actuation [3].

Significant research has been carried out in the design of MAS for the operation of microgrid. Ref. [4] proposes three kinds of agents, control agents, management agents and ancillary agents for optimal energy exchange between the production units of the microgrid and the local loads, as well as the main grid. Ref. [5] presents a decentralized system to discover the capacity of microgrid in a distribution feeder. Ref. [6] discusses a centralized architecture of MAS to control microgrid. Ref. [7] describes a centralized architecture of MAS for the energy management of microgrid. Ref. [8] illustrates MAS can facilitate optimal dispatching of distributed generators on a distribution feeder to provide voltage support. Ref. [9] proposes a centralized MAS architecture for electricity trading and maximizes revenue from the microgrid.

In this paper, we propose a decentralized architecture of multi-agent system for the microgrid with power electronic interfaces, in which all the agents are equal and autonomous and they only communicate with their neighbors to collaborate to achieve the power balancing within the micro-system, therefore, maintain the voltage at any point in the system within required limits and keep the system stable. The advantage of adopting a decentralized architecture instead of centralized is the following: for a centralized architecture, the system relies highly on a central agent that has global information and makes decisions for all other agents. Centralized architecture is usually very sensitive to the system's failure for the reason that once the central agent fails to work, the whole system is under the risk of collapse. While for a decentralized architecture, all the agents within the system are identical, there is no master-slave relationship within the system. An agent that fails to work does not affect the performance of the whole system. Therefore, this architecture increases the system's reliability. Another benefit for employing decentralized architecture is that it can enhance microgrid's flexibility for the reason that all the agents are independent and equal, and can be added and removed readily without informing central agents. Benefit from these two characteristics, decentralized architecture tends to be a more effective and convenient way to control microgrid.

This paper is organized as follows: in the section II, a basic structure of microgrid is presented. Section III will cover communication algorithms for multi-agent system and build model of decentralized agents. Section IV will talk about power electronic interface control under this scheme of MAS and simulation results will be provided in section V.

This work was supported by the National Science Foundation under grant ECS-0702208.

The authors are with the Department of Electrical & Computer Engineering, Michigan State University, East Lansing, MI 48824, USA. (email: cainiann@msu.edu)

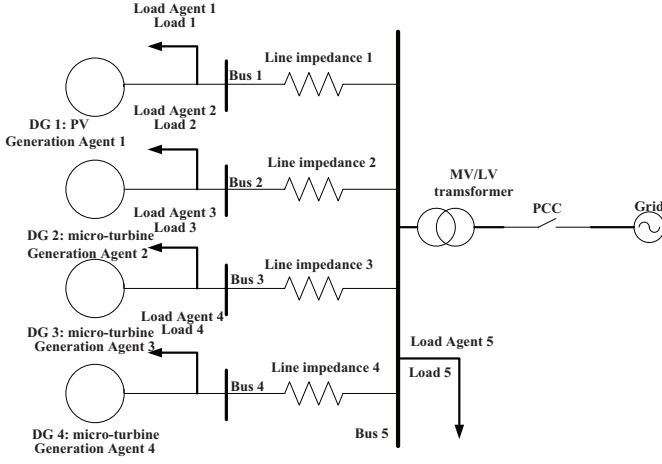


Fig. 1. Configuration of microgrid

II. SYSTEM CONFIGURATION

A basic structure of the microgrid is shown in Fig. 1. A single point called PCC (point of common coupling) connects microgrid to the utility. There are four distributed energy resources in this microgrid, which could be wind generator, photovoltaic cells, micro-turbine, fuel cells or energy storage units. In this configuration, at the sending point of each distributed generator, there is individually a local load. Four DGs also deliver power through transmission line to a common bus where there is a centralized load. In normal operation, the breaker at PCC is closed, and microgrid is operated at grid-connected mode, all DGs can generate at its rated capacity and loads do not need to be shed due to the shortage of local generation. However, when the outage of grid happens, the breaker at PCC will be opened and microgrid is operated at islanded mode. In this situation, the unavailability of main grid to provide or consume extra power within microgrid introduces a problem that DGs cannot simply generate at their willing and loads also cannot simply consume at their need. Both of them need to collaborate and reach a balance that in any instantaneous time, real and reactive power generated should equal to those of consumed within microgrid.

III. MULTI-AGENT SYSTEM

In this part, algorithm for agents' neighbor to neighbor communication to get power mismatch within microgrid is presented first. Section III-B will discuss real and reactive power assignments to agents, and decentralized agent model is given in Section III-C.

A. Algorithm for communication between agents

Consider a system structure shown in Fig. 2. Each node represents an agent. If there is a line between two nodes, it means that these two agents are neighbors, and they can communicate with each other. Two agents without a line between them cannot communicate with each other. An array besides nodes stands for power injection, positive value means power is injected into the point while negative value means power is absorbed from the node. Complex number scales the value of injected power, real number is the value of real

power and imaginary part is the value of reactive power in that point. For a stable microgrid, keeping power generation equal to power consumption is an important target. In order to achieve this target, whether the power generation within the system exceeds power consumption or vice versa is what needs to be known. The other information required to be obtained is that how much mismatch there is between power generation and consumption under current situation. So the problem now under consideration is that with only neighbor to neighbor communication, how can we get power mismatch in this system?

For a decentralized architecture, all agents are equal. Therefore the algorithm for them to get power mismatch within the system should also be identical. The algorithm for each agent to coordinate to get power mismatch includes three steps:

Step 1: Process. Add power of current node into power mismatch and marked current node as "processed".

Step 2: Next. Transmit the power mismatch value to the neighbor who is marked "unprocessed".

Step 3: Return. If the current node has no neighbors or all the neighbors are marked "processed", then return the power mismatch value to the last node. Consider the system shown in

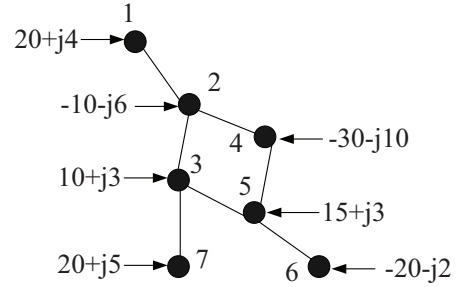


Fig. 2. Multi-agent system structure

Fig. 2, first mark all the agents in the system as "unprocessed" and select an agent as starting agent randomly. Here, assume agent 1 is chosen as starting agent. It has only one neighbor, namely agent 2, which is also marked "unprocessed". So first, agent 1 will go "Process" step, add its power value $20 + j4$ onto power mismatch, which is 0 at the beginning. Then agent 1 gets a new power mismatch $20 + j4$. Second, agent 1 will go "Next" step. It will transmit the new power mismatch to its unprocessed neighbor, agent 2. When agent 2 receives message containing power mismatch value, it will first process as agent 1, add its value onto power mismatch and get new power mismatch $10 - j2$, then it will choose one of its unprocessed neighbors randomly and transmit the new power mismatch to its neighbor. Assume that this time, agent 2 chooses agent 4. The following process will be the same for agent 4 and it will obtain new power mismatch $-20 - j12$ and transmit it to agent 5. Assume that agent 5 selects agent 3 and transmits new power mismatch $-5 - j9$ to agent 3. Then agent 3 add its value to power mismatch and finds that agent 2 is marked "processed", so it can only choose agent 7 and transmit new power mismatch $5 - j6$ to agent 7. When agent 7 receives $5 - j6$ from agent 3, it adds its value onto the power mismatch and gets new value $25 - j1$. However, after that, agent 7 finds that it

has no neighbors, so it goes to “Return” step and returns the new value back to last agent, namely agent 3. When agent 3 receives new value from agent 7, it finds that all of its neighbors are marked “processed”, so it also returns the value to its last agent, namely agent 5. When Agent 5 receives the new value, it finds that one of his neighbors, agent 6, is marked “unprocessed”, so it sends new value to agent 6. Agent 6 adds its value onto the power mismatch and gets $5 - j3$. After that agent 6 finds that it has no neighbors, so it returns the new value back to agent 5. And the return process is the same for the last few agents: agent 5 will return value to agent 4, agent 4 will return to agent 2, agent 2 will return to agent 1. When agent 1 gets the new value and finds all of its neighbors are marked “processed”, the spanning process is over. And at last the final value of power mismatch is obtained in the system using this three step methods.

The message flowing is shown in Fig. 3. The number beside the array indicates the message flowing sequence. Notice that this message flowing sequence is not unique in this system, since at some points, to whom to transmit the message is decided randomly. For example, when agent 2 receives message from agent 1, it can send message to agent 4 first or it can send message to agent 3 first, which are decided by agent 2 randomly. However, no matter to whom agent 2 sends message first, it does not affect the resulting value of power mismatch in the system.

In the algorithm proposed above, all the agents in the system are equal. There is no master-slave relationship between them; This makes the control both stable and flexible.

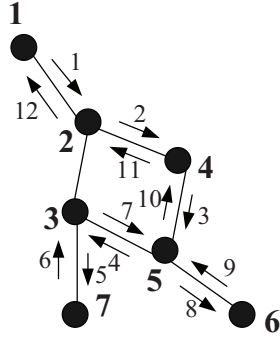


Fig. 3. Message flow sequence for multi-agent system

The MG power mismatch acquisition programming algorithm above is summarized as follows:

- 1) Mark all the agents in the system as “unprocessed”.
- 2) Select an agent randomly as starting agent A_i .
- 3) Starting agent A_i marks itself as “processed” and selects one agent among its neighbors randomly, and sends its corresponding value of real and reactive power P_i , Q_i as P , Q to its neighbor agent A_{i+1} .
- 4) If A_j receives P , Q from A_{j-1} , A_j first checks if A_j is the starting agent and all the neighbors of it are all marked “processed”. If yes, go to 6. Else, A_j checks whether A_j is marked “processed” or “unprocessed”. If A_j is marked as “processed”, go to 5. Else, mark itself

as “processed”, adds its real and reactive power of P_j , Q_j onto P , Q .

- 5) If A_j has no neighbors or all the neighbors are marked “processed”, return the new power mismatch value P and Q to the last agent. Else, selects one agent marked “unprocessed” among its neighbors randomly, and sends the new power mismatch P and Q to it. Then go back to 4.

The flow chart of the agents’ communicating and processing algorithm is shown in Fig. 4.

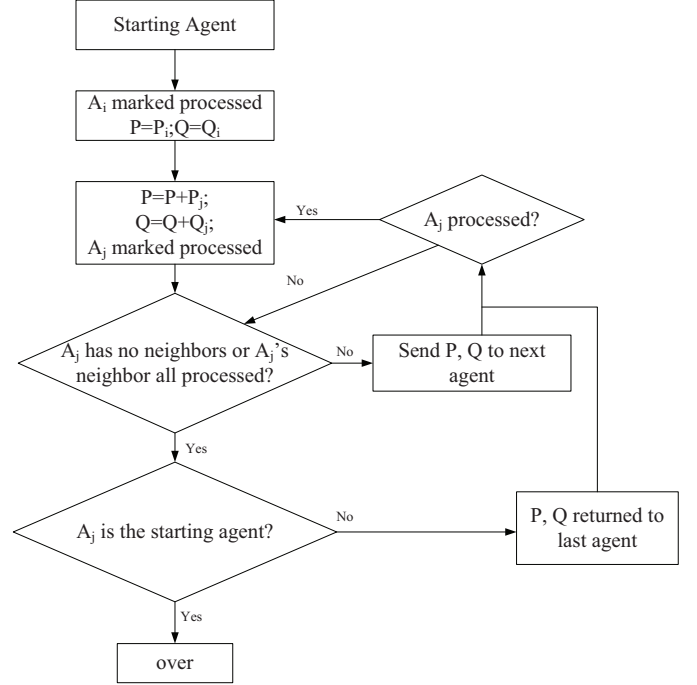


Fig. 4. Flow chart of MG power mismatch acquisition algorithm

After the process, the final P and Q got by the starting agent are the mismatch power in microgrid without considering the loss in the system.

B. Algorithm to balance power in the microgrid

After total power mismatch is obtained in Section III-A, in order to make microgrid stable, a real-time on line operation to keep instantaneous power balance is required. In other words, after considering power losses in the microgrid, the system’s reliability need total power mismatch approaches zero. To meet this requirement, agents cannot generate or consume power at its rated capacity. Instead, they need to collaborate to get new assignments of P and Q . Therefore, another problem arise is that how to on-line control multi-agent system to make total power mismatches within microgrid approach zero by giving out new assignments of P , Q to each agent.

Assume that P , Q represents the real and reactive power mismatch within the micro-system after considering the power losses. When $P, Q > 0$, it means that generation capacity is larger than loads’ demand, so that some amount of generation need to be shed. When $P, Q < 0$, it means that generation capacity is smaller than loads’ demand, so that some load need

to be cut off to make the microgrid stable. Now the problem occurs when $P, Q \neq 0$, which requires load or generation to be resuced in order to achieve power balance.. To achieve this target, loads and generators are designated as high priority, medium priority and low priority.

The “three step” spanning method mentioned in Section III-A will be used to solve this problem again. The algorithm is shown as following:

- 1) Mark all the agents in the system “unprocessed” and select the starting agent in Section III-A also as starting agent A_i .
- 2) Starting agent A_i first subtracts power loss from the power mismatch value P and Q it gets in Section III-A and get new power mismatch P_{i-1} and Q_{i-1}
- 3) A_i first checks if A_i is the starting agent and all the neighbors of it are all marked “processed”. If yes, go to 6. Else, A_i checks whether P_{i-1} and Q_{i-1} are all zeros. If yes, go to 5. Else, checks whether A_i is marked “processed” or “unprocessed”. If A_i is marked as “processed”, go to 4. Else, Mark A_i as “processed” and check whether A_i is marked as low priority. If it is not low priority, go to 4, Else, it means the generators or loads corresponds to agent A_i is not important, then A_i cut off its load or shed its generators to make it best to let the power mismatch approach zero. Consider that A_i can contribute ΔP and ΔQ to the total power mismatch. When $P_{i-1} - \delta P = 0$ and $Q_{i-1} - \delta Q = 0$, which indicates that after the regulation of agent A_i , the power mismatch can be regulated to zero. Then the process is over, go to 5. If $P_{i-1} - \delta P = P_i$ and $Q_{i-1} - \delta Q = Q_i$, while P_i and Q_i are not equal to zero, which means after the effort of A_i , the power mismatch still cannot be regulated to zero. Then A_i memorizes its effort δP and δQ , go to next step.
- 4) If A_i has no neighbors or all the neighbors are marked “processed”, return the power mismatch value to the last agent. Else, selects one agent marked “unprocessed” among its neighbors randomly, and sends its corresponding value of real and reactive power P_i, Q_i as power mismatch to its neighbor agent A_{i+1} . Then go back to 3.
- 5) Return P_i and Q_i as 0 to the last agent A_{i-1} .

The flow chart of this algorithm is shown in Fig. 5. Notice this algorithm is implemented to regulate low-priority generators or loads to satisfy power balance requirement. If this algorithm is over while the power mismatch returned to the starting agent is still not zero. It means that even shedding all the low-priority generators when generation is larger than consumption or cutting off all the low-priority loads when consumption is larger than generation, the balance between the generation and consumption cannot be achieved. Then another iteration, which is similar to the one mentioned above to control medium-priority will begin to further balance the power. After that, if the power still could not be balanced, further iteration to control high-priority group will start until the power is balanced within microgrid.

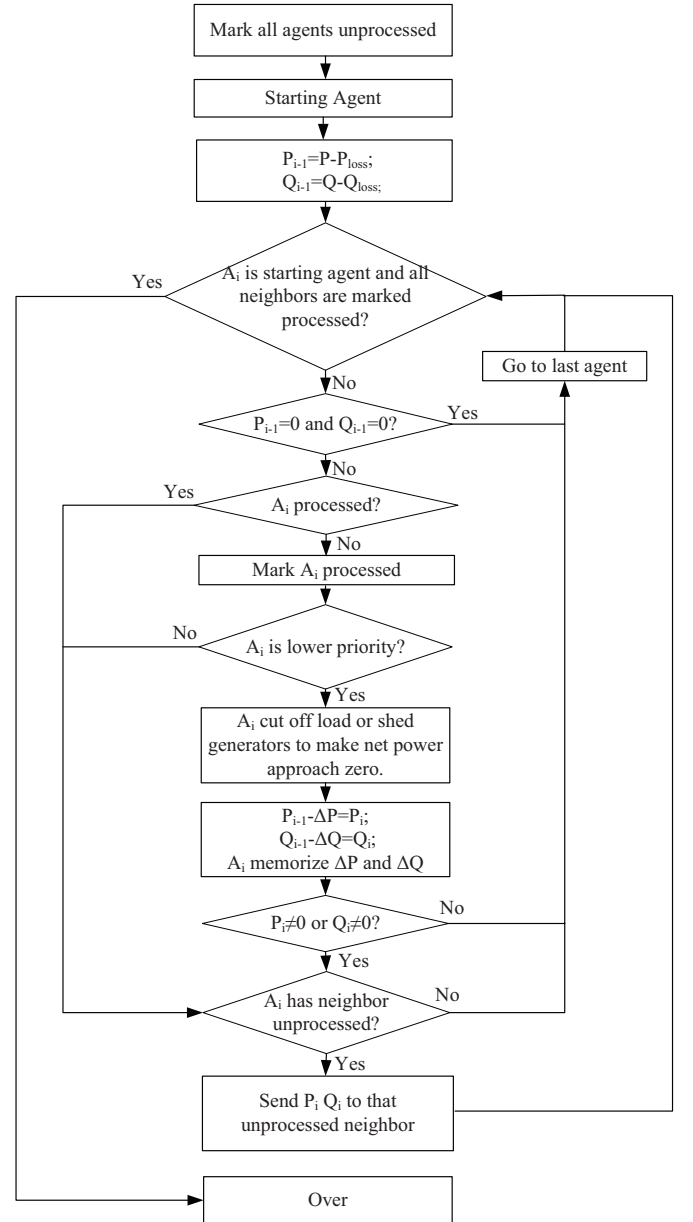


Fig. 5. Flow chart of real and reactive power assignment to agents

C. Agent Model

The role of implementing multi-agent system is to break down a complicated control problem into several small simple problems, and each simple problem will be handled by an entity called agent. A decentralized architecture is beneficial for a flexible and robust multi-agent system, since it can remove and add agents without reconstructing the system and tends to accomplish “plug and play” function.

Components of agent used for microgrid is shown in Fig. 6. An agent within MAS needs a unique ID to distinguish itself from others; apart from that, it also needs to store its neighbors’ ID which is used for communication with neighbors, required by the communication algorithm illustrated in Section III-A. Behaviors enable agents to grasp some ability to take actions. When agents receive power mismatch information from

its neighbors, it needs to judge how to process this information and to whom it needs to deliver the updated information. On the other hand, when real and reactive power assignment is received, agents need to send signals to corresponding power electronic control mode to regulate output. In memory slot, agents need to store basic information of its corresponding generators and loads. For example, the priority sequence, the capacity and the upper limit and lower limit of generators or the load demand. Based on this information, agents can instruct its behaviors.

Agent model
ID:
Neighbors' ID:
Behaviors:
Memory:

Fig. 6. Components of an agent

IV. CONTROLLER DESIGN

Generally, DGs in microgrid are not fully controllable synchronous units that normally responsible microgrid for the voltage and frequency control, and their output are dc or non-utility grade ac that cannot be directly connected to the grid, therefore they need interface with the system by power electronic devices. For dc micro-sources, such as PV or fuel cell, power electronic interface can be a single-stage DC/AC shown in Fig. 7(a) or double-stage DC/DC/AC shown in Fig. 7(b) [10]. Normally, single-stage inverter's topology is simpler and need less components, while its control strategies is more complex for coupling more control variables, and it suffers limited power capacity, compromised output quality and restricted operation range. Double-stage topology can decouple multiple control variables and benefit from simple control strategies. It adopts dc/dc converter to boost the limited output dc voltage of dc micro-source and benefits from larger output range. For non-utility-grade ac, power electronic interface can be two-stage AC/DC/AC converter shown in Fig. 8(a) or multi-stage AC/DC/DC/AC converter shown in Fig. 8(b). Collaborating with each other and achieve a global

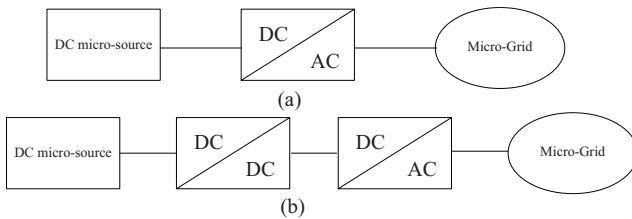


Fig. 7. (a) Single-stage power electronic interface for dc micro-source and (b) double-stage power electronic interface for dc-micro-source

goal, these power electronic devices within microgrid are under control of multi-agent system, which gives instructions to the power electronic interfaces to output specific voltage

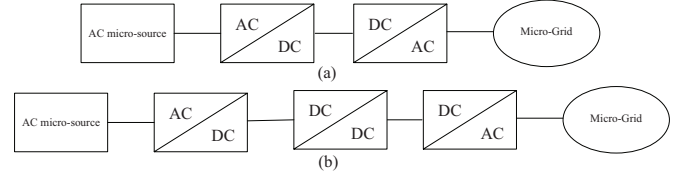


Fig. 8. (a) Double-stage power electronic interface for ac micro-source and (b) multi-stage power electronic interface for ac micro-source

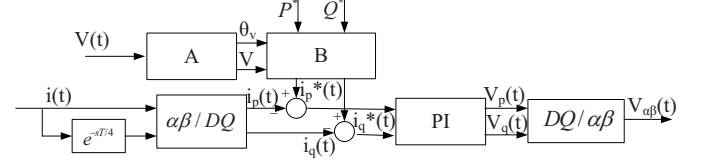


Fig. 9. Control block for power electronic interface

or current and keep the micro-system stable. In the islanded microgrid, when multi-agent system finishes communications and decides the output active and reactive power of each agent described in Section III, agents will set these values as reference to corresponding power electronic control module, which controls the power electronic interfaces to output the demanded power. Thus, the requirement for the control mode is that input P and Q signals, as well as voltage at the sending point of the interface, output corresponding current to meet P and Q requirement.

The control block diagram proposed is shown in Fig. 9. Block A's input is the voltage measured at the sending point of power electronic interface. It is operated as low-pass filter and phase locked loop and outputs the amplitude value V and phase θ_v of $V(t)$.

P^* , Q^* are given from local agent. Block B uses value of V , θ_v , P^* , Q^* to calculate instantaneous value of reference current $i_p^*(t)$, $i_q^*(t)$ by means of equations (1), (2):

$$i_p^*(t) = \text{Re} \left(\frac{P^* + jQ^*}{V \cos \theta_v + jV \sin \theta_v} \right) \quad (1)$$

$$i_q^*(t) = -\text{Im} \left(\frac{P^* + jQ^*}{V \cos \theta_v + jV \sin \theta_v} \right) \quad (2)$$

$i(t)$ is the current measured at the sending point of power electronic interface. After d-q transform illustrated in [11], $i_p(t)$ and $i_q(t)$ can be obtained from $i(t)$. With error between $i_p(t)$ and $i_p^*(t)$, $i_q(t)$ and $i_q^*(t)$ going through a PI controller, $V_p(t)$ and $V_q(t)$ are produced respectively. Then using inverse d-q transform, reference signals for inverter can be acquired.

V. SIMULATION RESULTS

A. Steady-state process

To validate the decentralized architecture, the system displayed in Fig. 1 is simulated in the MATLAB/SIMULINK. DGs' capacities and priority sequence is shown in Table I. DG4 has the largest capacity and can be used to provide voltage reference of microgrid, thus it is set to be high priority. Renewable energy, which costs less and does not

cause pollution, is suitable to produce as much as possible, therefore they are marked as high priority.

TABLE I
GENERATION CAPACITY AND PRIORITY

DG type	Capacity (kVA)	Priority
DG1: PV cells	$1 + j0.5$	medium
DG2: micro-turbine	$2 + j1$	low
DG3: wind-turbine	$2 + j1$	medium
DG4: micro-turbine	$4 + j2$	high

In this steady state simulation, the load demand and priority is shown in Table II:

TABLE II
LOAD DEMAND AND PRIORITY

Load name	Demand (kVA)	Priority
Load 1	$0.3 + j0.1$	high
Load 2	$1 + j0.4$	high
Load 3	$0.5 + j0.2$	medium
Load 4	$2 + j0.5$	medium
Load 5	$7 + j3$	low

The total demand in this case is $10.8 + j4.2$ kVA, which is beyond the total generation capacity $9 + j3.5$ kVA. So, the agent corresponding to Load 5 will cut off some loads, since it is the least important load within this microgrid. Simulation result is shown below:

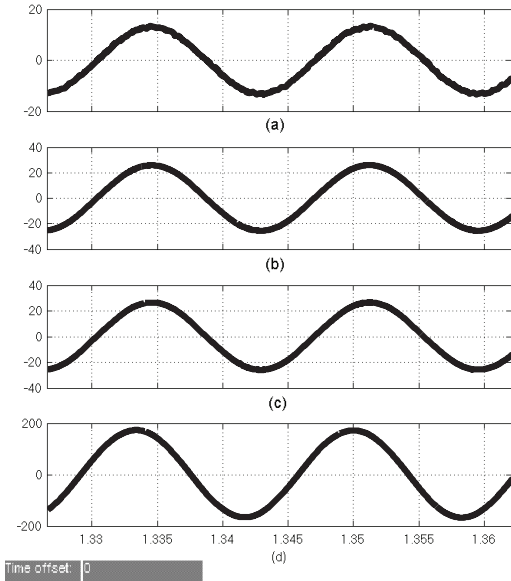


Fig. 10. (a) Current at terminal of DG 1. (b) Current at terminal of DG 2. (c) Current at terminal of DG 3. (d) Voltage at terminal of DG 4.

B. Response to system events

In this case, response to system events are simulated. Generation capacity and priority is also shown in Table III, Load demand and priority is shown in Table IV.

When system changes, such as losing some loads or increasing some generations, the specific agent will detect the changes and start a new three-step iteration to span the system and get new information of the system. During this iteration, all agents within the system can get new control values and operate the system in a new steady state.

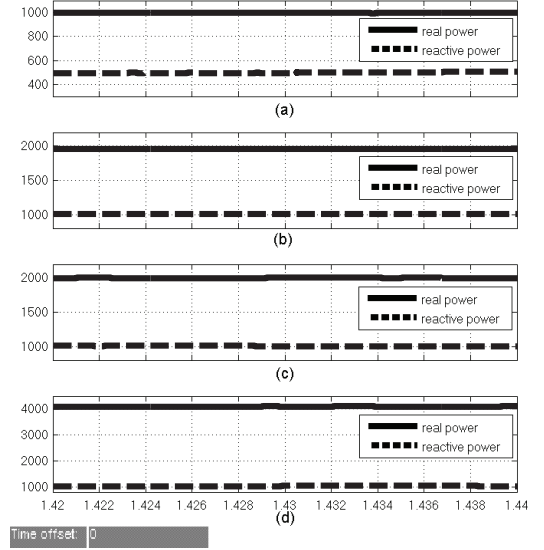


Fig. 11. (a) real and reactive power of DG 1. (b) real and reactive power of DG 2. (c) real and reactive power of DG 3. (d) real and reactive power of DG 4.

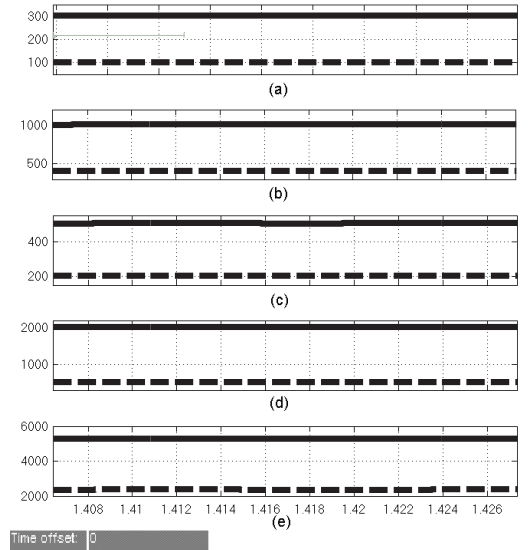


Fig. 12. (a) Real and reactive power of Load 1. (b) real and reactive power of Load 2. (c) real and reactive power of Load 3. (d) real and reactive power of Load 4. (e) real and reactive power of Load 5.

TABLE III
GENERATION CAPACITY AND PRIORITY

DG type	Capacity (kVA)	Priority
DG1: PV cells	$1 + j0.5$	medium
DG2: micro-turbine	$2 + j0.5$	low
DG3: wind-turbine	$1 + j1$	medium
DG4: micro-turbine	$2 + j1$	high

TABLE IV
LOAD DEMAND AND PRIORITY

Load name	Demand (kVA)	Priority
Load 1	$0.3 + j0.1$	high
Load 2	$1 + j0.4$	high
Load 3	$1.5 + j1.2$	medium
Load 4	$1 + j0.5$	medium
Load 5	$3 + j1$	low

In this case at first, the total load in this microgrid is $6.8 + j3.2$ kVA, while the total generation is $6 + j3$ kVA. Thus load 5, as the least priority load, needs to be partially shed and load 5 can only be provided $2.2 + j0.8$ kVA. In this case at 1.2s, load 4 fails to connect to the microgrid, and the breaker of load 4 is opened. At this time, the total load in this system is $5.8 + j2.7$ kVA, which is less than the total generation. Therefore, all the loads in microgrid can be fed enough power at their capacity. While DG2 which is the least priority generation needs to reduce its output power to $1.8 + j0.2$ kVA. Simulation results are shown in Fig. 13, 14 and 15.

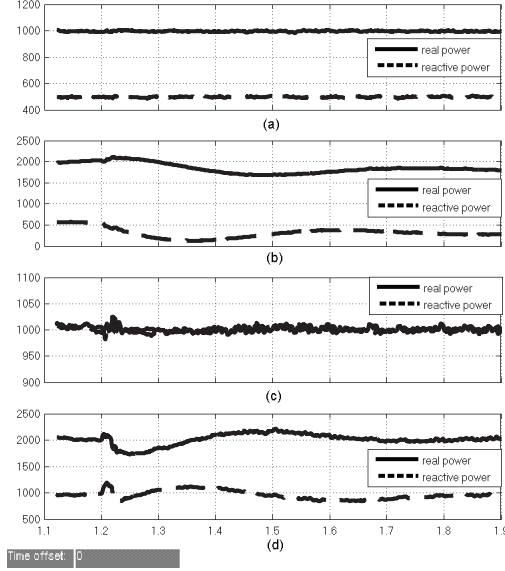


Fig. 13. (a) real and reactive power of DG1. (b) real and reactive power of DG2. (c) real and reactive power of DG3. (d) real and reactive power of DG4.

In Fig. 13, it can be seen that DG2 reduces its power from $2 + j0.5$ to $1.8 + j0.2$ under control of MAS.

In Fig. 14, at 1.2s, load 4 fails to connect to microgrid; Therefore, the power consumed by load 4 becomes zero. Since there are enough power generations within microgrid, load 5 can increase its value and consume at its capacity.

In Fig. 15, at 1.2s, the current output of DG2 reduces under control of MAS.

VI. CONCLUSION

This paper discusses the design and simulation of a decentralized multi-agent system for the operation of microgrid with power electronic interfaces. All agents within the system are identical, so the robustness and flexibility of the system have been tremendously enhanced. Three step communication algorithms enable the system work with least communication data, only real and reactive power mismatch data for neighbor to neighbor communication. A simple control mode can be used for agents to control power electronic interface. Simulation results indicate that this MAS can operate microgrid effectively.

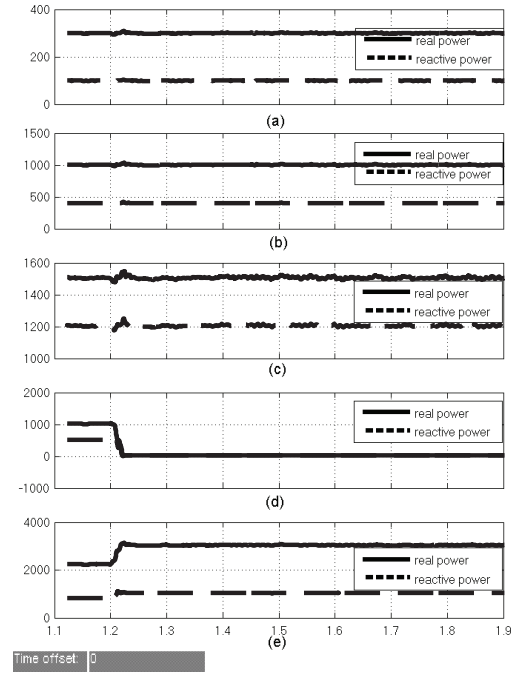


Fig. 14. (a) Real and reactive power of Load 1. (b) real and reactive power of Load 2. (c) real and reactive power of Load 3. (d) real and reactive power of Load 4. (e) real and reactive power of Load 5.

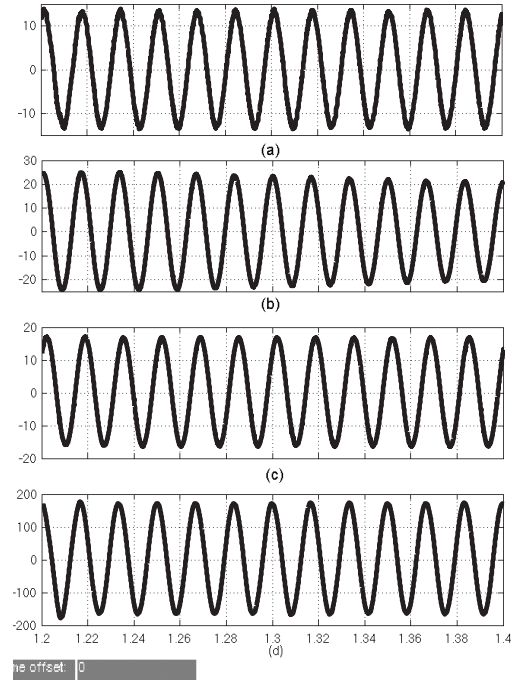


Fig. 15. (a) Current at terminal of DG1. (b) current at terminal of DG2. (c) current at terminal of DG3. (d) voltage at terminal of DG4.

REFERENCES

- [1] US Department of Energy, "International Energy Outlook 2007 (IEO2007)," Report #DOE/EIA-0484, May 2007, available at: <http://www.eia.doe.gov/oiaf/ieo/index.html>.
- [2] S. D. J. McArthur, E. M Davidson and V. M. Catterson, "Multi-agent systems for power engineering applications-part I: Concepts, Approaches, and Technical Challenges," *IEEE Trans. Power Syst.*, vol. 22, no. 4 pp. 1743-1752, 2007.
- [3] S. Suryanarayanan and J. Mitra, "Enabling Technologies for the

- Customer-driven Microgrid,” *IEEE Power & Energy Society General Meeting*, Calgary, AB, July 2009, pp. 1-3.
- [4] Aris L. Dimeas, Nikos D. Hatziaargyriou, “Operation of a Multi-agent System for Microgrid Control”, *Power Systems, IEEE Transaction on*, vol. 20, no. 3, pp. 1447-1455, 2005.
 - [5] Palak Jain, Satish. J. Ranade, “Capacity Discovery in Customer-Driven Microgrids,” *Power Symposium, Proc. of the 41th Annual North American*, Mississippi State University, Oct. 2009.
 - [6] M. Pipattanasomporn, H. Feroze, S. Rahman, “Multi-Agent systems in a distributed smart grid: design and implementation,” *Power Systems Conference and Exposition*, Seattle, WA, March 2009, pp. 1-8.
 - [7] J. Oyarzabal, J. Jimeno, J. Ruela et al., “Agent-based Microgrid Management System,” *Future Power Systems, International Conference on*, Amsterdam, Nov. 2005, pp. 1-6.
 - [8] M. E. Baran and I. M. Markabi, “A multi-agent-based dispatching scheme for distributed generators for voltage support on distribution feeders,” *Power Systems, IEEE Transactions on*, Volume 22, Issue 1, pp. 52-59, 2007.
 - [9] T. Funabashi, T. Tanabe, T. Nagata et al., “An Autonomous Agent for Reliable Operation of Power Market and Systems Including Micro-grids,” *Electric Utility Deregulation and Restructuring and Power Technologies, Third International Conference on*, Nanjing, Apr. 2008, pp. 173-177.
 - [10] Peng, F.Z., YunWei Li, L.M. Tolbert, “Control and Protection of Power Electronics Interfaced Distributed Generation System in a Customer-Driven Microgrid,” *Power & Energy Society General Meeting*, Calgary, AB, July, 2009, pp. 1-8.
 - [11] A. Roshan and R. Burgos, A.C. Baisden et al., “A D-Q frame controller for a full-bridge single phase inverter used in small distributed power generation systems,” in *Proc. IEEE Applied Power Electronics Conference and Exposition - APEC*, Anaheim, CA, Feb. 2007, pp. 641-647.

VII. BIOGRAPHIES

Niannian Cai (S '09) received his B.S. from Huazhong University of Science and Technology. He is currently pursuing a Ph.D. degree in Electrical Engineering at Michigan State University. His doctoral research focuses on microgrid architecture and multi-agent control of microgrids. His research interests include power system planning and control. (email: cainiann@msu.edu)

Joydeep Mitra (S '94, M '97, SM '02) is Associate Professor of Electrical Engineering at Michigan State University, East Lansing. Prior to this, he was Associate Professor at New Mexico State University, Las Cruces, Assistant Professor at North Dakota State University, Fargo, and Senior Consulting Engineer at LCG Consulting, Los Altos, CA. He received a Ph.D. in Electrical Engineering from Texas A&M University, College Station, and a B.Tech. (Hons.), also in Electrical Engineering, from Indian Institute of Technology, Kharagpur. His research interests include power system reliability, distributed energy resources, and power system planning. (email: mitraj@msu.edu)