IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 22, 2017, accepted September 14, 2017, date of publication September 18, 2017,
date of current version October 12, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2753463

Distributed Large-Scale Co-Simulation for IoT-Aided Smart Grid Control

XIN LI, QIUYUAN HUANG, AND DAPENG WU, Fellow, IEEE
Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611, USA
Corresponding author: Dapeng Wu (dpwu@ieee.org)

This work was supported by Grant HDTRA1-14-1-0055.

ABSTRACT An important goal of smart grid is to leverage modern digital communication infrastructure to help control power systems more effectively. As more and more Internet of Things (IoT) devices with measurement and/or control capability are designed and deployed for a more stable and efficient power system, the role of communication network has become more important. To evaluate the performance of control algorithms for inter-dependent power grid and communication network, a test bed that could simulate inter-dependent power grid and communication network is desirable. In this paper, we demonstrate the design and implementation of a novel co-simulator, which would effectively evaluate IoT-aided algorithms for scheduling the jobs of electrical appliances. There are three major features of our co-simulator: 1) large-scale test is achieved by distributed modules that are designed based on a Turing-indistinguishable approach; 2) remote servers or test devices are controlled by local graphical user interface (we only need to configure the simulator on a local server); 3) a software virtual network approach is employed to emulate real networks, which significantly reduces the cost of real-world test beds. To evaluate our co-simulator, two energy consumption scheduling algorithms are implemented. Experimental results show that our co-simulator could effectively evaluate these methods. Thus our co-simulator is a powerful tool for utility companies and policy makers to commission novel IoT devices or methods in future smart grid infrastructure.

INDEX TERMS Smart grid, IoT, real time, large scale experimentation, software virtual network.

I. INTRODUCTION

Smart grid represents a world-wide trend that transforms the traditional power system into a modern one in the information age with communication and control technologies. In the past years, a variety of smart grid technologies such as demand response (DR) technologies with wide area monitoring, protection and control, and new sensing and measurement equipment such as phasor measurement units (PMU) and advanced metering infrastructures (AMI) [1], have been developed to improve the performance and real-time operation of power grids. Furthermore, to make power system more economical and flexible, distributed sources and batteries were introduced to help reduce the distance between production and consumption. Thus DR method or scheduling algorithms could improve the efficiency and stability of power system by helping reduce peak average ratio (PAR) of power consumption and improve grid stability.

The central problem of these methods in a smart grid is the integration of power systems and communication networks because a smart grid requires communication networks to convey measurement data and remote control commands. In a smart grid, PMUs are used to collect phasor data; then phasor data needs to be transferred to Phasor Data Concentrators (PDC) for real-time state estimation and control of power systems. However, many issues remain elusive when considering integrated power grid and communication networks, e.g., it is not clear how device failures cascade between a power grid and a communication network [2]. In addition, before upgrading traditional power grids with new technologies, decision makers need to know how much benefit they will receive from the new technologies, compared to the upgrade cost.

To study the impact of the aforementioned smart grid technologies, a well designed simulator is a much-needed tool. However, developing a new simulator from scratch is complicated, expensive, and time consuming [3], especially for simulating a smart grid, which needs knowledge from both power system engineering and information & communication technology (ICT). Due to simplicity, many researchers seek to develop a co-simulator, which combines existing power grid

TABLE 1. Comparison of existing smart grid co-simulators.

	Power	Communication	Scalability	Real-time	Portability	communication network fidelity
EPOCHS [5]	PSCAD, PSLF	ns-2	Yes	No	No	simulated
GECO [3]	PSLF	ns-2	No	No	No	simulated
ORNL [6]	Adevs	ns-2, OMNeT++	No	No	No	simulated
GridSim [7]	TSTAT	GridStat	No	Yes	No	real
IBCN [8]	Matlab	OMNeT++	No	No	No	simulated
SGiC [9]	OpenDSS, VPP	Black-box model	No	No	No	real with low fidelity
GridSpice [10]	Gridlab-D, MATPOWER	NA	Yes	Yes	Yes	NA
Our Work	Gridlab-D	CORE	Yes	Yes	Yes	real with high fidelity

and communication network simulators into a smart grid simulator [3]. Commonly used power system simulators include PSCAD/EMTDC, DigSilent Power Factory, Siemens PSS, EMTP-RV, PowerWorld, ETAP PSMS, Cymdist, EuroStag, Homer, OpenDss, ObjectStab, and Gridlab-D; and widely used communication network simulators include ns-2 and ns-3, OMNeT++, Nessi, OPNET. However, the main challenge of a smart grid co-simulator is to connect, handle and synchronize data and interactions between a power grid simulator and a communication network simulator [4].

Different from existing co-simulators, in this paper, we develop a co-simulator, based on Gridlab-D simulator and Common Open Research Emulator (CORE). Our co-simulator addresses the challenge of synchronization and interaction between Gridlab-D and CORE, by utilizing 1) a Graphic User Interface (GUI) to provide high efficiency, 2) a software emulation approach to achieve high fidelity, and 3) an Ethernet-tunnel-based distributed module to achieve scalability. To conduct experiments, we can directly run Linux applications on any virtual node by configuring the GUI of our co-simulator, since virtual nodes are based on a light-weight virtualization technique supported by mainstream Linux kernel. Since the computing capability of a single host machine is limited, a large-scale experiment can be done by allocating virtual nodes to multiple host machines, which are coordinated by distributed modules. Furthermore, both real-time and non-realtime mode are supported in our co-simulator to serve different kinds of experiment needs. To evaluate the effectiveness and capability of our co-simulator, we implement scheduling algorithms and conduct experiments for two test cases. In Test Case 1, we implement a scheduling algorithm with real-time pricing information, which helps verify the real-time mode of our co-simulator; and in Test Case 2, we implement a day-ahead job scheduling algorithm on a large number of virtual nodes, which proves the feasibility of our co-simulator for large-scale experiments.

The rest of this paper is organized as follows. Section II describes the related work. Section III presents our co-simulator architecture and the implementation of each module based on Gridlab-D and CORE. Section IV describes Test Case 1 to verify the real-time mode of our co-simulator. Section V describes Test Case 2 to demonstrate the effectiveness of our co-simulator in large-scale experiments. Finally, Section VI concludes the paper.

II. RELATED WORK

To study the impact of new technologies on a smart grid, researchers tend to use a software-assisted platform (at least in the initial phase of the study) due to its low cost. According to the structure of a platform, existing software-assisted platforms for smart grids can be categorized into three types: hardware-in-the-loop (HITL) platforms, software simulators and software emulators. A hardware-in-the-loop platform consists of a hardware system (to be tested), electrical emulation of sensors and actuators, and a software-based controller. The sensors and actuators act as the interface between the controller and the hardware system under test. An HITL approach can achieve high fidelity, but it is costly for large-scale experiments under the HITL approach. On the other hand, software based platforms such as software simulators and software emulators achieve better scalability and lower cost. Co-simulation, which utilizes existing stable and extensible simulators of heterogeneous systems, has become a popular and efficient way to integrates power system models and ICTs. We summarizes existing co-simulators for smart grids and their performance in Table 1 where ''portability'' means that the system can be implemented on any operating system such as Linux, MS Windows, and Apple iOS; under ''communication network fidelity'', ''simulated'' means a fidelity level achieved software simulation, and ''real'' means that the communication system is capable of interacting with the real systems.

EPOCHS [5] ,ORNL [6], GECO [3] and GridSim [7] adopt a co-simulation approach to test various scenarios for wide-area monitoring, protection and control. EPOCHS is an agent-based electric power and communication co-simulator. EPOCHS consists of three major components, namely, transient-time-scale component (PSCAD/EMTDC), power system component (PSLF) and computer network component (ns-2). With the runtime infrastructure (RTI) module, EPOCHS is able to achieve distributed simulation, which enables its capability of large scale simulation. Similar to EPOCHS, the ORNL platform [6] is based on discrete event system specification (DEVS) to ensure correct simulation of integrated system. The GECO platform [3] utilizes ns-2 (which simulates a communication network) and PSLF (which simulates a dynamic power system); in GECO, a global event driven mechanism is employed in order to overcome the accuracy problems in simulator synchronization. The Gridsim platform [7] is a real-time co-simulator with

integrated power grid, communication network and control system. It operates in real-time to ensure that it can interact with real power system components. Compared to the aforementioned platforms, our co-simulator uses software virtualization to achieve large-scale experiments and high fidelity; it uses real communication devices for communication, instead of communication simulators.

IBCN [8], SGiC [9] and GridSpice [10] are co-simulators that focus on demand response or demand side management test. The IBCN co-simulator consists of OMNet++ using the INET framework and a power system simulator implemented in Matlab; in IBCN, high level applications and services make use of the middleware layer, which is designed to provide generic functionality for various services. SGiC [9] is a web-based software platform with social network functionality to encourage consumers to participate in demand response and demand side management tests; in SGiC, all decisions and power system simulations are performed by OpenDSS. GridSpice [10] is a cloud-based co-simulator built on top of Gridlab-D and MATPOWER. However, it lacks a communication network component. In contrast, our co-simulator does not need specific middleware. So it is easy to port or migrate our co-simulator to a different operating system.

III. SYSTEM DESIGN AND IMPLEMENTATION

A. GRIDLAB-D

Gridlab-D is an open-source power system simulator that can be integrated with third-party data management and analysis tools [11]. Different from the traditional difference-based simulators, Gridlab-D implements Newton-Raphson algorithm and the Back-Forward-Sweep method on an unbalanced distributed power flow model with much higher accuracy. Furthermore, Gridlab-D supports power systems with millions of independent devices. The software also handles a wide range of time scales from seconds to many years. Gridlab-D has both Linux version and MS Windows version. We implement our testbed based on the Linux version of Gridlab-D version 3.1.

Agent-based and information-based modeling tools make Gridlab-D quite extensible. On one hand, the software allows users to create models of novel end-user appliances, distributed energy resources, distribution, distribution automation corresponding to environmental data (such as weather data, market price data and power consumption profile). On the other hand, it supports a server mode, under which other software is able to get device status (such as voltage and current), or to configure the property values (such as switch on-off and market price) as the simulation progresses.

B. CORE

To equip power devices in Gridlab-D with communication capability, we use the Common Open Research Emulator (CORE) platform [12]. CORE is a highly customizable tool, which supports efficient distributed emulation. Under CORE, any user-designed application is allowed to run on a virtual node, which provides a convenient way to verify algorithms of smart-grid devices. Furthermore, these applications could be easily portable to real Linux based devices without modification. In addition, in a distributed emulation mode, multiple emulation servers can work together and be controlled by a single GUI.

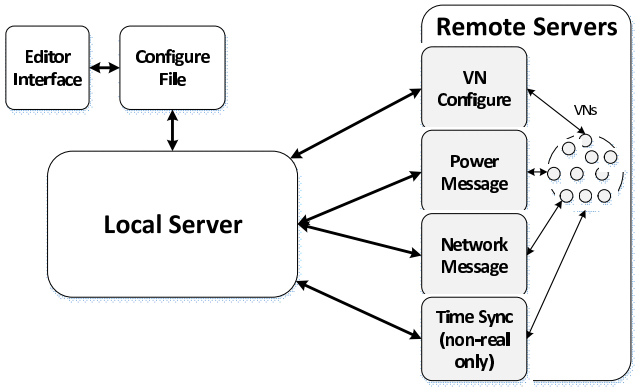

FIGURE 1. Architecture of our co-simulator.

C. DESIGN AND IMPLEMENTATION

Our co-simulator combines the Gridlab-D power distribution system simulator and the communication network emulator CORE. Fig. 1 illustrates the architecture of our co-simulator with one local server and one remote server. Since the maximum number of VNs running on each remote server is limited to a certain number, say, 300, multiple remote servers need to be used if the number of VNs to be simulated is more than 300. The whole system for co-simulation consists of a user interface, configuration files, a power simulator, a communication network emulator, a distributed simulation module, and a timer synchronization module. The local server coordinates all operations among users and simulators.

1) USER INTERFACE

The user interface allows users to visually create both power grid and corresponding communication network topology as shown in Fig. 2. It is developed based on the GUI (graphical user interface) of CORE, which is convenient for users to draw nodes and network devices on the canvas [13]. Power grid devices (such as transformers, regulators, power lines, switches and loads) and their configurations/parameters (such as nominal voltage, real power, reactive power, and appliances in each residential house) are described in Tcl/Tk 8.5. In addition to a graph editing tool, the user interface also provides tools for setting simulation pairs, distributed simulation servers and time synchronization modes.

2) CONFIGURATION FILES

There are three types of configuration files, which can be created by editor tools on a local server: GLM files (GridLab-D Model files), CORE scenario files, and

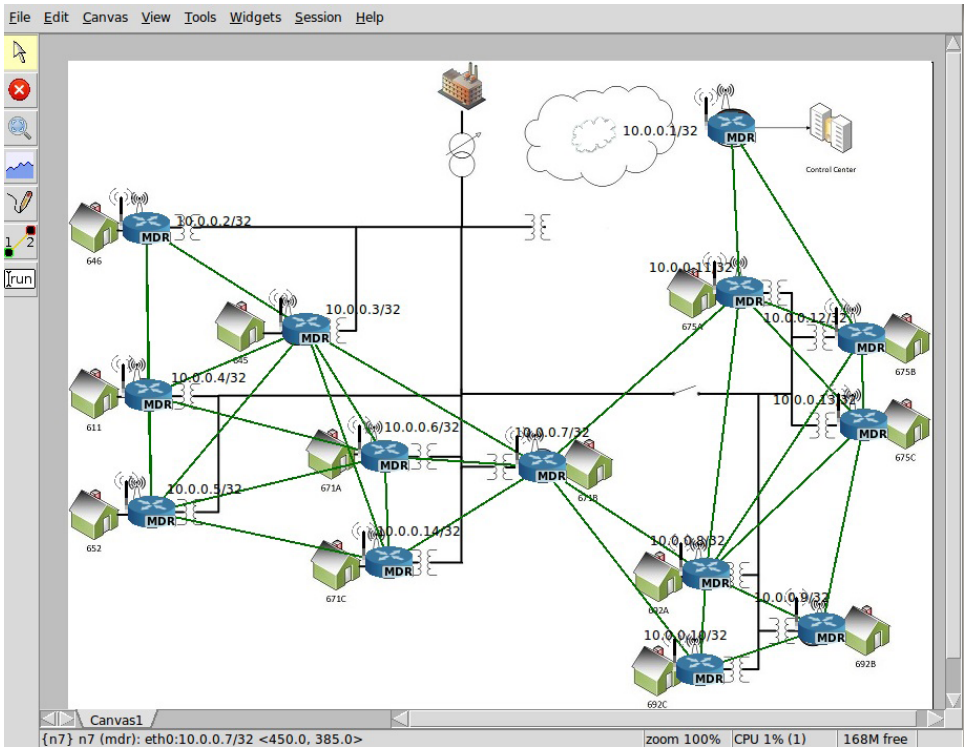

FIGURE 2. Editor in user interface.

simulation-pairs configure files. GLM files are used to set up the parameters for the power flow calculation algorithm, power-grid device models, energy consumption scheduling algorithms, results recording, and weather data. CORE scenario files record the communication network topology and device configuration. Simulation-pairs configure files contain identities of the corresponding power devices and communication devices pair; meanwhile, they record the path of each application running on the application layer of communication devices.

3) SOFTWARE VIRTUAL NETWORK (SVN) APPROACH

To significantly reduce the cost of real-world testbeds, our co-simulator employs a Software Virtual Network (SVN) approach to emulate real communication networks. We take a Turing-indistinguishable approach developed in [14] in the design of software virtual networks. For details, refer to [14].

4) CORRESPONDENCE MODULE

To extend the distributed emulation capability of CORE to our co-simulator, we design an interface that provides message passing between each simulation pair that consists of an object in Gridlab-D and its corresponding virtual node in CORE. This interface regularly acquires status messages from an object in Gridlab-D (such as a power meter, a switch, a transformer), and then delivers messages to the corresponding virtual node. Meanwhile, each virtual node can also send control messages from the application layer to its corresponding object in Gridlab-D so as to set the properties/attributes of an object in Gridlab-D through the interface. Each interface is dynamically created between a communication pair as shown in Fig. 3.

As shown in Fig. 3, after a simulation begins, the correspondence module loads simulation-pair data from configuration files, in order to register these pairs. Once timer t decrease from T to 0, the correspondence module creates an interface between each power grid object and its corresponding virtual node, which updates the property of its corresponding power grid object periodically. Constrained by the minimal time interval of 1s in Gridlab-D [15], the length of a time slot T should be no larger than 1s. Within one time slot, incoming control messages from virtual nodes will be stored until the timer times up.

Messages between VNs (which might be on different servers) are transmitted with User Datagram Protocol (UDP) on the transport layer. This results in a smaller round-trip communication time between virtual nodes. The delay is typically less than 0.6ms; hence, message transfer delay is considered to be negligible.

5) DISTRIBUTED SIMULATION MODULE

Since the maximum number of virtual nodes that can run on one single computer is limited (say, N_v is the maximum number of virtual nodes), we need to run CORE on multiple computers if the number of virtual nodes is more than

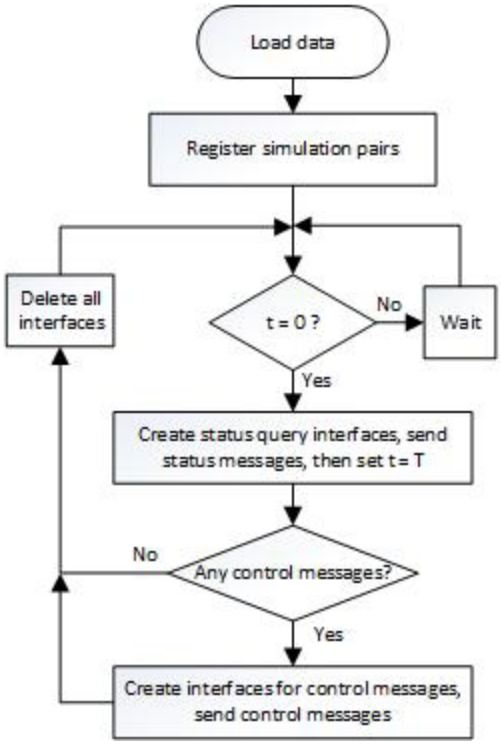

FIGURE 3. Correspondence module dynamically creates process.

N_v. We take a Turing-indistinguishable approach developed in [14] in the design of the distributed simulation module. For details, refer to [14]. To connect virtual nodes on multiple computers, we set up RJ45 tool or a tunnel tool, i.e., virtual nodes on different computers are physically linked by cable through RJ-45.

After loading data from configuration files, the IP addresses and port numbers of remote servers would be used as parameters to create tunnels between a local server and remote servers. Then distributed simulation module transfers SVN topology and configuration data to initialize virtual nodes on remote servers that host these virtual nodes. Each virtual node is able to run applications on its host remote server as an individual communication network device.

6) TIMER SYNCHRONIZATION MODULE

Two modes are supported in our co-simulator: real-time mode and non-real-time mode. In the real-time mode, the simulation clock in Gridlab-D and the emulation clock in CORE are synchronized by a real-time global clock. In the non-real-time mode, a modified barrier synchronization algorithm [16] is used in the timer synchronization module.

IV. TEST CASE 1: VERIFICATION OF REAL-TIME MODE OF CO-SIMULATOR

To demonstrate that our co-simulator is capable of integrating communication network and power grid and providing performance measures for algorithms used in smart grids in real-time mode, we implement a real-time price based distributed power consumption optimization algorithm on our co-simulator for a group of households that are equipped with smart appliances. In this test case (Test Case 1), the smart grid infrastructure is based on Advanced Metering Infrastructure (AMI) network and IEEE PES 37 bus distribution system test feeder as shown in Fig. 4.

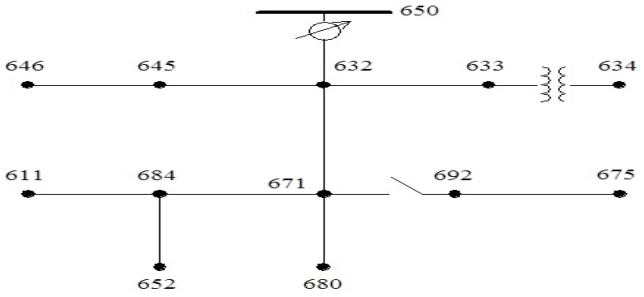

FIGURE 4. One-line diagram of the IEEE 13 node feeder in [17].

A. EXPERIMENTAL SETUP

1) IEEE 13 NODE TEST FEEDER

The IEEE 13 node test feeder is a well studied feeder that provides different types of common distribution system elements. It contains one wye connected substation voltage regulator, shunt capacitor banks, in-line transformer, unbalanced split, and lines with a variety of phasing [17]. The Gridlab-D team had implemented this feeder on Gridlab-D. To simulate energy consumption in a small community, the IEEE 13 node test feeder would be a good choice.

2) LOADS AND APPLIANCES

In Test Case 1, the power system consists of 13 households that locate under nodes as shown in Fig. 4. The 13 households are given identification number as H611, H645, H646, H652, H671A, H671B, H671C, H675A, H675B, H675C, H692A, H692B and H692C (where A, B, C are the phases on distribution grid). Each household is equipped with a smart meter running scheduling algorithms that control switches of all smart appliances. These switches are called demand response switches (DRS), which are already available in the market. Algorithm 1 (scheduling) and DRS are implemented in the Gridlab-D part of our co-simulator. Each household has both non-deferrable electrical appliances (such as water heaters, lighting devices, and refrigerators) and deferrable electrical appliances (such as air conditioners); a deferrable electrical appliance can defer its usage while a non-deferrable electrical appliance cannot.

Algorithm 1 is used to schedule deferrable electrical appliances. In each time slot t, Algorithm 1 determines whether a given deferrable electrical appliance should be switched on or not. The decision made by Algorithm 1 will be conveyed to the DRS of the corresponding appliance and the DRS will switch on or switch off the appliance accordingly.

3) AMI NETWORK

An AMI network provides communications and interactions between a utility control center and residential households.

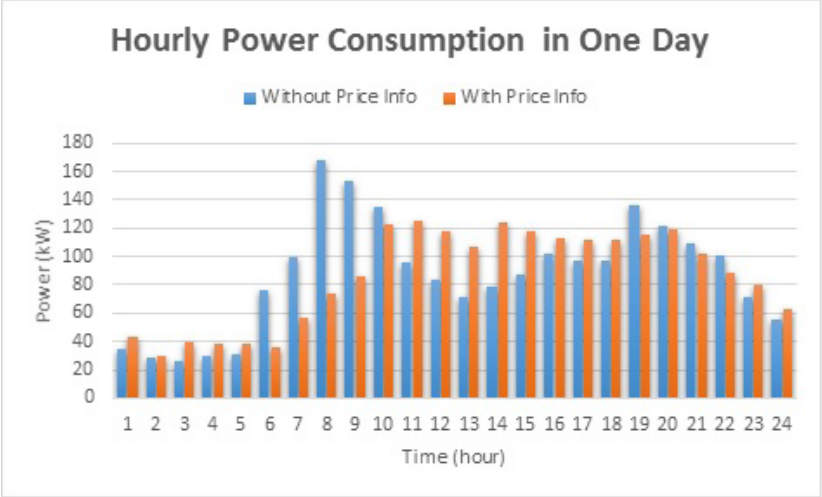

FIGURE 5. Hourly total power consumption for 13 households in one day.

Algorithm 1 Scheduling Deferrable Electrical Appliances

Input: Real-time energy price P_t in current time slot t; temperature T_t in current time slot t; maximum number N_p of prices $\{P_i\}$ stored in the price history buffer.
Output: DRS status S_t in current time slot t.

Calculate temperature difference ΔT_t between current temperature T_t and desired temperature T_d, i.e., $\Delta T_t = T_t - T_d$;
if $t \leq N_p$ **then**
  Calculate average price $P_{avg} = \frac{1}{t}\sum_{i=1}^{t} P_i$;
else
  $P_{avg} = \frac{1}{N_p}\sum_{i=t-N_p+1}^{t} P_i$;
Obtain an acceptable price P_{accept} by $P_{accept} = f(\Delta T_t, P_{avg})$ where $f(\cdot)$ is a user-specified function;
if Real-time price $P_t \leq P_{accept}$ **then**
  Set DRS status $S_t = ON$;
else
  Set DRS status $S_t = OFF$;
if $t > N_p$ **then**
  Delete P_{t-N_p} from the price history buffer;
Insert price P_t into the price history buffer.

The AMI network consists of all communication nodes in a smart grid, including smart meters with communication capability and relay routers. In each time slot, the utility control center gathers energy consumption data from smart meters in each household. Then a control program in the utility control center calculates a real-time energy price using the following quadratic pricing model [18] in Eq. (1) every five minutes:

$$C_t = r \times (L_t)^2, \tag{1}$$

where C_t is the energy price in time slot t, L_t is the total energy-consumption in slot t, and r is a rate and we set $r = 0.02$. Meanwhile, the utility control center also generates real-time bills for each household. Once a real-time price is calculated, the utility control center broadcasts the price information to smart meters in each household.

4) WEATHER DATA

In Test Case 1, we use real weather data in Gainesville, Florida from [19]. Hourly local temperature is used to provide parameters for the household model in Gridlab-D.

B. EXPERIMENTAL RESULTS

We emulate Test Case 1 in two scenarios: one with price information for smart meters, and one without price information for smart meters. For Test Case 1, we run our co-simulator under the real-time mode with a single computer. Under the scenario with price information for smart meters, the utility control center gathers power consumption data from smart meters in the power grid, and then broadcasts real-time energy prices to smart meters in each household. Under the scenario without price information for smart meters, we switch off the corresponding modules in our co-simulator, and thus the devices in the power grid will not receive price information.

Fig. 5 shows hourly total power consumption of 13 households in a span of 24 hours for the two scenarios (with and without price information). Fig. 6 shows hourly total bill payment of 13 households in a span of 24 hours for the two scenarios (with and without price information). As shown in Figs. 5 and 6, when smart meters receive no price information from the utility control center, the PAR (peak average ratio) is 1.93, and the total bill payment is $4,353.85 under the price model described in Eq. (1); when real-time prices are received by smart meters in each household, the PAR reduces to 1.46 (which is 32.2% less than the scenario without pricing information), while the total energy consumption keeps

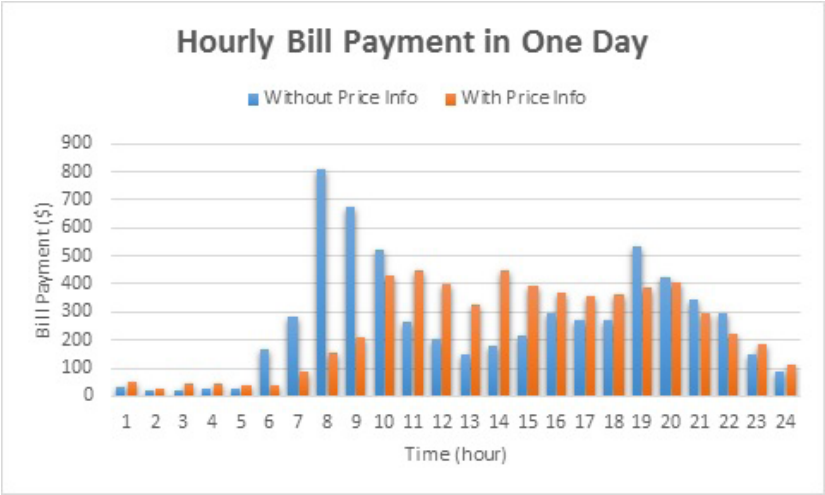

FIGURE 6. Hourly total bill payment for 13 households in one day.

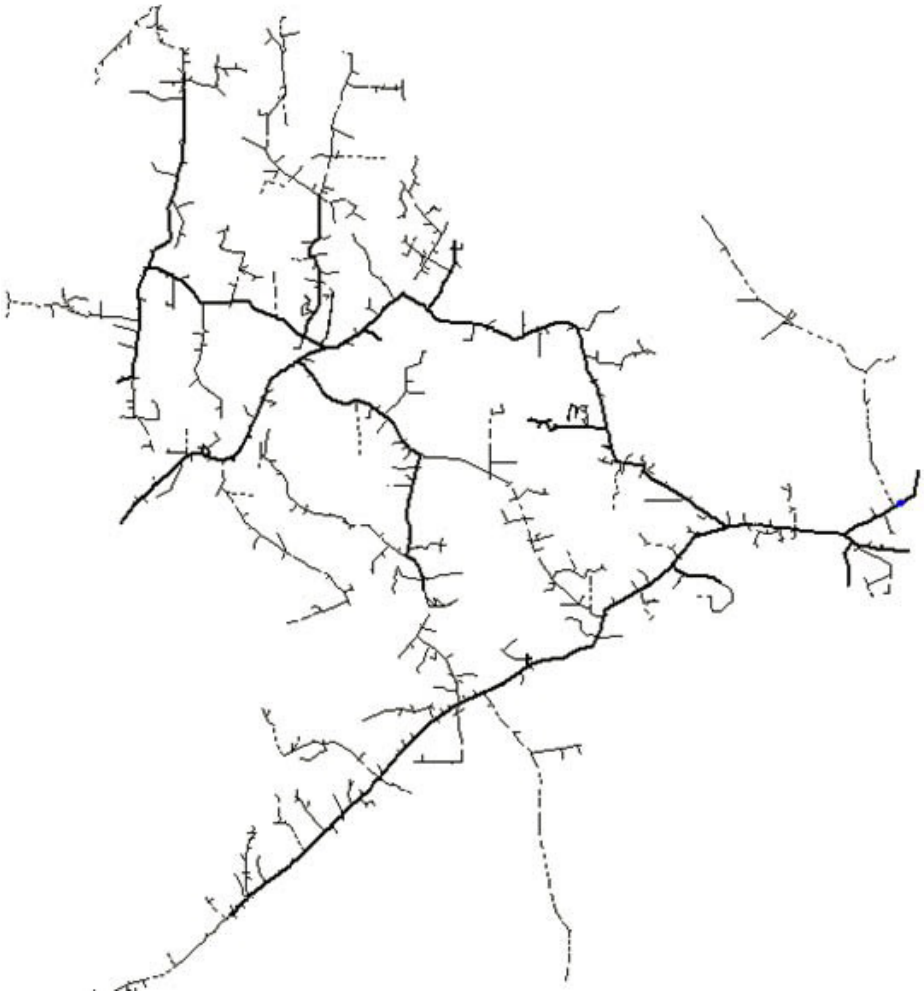

FIGURE 7. One-line diagram of the IEEE 8500 node feeder [20].

unchange, and the total bill payment reduces to $4,063.06 (which is 7.2% less than the scenario without pricing information).

Test Case 1 demonstrates that our co-simulator is capable of simulating scheduling algorithms in real-time mode for a smart grid so that utility companies can obtain

valuable insights before full-fledged scheduling algorithms are deployed in a real smart grid.

V. TEST CASE 2: VERIFICATION OF LARGE-SCALE SIMULATION CAPABILITY OF CO-SIMULATOR

To demonstrate the scalability of our co-simulator, in Test Case 2, we simulate the IEEE 8500 node test feeder [20], including 1000 household loads. Since the maximum number of communication virtual nodes (VNs) running on one server is limited (usually a few hundred nodes), we allocate 1000 communication nodes to four servers, each of which employs a distributed simulation module of our co-simulator. We implement a social-network based scheduling algorithm described in [21] for this large-scale test case.

A. EXPERIMENTAL SETUP

1) IEEE 8500 NODE TEST FEEDER

The IEEE 8500 node test feeder is derived from a real radial distribution circuit in the US [20]. Its topology is shown in Fig. 7. It was developed by the Test Feeder Working Group (WG) of the Distribution System Analysis Subcommittee of Technical Committee on Power Systems Analysis, Computing , and Economics (PSACE), IEEE Power Engineering Society. It contains Medium Voltage (MV) level, Low Voltage (LV) level, and their connecting transformers [20]. It is a moderately large circuit that consists of 4800 1-, 2-, and 3-phase buses. Instead of these balance and imbalance buses, the test feeder also contains substation transformers and regulators, three sets of regulators with output voltage control, three actively switched capacitor banks and one static capacitor bank. Voltage override is also characterized in each controlled capacitor, which turns ON at 0.9875pu and turns OFF at 1.075pu. These characteristics are very important for daily and annual simulations. Test Case 2 is based on the IEEE 8500 test feeder model, which has been validated by the Gridlab-D team previously.

2) LARGE-SCALE SCENARIO

To evaluate the performance of the scheduling algorithm in [21],we consider 1000 households with both deferrable and non-deferrable demands. First, we use the LFR network generator [22] to create a small-world scale-free network of 1000 nodes, which represent 1000 households. We use 4 servers to run our co-simulator, each of which holds 250 virtual nodes out of the 1000 nodes. We use the minimum-variance grouping algorithm proposed in [21] to partition the 1000 nodes into groups, each of which has no more than five nodes, i.e., the maximum group size $N_c = 5$. Electrical appliances of the users in the same group will be scheduled together to reduce payment, similar to the family plan in wireless cell phone communication. Each node (or each household) is assigned with an ID, which also serve as the public encryption key for communication between the nodes in the same group. Each node (or each household) exchanges its demand information with other nodes in the same group; the demand messages (containing the information of electrical appliances that request for scheduling jobs in the next 24 hours from 00:00 to 23:59) are encrypted by public-key cryptography for the purpose of privacy protection. Then each node runs the distributed Earliest Deadline First (EDF) scheduling algorithm proposed in [21] and obtains the job schedule, which indicates the start time of each electrical appliance that requests for scheduling; these electrical appliances will be switched on at the specified time on our co-simulator.

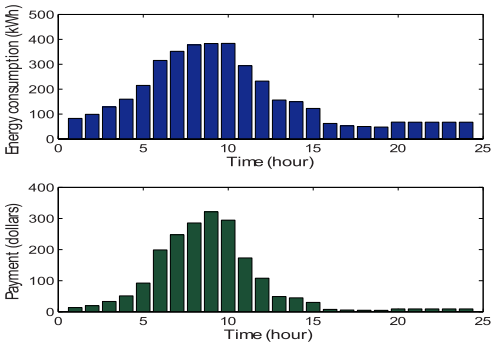

FIGURE 8. Energy consumption and payment under random scheduling.

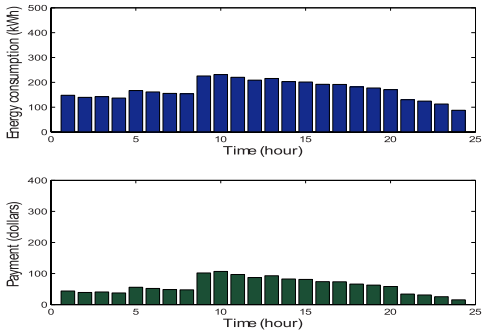

FIGURE 9. Energy consumption and payment under the EDF scheduling.

B. EXPERIMENTAL RESULTS

In this section, we evaluate the performance of the distributed mode of our co-simulator, i.e., running multiple servers in a distributed fashion. We implement the distributed Earliest Deadline First (EDF) scheduling algorithm proposed in [21] on our co-simulator. To demonstrate the effectiveness of the EDF scheduler, we use a random scheduling algorithm as a benchmark; for each deferrable electrical appliance, the random scheduling algorithm randomly selects a start time between the earliest start time and the latest start time that are specified by the user. As in Section IV, here we use the same pricing model as in Eq. (1) to calculate the user payment.

Fig. 8 shows energy consumption and user payment as a function of time, under random scheduling. Fig. 9 shows energy consumption and user payment as a function of time, under the EDF scheduling. As shown in Fig. 8, when

the random scheduling is used, the PAR for 1000 households is 1.99 and the total user payment is \$2,321. In contrast, when the EDF scheduling is used, the PAR reduces to 1.34 (48.5% less) and the total user payment reduces to \$1,234.34 (88% less). In addition, when the EDF scheduling is used, the load becomes much smoother, compared to the random scheduling.

In summary, the results of Test Case 2 demonstrate that our co-simulator is able to simulate the effect of the EDF scheduling algorithm under a distributed large-scale scenario. In addition, the quantitative simulation results obtained by our co-simulator can help utility companies or policy makers to gain insights and performance metrics of demand response algorithms before the algorithms are used in practice.

VI. CONCLUSIONS

In this paper, we presented a novel distributed large-scale co-simulator for verification and performance evaluation of algorithms used in smart grid infrastructure. Our co-simulator combines a power grid simulator with a communication network emulator. Meanwhile, it is able to work under a distributed mode, which supports large-scale experimentation. To evaluate our co-simulator, two cases were tested. Test Case 1 has shown that our co-simulator is able to test the performance of scheduling algorithms in real-time mode for a smart grid so that utility companies can obtain important experience and fine-tune the algorithms before the algorithms are deployed in practice. Test Case 2 demonstrated the capability of our co-simulator in simulating algorithms for a large number of households using a distributed mode. Furthermore, the scheduling algorithms implemented on our co-simulator under Linux can easily migrate to other operating systems.

REFERENCES

[1] J. C. Fuller, S. Ciraci, J. A. Daily, A. R. Fisher, and M. Hauer, "Communication simulations for power system applications," in *Proc. Workshop Modeling Simulation Cyber-Phys. Energy Syst. (MSCPES)*, May 2013, pp. 1–6.

[2] *National Power Grid Simulator Workshop Report*, U.D. Homeland Secur., Washington, DC, USA, Dec. 2008.

[3] H. Lin, S. S. Veda, S. S. Shukla, L. Mili, and J. Thorp, "GECO: Global event-driven co-simulation framework for interconnected power system and communication network," *IEEE Trans. Smart Grid*, vol. 3, no. 3, pp. 1444–1456, Sep. 2012.

[4] K. Mets, J. A. Ojea, and C. Develder, "Combining power and communication network simulation for cost-effective smart grid analysis," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 3, pp. 1771–1796, 3rd Quart., 2014.

[5] K. Hopkinson, X. Wang, R. Giovanini, J. Thorp, K. Birman, and D. Coury, "EPOCHS: A platform for agent-based electric power and communication simulation built from commercial off-the-shelf components," *IEEE Trans. Power Syst.*, vol. 21, no. 2, pp. 548–558, May 2006.

[6] J. Nutaro, "Designing power system simulators for the smart grid: Combining controls, communications, and electro-mechanical dynamics," in *Proc. IEEE Power Energy Soc. General Meeting*, Jul. 2011, pp. 1–5.

[7] R. Buyya and M. Murshed, "GridSim: A toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing," *Concurrency Comput., Pract. Exper.*, vol. 14, nos. 13–15, pp. 1175–1220, Nov./Dec. 2002.

[8] K. Mets, T. Verschueren, C. Develder, T. L. Vandoorn, and L. Vandevelde, "Integrated simulation of power and communication networks for smart grid applications," in *Proc. IEEE 16th Int. Workshop Comput. Aided Modeling Design Commun. Links Netw. (CAMAD)*, Jun. 2011, pp. 61–65.

[9] J. E. S. de Haan, P. H. Nguyen, W. L. Kling, and P. F. Ribeiro, "Social interaction interface for performance analysis of smart grids," in *Proc. IEEE 1st Int. Workshop Smart Grid Modeling Simulation (SGMS)*, Oct. 2011, pp. 79–83.

[10] K. Anderson and A. Narayan, "Simulating integrated volt/var control and distributed demand response using GridSpice," in *Proc. IEEE 1st Int. Workshop Smart Grid Modeling Simulation (SGMS)*, Oct. 2011, pp. 84–89.

[11] *GridLAB-D Web Site*. Accessed: Aug. 15, 2017. [Online]. Available: http://www.gridlabd.org/

[12] J. Ahrenholz, C. Danilov, T. R. Henderson, and J. H. Kim, "CORE: A real-time network emulator," in *Proc. IEEE Military Commun. Conf. (MILCOM)*, Nov. 2008, pp. 1–7.

[13] *CORE Simulator Web Site*. Accessed: Jul. 1, 2016. [Online]. Available: http://downloads.pf.itd.nrl.navy.mil/docs/core/core-html/usage.html

[14] J. Kong, T. Li, and D. O. Wu, "RVIP: An indistinguishable approach to scalable network simulation at real time," *Wireless Commun. Mobile Comput.*, vol. 15, no. 18, pp. 2219–2232, Dec. 2015.

[15] B. Palmintier, B. Lundstrom, S. Chakraborty, T. Williams, K. Schneider, and D. Chassin, "A power hardware-in-the-loop platform with remote distribution circuit cosimulation," *IEEE Trans. Ind. Electron.*, vol. 62, no. 4, pp. 2236–2245, Apr. 2015.

[16] S. Tan, W.-Z. Song, S. Yothment, J. Yang, and L. Tong, "ScorePlus: An integrated scalable cyber-physical experiment environment for smart grid," in *Proc. IEEE Int. Conf. Sens., Commun. Netw. (SECON)*, 2015.

[17] W. H. Kersting, "Radial distribution test feeders," in *Proc. IEEE Power Eng. Soc. Winter Meeting*, vol. 2. Jan./Feb. 2001, pp. 908–912.

[18] A.-H. Mohsenian-Rad, V. W. S. Wong, J. Jatskevich, and R. Schober, "Optimal and autonomous incentive-based energy consumption scheduling algorithm for smart grid," in *Proc. Innov. Smart Grid Technol. (ISGT)*, Jan. 2010, pp. 1–6.

[19] *Accuweather Web Site*. Accessed: Apr. 15, 2016. [Online]. Available: http://www.accuweather.com/

[20] R. F. Arritt and R. C. Dugan, "The IEEE 8500-node test feeder," in *Proc. IEEE Transmiss. Distrib. Conf. Expo.*, Apr. 2010, pp. 1–6.

[21] Q. Huang, X. Li, J. Zhao, D. Wu, and X.-Y. Li, "Social networking reduces peak power consumption in smart grid," *IEEE Trans. Smart Grid*, vol. 6, no. 3, pp. 1403–1413, May 2015.

[22] A. Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 78, no. 4, p. 046110, Oct. 2008.

XIN LI received the B.E. degree in electrical engineering from Zhejiang University, Zhejiang, China, in 2009, the M.E. degree in electrical engineering from Zhejiang University, Zhejiang, China, in 2012, and the Ph.D. degree in electrical and computer engineering from the University of Florida, Gainesville, FL, USA, in 2016. His research interests are in the areas of networking and smart grid.

QIUYUAN HUANG received the B.S. degree in computer science from the University of Science and Technology of China, Hefei, China, in 2011, and the Ph.D. degree in electrical and computer engineering, University of Florida, Gainesville, FL, USA, in 2017. Her research interests are networking, smart grid, cyberphysical systems, machine learning, and security.

DAPENG WU (S'98–M'04–SM'06–F'13) received the B.E. degree in electrical engineering from the Huazhong University of Science and Technology, Wuhan, China, in 1990, the M.E. degree in electrical engineering from the Beijing University of Posts and Telecommunications, Beijing, China, in 1997, and the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, USA, in 2003.

He is currently the Professor with the Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL, USA. His research interests are in the areas of networking, communications, signal processing, computer vision, machine learning, smart grid, and information and network security.

Dr.WU received the University of Florida Term Professorship Award in 2017, University of Florida Research Foundation Professorship Award in 2009, AFOSR Young Investigator Program (YIP) Award in 2009, ONR YIP Award in 2008, NSF CAREER award in 2007, the IEEE Circuits and Systems for Video Technology Transactions Best Paper Award for Year 2001, and the Best Paper Awards in the IEEE GLOBECOM 2011 and International Conference on Quality of Service in Heterogeneous Wired/Wireless Networks (QShine) 2006. He has served as a Technical Program Committee (TPC) Chair for the IEEE INFOCOM 2012, and TPC Chair for the IEEE International Conference on Communications (ICC 2008), Signal Processing for Communications Symposium, and as a member of executive committee and/or technical program committee of over 80 conferences. He was elected as a Distinguished Lecturer by the IEEE Vehicular Technology Society in 2016. He has served as Chair for the Award Committee, and Chair of Mobile and wireless multimedia Interest Group (MobIG), Technical Committee on Multimedia Communications, the IEEE Communications Society. He was an elected member of Multimedia Signal Processing Technical Committee, the IEEE Signal Processing Society from 2009 to 2012. He currently serves as an Editor in Chief of the IEEE Transactions on Network Science and Engineering, and an Associate Editor of the IEEE Transactions on Communications, the IEEE Transactions on Signal and Information Processing Over Networks, and the IEEE Signal Processing Magazine. He was the Founding Editor-in-Chief of *Journal of Advances in Multimedia* between 2006 and 2008, and an Associate Editor for the IEEE Transactions on Circuits and Systems for Video Technology, the IEEE Transactions on Wireless Communications and IEEE Transactions on Vehicular Technology. He is also a Guest-Editor for the IEEE Journal on Selected Areas in Communications, special issue on cross-layer optimized wireless multimedia communications.