Probabilistic Model Checking and Scheduling Implementation of an Energy Router System in Energy Internet for Green Cities

Min Gao, Kun Wang, *Senior Member, IEEE*, and Lei He, *Senior Member, IEEE*

***Abstract*—Energy router (ER) based system is a crucial part of the energy transmission and management under the circumstance of energy Internet for green cities. During its design process, a sound formal verification and a performance monitoring scheme are needed to check its reliability and meaningful quantitative properties. In this paper, we provide formal verification solutions for an ER-based system by proposing a continuous-time Markov chain model describing the architecture of the ER-based system. To verify real-world function of the ER-based system, we choose electricity trading to propose a Markov decision process model based on an ER subsystem to describe the trading behavior. To monitor the system performance, we project the energy scheduling process in the ER-based system, and then implement this scheduling process on top of a cloud computing experiment tool. Finally, we perform extensive experiment evaluations to investigate the system reliability properties, quantitative properties, and scheduling behaviors. The experiment verifies the effectiveness of the proposed models and the monitoring scheme.**

***Index Terms*—Energy Internet (EI), energy router (ER), formal verification, green cities, model checking.**

I. Introduction

ENERGY crisis and carbon emission have become two seriously concerned issues in green cities [1]–[3] recently. As a feasible solution, energy Internet (EI) [4]–[6] has aroused global concern since it has been proposed. EI is a new power generation developing a green vision of evolution of smart grids into the Internet. Its organization is shown in Fig. 1. The key device to compromise EI is an energy router (ER) [7]. ER communicates with users similar to an Internet router, thus can perform immediate communication and control according to real-time user status to achieve green efficient energy management. This is vital to realizing green cities [8], [9].

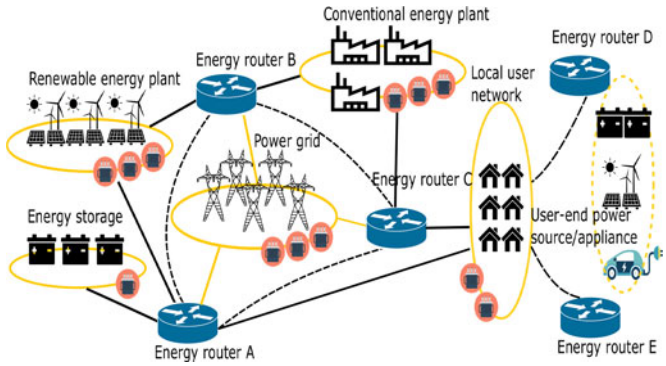


Fig. 1. ER system in energy Internet for green cities.

A variety of research works have been done on the ER and green city topics. He *et al.* [10] propose a multitier fog computing model with large-scale data analytics service for smart cities applications. Wang *et al.* [11] propose a system-level stability evaluation model in the EI based on a critical energy function to explore small disturbance stability region. Zhong *et al.* [12] present a demand response model of vehicle-to-grid mobile energy network in which the EVs generally move across different districts represented as network nodes. Zhong *et al.* [6] provide a survey to introduce EI communication for sustainability. Zhang *et al.* [13] introduce architecture, standards, and QoS improvement for home machine-to-machine networks. Wang *et al.* [11] evaluate EI system's stability using big data analytics. Wang *et al.* [14] propose a big data computing architecture for smart grid analytics, which involves data resources, transmission, storage, and analysis. Xin *et al.* [15] propose design and applications of the ER to realize EI. Yi *et al.* [16] deploy ERs in an EI based on electric vehicles. It also provides feasibility analysis for applying ERs from power marketing, district heating, and control perspective. Behi *et al.* [17] design a wireless green ER platform for controlling and scheduling of energies in energy-efficient buildings. In the ER design and power transfer area, Sanchez-Squella *et al.* [18] consider green efficient

Manuscript received July 30, 2017; revised November 12, 2017 and December 4, 2017; accepted December 10, 2017. Date of publication January 9, 2018; date of current version April 3, 2018. This work was supported in part by the National Natural Science Foundation of China under Grant 61572262, in part by the China Postdoctoral Science Foundation under Grant 2017M610252, and in part by the China Postdoctoral Science Special Foundation under Grant 2017T100297. Parts of this paper have been published on IEEE Global Communications Conference, 2017. Paper no. TII-17-1699. *(Corresponding author: Kun Wang.)*

M. Gao and L. He are with the State Key Laboratory of VLSI and Systems, Fudan University, Shanghai 201203, China, and also with the Electrical and Computer Engineering Department, University of California, Los Angeles, CA 90095 USA (e-mail: gaom@seas.ucla.edu; lhe@ee.ucla.edu).

K. Wang is with the National Engineering Research Center of Communications and Networking, Nanjing University of Posts and Telecommunications, Nanjing 210003, China, and also with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: kwang@njupt.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2018.2791537

energy transfer among subsystems connected by the ERs. Miao *et al.* [19] formulate the steady-state power flow model of the ER-embedded system network and related optimal power flow formulation to optimize power system operation. Hambridge *et al.* [20] introduce an economic-based energy routing strategy utilizing energy storage to reduce consumption of grid power. However, all the work mentioned above lack the consideration of soundness and completeness of the system design objective.

Formal verification proves the correctness of the system with respect to a certain formal specification or property. For example, in the software field, many testing tools exploit formal verification in bug finding [21]. One of the major approaches in formal verification is model checking [22], which formulates the system as transition graphs. It exhaustively traverses the graph and verifies whether the model satisfies the formula representing the property. In the real world, systems are inherently probabilistic; thus, quantitative properties are of the greatest interest to verify in addition to logic properties. Probabilistic model checking [23] formulates systems into probabilistic transition models such as discrete-time Markov chains (DTMC), continuous-time Markov chains (CTMC), and Markov decision processes (MDP). A quantitative logic property is then applied to the model to check the result and return a counter-example if a property is not satisfied.

In this paper, we propose a probabilistic model checking method to ER-based system design and monitor ER-based system's running behavior via our scheduling scheme. Specifically, we first propose a CTMC model on an ER-based system containing multiple ERs to check the reliability of the system operation. Then, we propose a CTMC model on a green ER-based subsystem to perform model checking on its reliability and communication count properties. To apply all the communication functions into the real scenario for green cities, we propose an MDP electricity trading model, and model check quantitative properties on the service requester's cost and the service provider's loss. We also propose an energy scheduling simulation scheme for the ER-based system. In this scheme, we divide a load demand curve into multiple time windows, and then project each demand in a time window into a cloudlet in cloud computing, and then the energy scheduling process in the ER-based system is projected to host allocation process in cloud computing. We define our own host allocation policy to complete the scheme.

The contributions of this paper are summarized as follows.

1) We introduce CTMC and MDP state machines to model ER-based systems. To the best of our knowledge, it is the first time that a formal verification technique is applied to ER-based systems.
2) We project the energy scheduling of the ER-based system into cloud computing area, and implement a tool to observe the performance of ER-based systems due to the similarity of these two areas. It is the first time that a cloud computing tool is tailored to suit the need for ER-based systems.
3) We consider both electricity price and line loss during power transmission during the selection of power service providers. Extensive experiment verifies the effectiveness of the proposed models and the monitoring scheme.

The rest of the paper is organized as follows. Section II introduces preliminary knowledge. Sections III and IV introduce probabilistic model checking on architecture modeling and trading behavior modeling of the ER-based system, respectively. Section V mentions scheduling for the ER-based system. Section VI shows experiments and results. Section VII concludes the paper.

II. Preliminary

A. Continuous-Time Markov Chain

CTMC is an automata in a continuous-time domain that preserves Markov property [23].

Definition 1. (Continuous-Time Markov Chain): A continuo us-time markov chain C is a 4-tuple (S, s_{init}, R, L) where S represents a finite set of states, $s_{init} \in S$ is the initial state, $R : S \times S \to \mathbb{R}_{\geqslant 0}$ is the state transition rate matrix, and $L : S \to 2^{AP}$ is a labeling function that assigns to each state $s \in S$ a label with atomic propositions.

The transition rate is used as the parameter of the exponential distribution. The probability triggered to make state transition before t time units for a rate r is $1 - e^{-rt}$.

As for model checking for the CTMC, continuous stochastic logic (CSL) [23] is widely used to specify properties.

Definition 2. (Augmented Continuous Stochastic Logic): The state formula Φ and the path formula ϕ of Augmented CSL is defined by

1) $\Phi ::= true|a|\neg\Phi|\Phi \wedge \Phi|P_{\sim p}[\phi]|S_{\sim p}[\Phi]$
$|R_{\sim r}[I^{=t}]|R_{\sim r}[C^{\leqslant t}]|R_{\sim r}[F\Phi]|R_{\sim r}[S]$
2) $\phi ::= X\Phi|\Phi \cup^{I} \Phi$

where a is an atomic proposition, P is the probability operator for all paths, S is the steady-state operator, R stands for the reward function, I in Φ is an immediate time stamp, C stands for the accumulated time, F stands for the future states, X stands for the next state, I in ϕ is an interval of $\mathbb{R}_{\geqslant 0}$, $\cup$ is an until operator, $\sim$ is a logical operator $\in \{<, \leqslant, >, \geqslant\}$, $p \in [0, 1]$, $r, t \in \mathbb{R}_{\geqslant 0}$.

Given a CTMC model C and a CSL formula ϕ, CTMC model checking outputs a set of states $\text{Sat}(\phi) = \{s \in S | s \models \phi\}$. The major task is to calculate probabilities or rewards for $P_{\sim p}[\cdot]$, $S_{\sim p}[\cdot]$, and $R_{\sim r}[\cdot]$, which is discussed in [23].

B. Markov Decision Process

An MDP is a nondeterministic state transition automata that preserves Markov property [23].

Definition 3. (Markov Decision Process): A Markov decision process M is a 4-tuple $(S, s_{init}, Steps, L)$ where S represents a finite set of states, $s_{init} \in S$ is the initial state, $Steps : S \to 2^{Act \times Dist(S)}$ stands for the state transition probability function where Act is a set of actions and $Dist(S)$ is the set of probability distributions over the set S. $L : S \to 2^{AP}$ stands for a labeling with atomic propositions.

Probabilistic computation tree logic (PCTL) and its augmented version serve as the backbone in model checking an MDP.

Definition 4. (Augmented Probabilistic Computation Tree Logic): The syntax of augmented PCTL is shown as follows:

1) $\Phi ::= true|a|\neg\Phi|\Phi \wedge \Phi|P_{\sim p}[\phi]$

$$|R_{\sim r}[I^{=k}]|R_{\sim r}[C^{\leqslant k}]|R_{\sim r}[F\Phi]$$

2) $\phi ::= X\Phi|\Phi \cup^{\leqslant k} \Phi|\Phi \cup \Phi$

where a is an atomic proposition, P is the probability operator for all paths, R stands for the reward function, I in Φ is an immediate time stamp, C stands for the accumulated time, F stands for the future reachable states, X stands for the next state, $\cup$ is an until operator, $\sim$ is a logical operator $\in \{<, \leqslant, >, \geqslant\}$, $p \in [0, 1]$, $r \in \mathbb{R}_{\geqslant 0}, k \in \mathbb{N} \cup \{\infty\}$ is an until bound.

The main unique task for augmented PCTL is to evaluate the probabilistic operator P and the reward operator R. Evaluating $P_{\sim p}[\phi]$ is reduced to evaluate either the minimum probability of ϕ holding or the maximum probability of ϕ holding depending on the $\sim$ operator over all the paths. If $\sim \in \{<, \leqslant\}$, then the maximum probability is calculated to check ϕ holding for $s \models P_{\sim p}[\phi]$, otherwise the minimum probability is calculated. The calculation of reward follows the similar idea.

C. Energy Router Based Subsystem Architecture

The architecture of an ER-based subsystem is shown in Fig. 2. As stated in [24], the router mainly contains a communication module and a control module. Ports provide plug-and-play interfaces controlled by the router to conventional power source, renewable power source such as wind turbine and solar cells, as well as energy storage and load. Router communicates to each port through wired cord or wireless channels. Power electronics devices such as solid-state transformers, converters, and inverters electrically connect all the ports. In our work, the main modeling focus is on the communication and control module instead of the power electronic devices. The functionality of the ER falls into user-level and gate-level. At user-level, main functions include user attachment/detachment, service request/termination, and status update. At gate-level, the ER performs communication between not only its local users but also other ERs. Real-time energy control, management, and trading are, thus, available through these basic functions.

III. Architecture Modeling of the Energy Router System

We propose two CTMC models to describe the green ER system architecture. The first model describes the behavior of a system consisting of multiple ERs shown in Fig. 1, and the other is a model on the operation of an ER-based subsystem shown in Fig. 2. In a Poisson process, the length of the interarrival time follows exponential distribution. Exponential distribution also describes naturally the time for a continuous process to change time. We choose CTMC because we assume state transitions follow exponential distribution, and the state transition is memoryless, and related work on dynamic energy management [25] models the behavior of a power-managed system as a CTMC model.

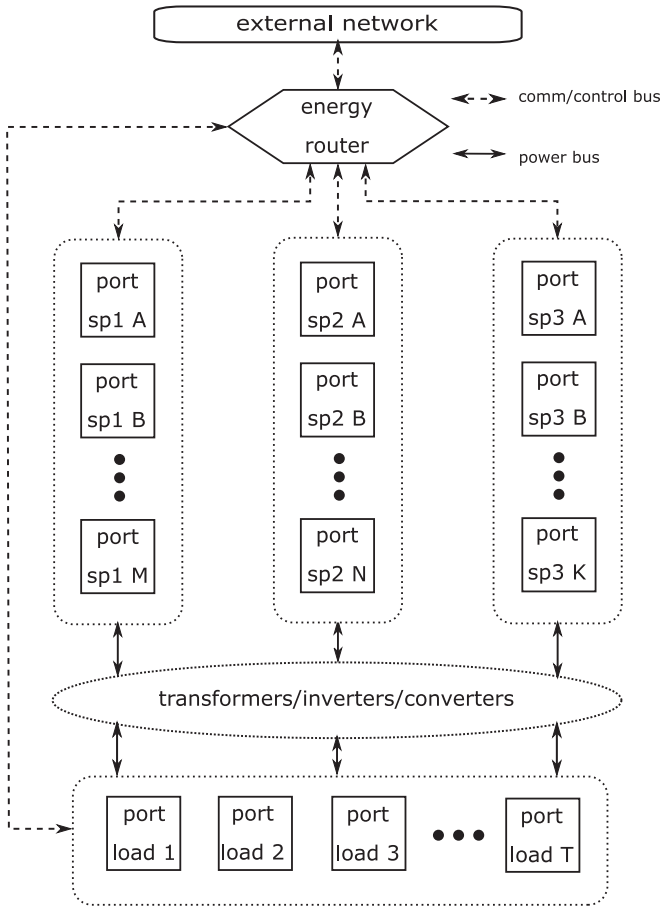


Fig. 2. Architecture of a single ER-based subsystem.

A. Modeling of the Multiple Energy Router System

A multiple-ER system contains multiple ERs and multiple associated ports. We generalize Fig. 1 to a controller module, an ER set module representing a set of ERs, and a port module representing the set of ports controlled by the routers. Modeling every operation in each component results in the scale of the model growing exponentially, thus causing the state-space explosion. Instead of directly modeling each component's broken and recovered state using 2^n states, we set the number of broken components c_{down} as the state variable because we do not care exactly which component is broken from the reliability perspective. c_{down} increases and decreases according to the failure and recovery rate. We assume the failure rate is dependent on the number of failed components. This is because once a failure happens, all the similar components will be checked to prevent the same issue from happening again easily. Thus, we set an coefficient to regulate the failure rate. The regulated rate r_{fail} is given by

$$r_{\text{fail}} = e^{\frac{-\text{comp}_{\text{down}}}{\text{comp}_{\text{total}}}} \times \text{MTTF}, \tag{1}$$

TABLE I
LOCAL STATE VARIABLE FOR A SINGLE ER-BASED SUBSYSTEM

Variable name	Value	Explanation
s_p (port state)	0	broken
	1	idle
	2	sending message and waiting response
	3	receiving message and processing
s_r (router state)	0	broken
	1	idle
	2	sleeping
	3	processing incoming message
	$\geqslant 4$	send message back to ports
q_{size}	0 to maxsize	number of requests in the queue
in_x	0–1	whether X is in the cell

where $\text{comp}_{\text{total}}$ is the total number of components, and MTTF is the mean time to failure of the component. This abstraction will reduce the state space to the order of n^2.

The controller module serves as the hub of the system, it contains three operation states—*SLEEP*, *BUSY*, and *BROKEN*. In the normal operation mode, the controller switches between *SLEEP* and *BUSY* if the number of down routers and ports is less than the maximum threshold. The controller itself transits to a *BROKEN* and reverts to *SLEEP* after repairing according to its rate.

The state transition of the three modules are synchronized through actions, i.e., one action triggers state transitions in multiple modules simultaneously. The failure of the system is defined by any of the component that is unable to meet the normal operation requirement. We label the system failure by "down" as

$$\text{down} = s_{\text{ctr}} == 0 \vee f_p > 0 \vee f_r > 0, \tag{2}$$

where s_{ctr} is the controller state, and f_p and f_r are the number of failed ports and routers, respectively. The reliability property $S = ?[\text{"down"}]$ is, thus, the main property we are interested in.

B. Modeling of a Single Energy Router Based Subsystem

We consider an ER-based subsystem shown in Fig. 2 whose control and communication functionality is achieved through wireless communication. Thus, in our model, a local cell module with a guarded channel[26] to reduce the handoff dropping probability is introduced as the communication medium. A handoff drops when a device enters a new cell but the new cell has no channel to provide. Guarded channels are reserved exclusively for the handoff calls. In this module, the number of calls in the cell nLocal and existence of the ports and routers in_x are set as state variables. We set the router as the handoff call and all the other ports as regular calls. Enter or departure of these calls will increase or decrease nLocal and change in_x correspondingly.

In port module, we define four operating states listed in Table I. The normal duty cycle circulates among port *IDLE* ($s_p = 1$), *SEND* ($s_p = 2$), and *RECEIVE* ($s_p = 3$) according to the transition rate. One of the guards for each state transition is $\text{in}_x = true$. The port may run into *BROKEN* from any states. A repair action resets *BROKEN* back to *IDLE*.

TABLE II
SINGLE ER-BASED SUBSYSTEM PROPERTIES

Property	Explanation
$S = ?[\text{"down"}]$	The system is down in the long run
$R\{\text{"comm"}\} = ?[C \leqslant T]$	The total number of services that the router has performed until time T

The router module receives all the messages and requests from the ports, processes all those messages and requests, and then sends results back to all the ports. To reduce computing complexity of the model checker, we choose to lump the service model as a sequential queue. Thus, we embed an M/M/1 service queue module to model the service queue inside of the router. Different ports send messages to the queue according to its own rate while increasing q_{size} by 1. The router first transits from *SLEEP* ($s_r = 2$) to *IDLE* ($s_r = 1$). After checking the local cell module for a true value of $\text{in}_{\text{router}}$, it fetches one request from the queue, and then transits to one of the sending back states. This is a nondeterministic transition whose transition probability is proportional to the request rate. We then have the following lemma.

Lemma 1: The state transition probability of a set nondeterministic of n choices in the router is given by $\text{req}_{\text{i}}/\Sigma\text{req}_{\text{i}}$.

Proof: The controller sends the message back to the head of the queue, which is the index of the ports sending out the request with the minimum time. Since all the ports follow an exponential distribution of parameter req_i, the minimum time distribution parameter is given by Σreq_i. For a single port, the probability is $\text{req}_i/\Sigma\text{req}_i$.

We then set the transition rate proportional to this probability to reflect the right transition distribution.

q_{size} is decreased by 1 after the choice is made, and s_r goes back to idle. *IDLE* transits to *SLEEP* according to a preset rate. The router may run into the *BROKEN* state from any of the states mentioned above. A repair action then resets the state into the sleeping state.

The main properties to be checked are listed in Table II. One crucial property of for this system is the reliability. The failure of the system is defined by any of the component that is under broken state or disconnected from local cells. We, thus, label the system failure as "down" by

$$\text{down} = s_r == 0 \vee s_{ps} == 0 \vee \text{in}_x == 0, \tag{3}$$

and intend to observe the probability of failure in the long run. Another property we are interested in is energy consumption during the system operation. We generalize it to the measurement of communication operation count by setting a reward of 1 to all the communication actions and labeling those actions with comm.

IV. MODELING OF ENERGY ROUTER SUBSYSTEM BASED ELECTRICITY TRADING

With the real-time communication support of ER, green energy management and trading can be applied. Based on the ER subsystem, we propose and model check an MDP electricity trading scheme, which extends the general trust model in [27]

TABLE III
LOCAL STATE VARIABLE FOR A SERVICE REQUESTER

Variable Name	Value	Explanation
s_{req} (requester state)	0	idle
	1	a load request is generated
	x1	requester is requesting sp_x
	x2	sp_x accepts the service and issues a price
	x3	sp_x refuses the service
$\text{sp}_{x,r}$	0–1	whether sp_x refuses service at this time stamp

TABLE IV
LOCAL STATE VARIABLE FOR A SERVICE PROVIDER

Variable name	Value	Explanation
s_{sp} (service provider state)	0	idle
	1	processing request
know_X	0	sp_x has no transaction record for req
	1	sp_x has transaction record for req
th_x	1–10	threshold for sp_x to refuse service
tr_x	1–10	reputation of req to spX

by considering power demand and demand response as a real-world application of the ERs. We choose MDP because state transitions are made based on decisions.

The whole scheme models the trading operations for n hours. Every 1.5 h, the requester req can choose whether to generate a load demand or not. Once a provider accepts the request, req has the nondeterministic choice on whether to accept and whether to actually pay for the service with a reference of its own price threshold. Prices should be paid after the service begins but before next request decision. Within n hours, req has to successfully receive and consume power k times.

The proposed model contains one service requester req and multiple service providers offering electricity from green energy sources. We assume that collaboration between requesters will not gain more reward, since in between a group of load ports, there are no additional devices to perform power redistribution. We also assume any supplier has enough power to provide once it agrees to provide service, and line loss for all the suppliers is the same to keep the scale of the proposed model reasonable.

A. Requster Modeling

Table III lists all the local state variables for a requester req. When a load request is generated ($s_{\text{req}} = 1$), req chooses one service provider randomly, setting $s_{\text{req}} = x1$. If a service provider sp_x agrees to provide the service after checking req's trust level T_x, a synchronized action takes place on both the provider x and the req along with a price P_x calculated, and thus, $s_{\text{req}} = x2$. Otherwise, another synchronized action sets $s_{\text{req}} = x3$. The reputation of the requester req at sp_x is denoted by tr_x, representing the extend of provider x trusting req. The trust level T_x is a linear combination of tr_x at sp_x and other service providers, which is given by

$$T_x = \alpha \cdot \text{tr}_x + (1 - \alpha) \cdot \text{rec}_x \tag{4}$$

where

$$\text{rec}_x = \begin{cases} \text{tr}_x, & \text{if know}_y = 0 \ \forall y \in Y \\ \frac{\Sigma \text{ite}(\text{know}_y, \text{trust}_y, 0)}{\Sigma \text{ite}(\text{know}_y, 1, 0)}, & \text{otherwise} \end{cases} \tag{5}$$

is the recommendation factor from other service providers, and know_y is true if there exists the transaction history on sp_y. ite is an "if-then-else" expression.

A uniqueness in the power system is that power demand fluctuates over time. Thus, we set P_x at different timestamps in accordance with the fluctuation. One more factor needed to be considered is the req's reputation, and we set an inverse scaling factor on the time-specific price to reflect the reputation. Thus, the price for a requester req once a provider x agreed to serve is given by

$$P_x = C \cdot \left(\frac{\beta}{\text{tr}_x} + 1 - \frac{\beta}{D} \right) \cdot \text{Pr}_{\text{demand}}(t) \tag{6}$$

where $C, \beta, D \in \mathbb{R}_{\geqslant 0}$, and $\text{Pr}_{\text{demand}}(t)$ is a piecewise constant demand approximation function whose data are given in [28]. C, β, D constitute the conversion rate from demand and price. β, D take trust into account in an inverse proportional fashion, and C is a conversion factor with a unit of \$/kwh.

Once P_x is issued, req compares P_x with its own target price P_{target}. P_{target} is in proportion to the power demand over time regulated by a parameter γ, which is global to both requester and providers. req has two nondeterministic choices when a low P_X is issued. One is to get the service and pay, or get the service without payment. If req finds P_x is higher than expected, it can withdraw the load request in addition to the previous two choices, which reflects the demand response.

If the service provider refuses the service request, then the corresponding $\text{sp}_{x,r}$ flag is set to 1. req continues to query other unvisited service providers until all the providers refuse to serve. $\text{sp}_{x,r}$ are reset to 0 before the next time stamp begins.

B. Service Provider Modeling

Table IV lists all the local state variables for a service provider sp_x. When a synchronized service request action arrives, sp_x compares the trust level T_x with th_x. An accept action sets s_{sp} to 1, know_x to 1, and changes s_{req} in req when $T_x \geqslant \text{th}_x$, or a reject action sets s_{sp} to 0 otherwise. It is reset to 0 after the request get processed before the next time stamp.

Getting paid service gains reputation bonus on req, and reputation penalty if the service is not paid. Accordingly, the trust threshold of req varies depending on the choice made. Unpaid service results in a higher threshold, and paid service results in a reasonable threshold. We set small penalties and threshold increase on lower P_x, and high penalties and threshold increase on higher P_x. All the actions are synchronized with req; thus, when corresponding conditions for an action on req and sp_x hold, both states changes simultaneously.

In order to guarantee the load having k times power to consume, we set a have-to-pay mode for req and a have-to-serve mode for the provider when the remaining time is limited. req stops request power when it is been served for k times.

TABLE V
GREEN ELECTRICITY TRADING PROPERTIES

Property	Explanation
$R\{\text{“cost”}\}\min = ?[F(\text{time} \geq \text{max\_time})]$	req's minimum cost to get required number of service
$R\{\text{“loss”}\}\max = ?[F(\text{time} \geq \text{max\_time})]$	sum of sp$X$'s maximum loss to provide required number of service

Table V lists the quantitative properties we check for the MDP model. From the requester perspective, req wants to get the service with the minimum amount of money. In our model, we label all the paid actions as “cost” and accumulate the cost until the end. From the service provider perspective, providers intend to estimate their maximum loss before participating the service. In our model, we label all the unpaid actions as “loss” and accumulate the loss until the end.

V. ENERGY ROUTER SYSTEM SCHEDULING

Probabilistic model checking for the ER-based system shows a feasible way of calculating safety probabilities and optimal costs. To observe single behavior or average performance of the ER-based system for better decision-making purposes, a scheduling platform is needed. This goal is similar to cloud computing since both cloud computing and ER-based system perform task scheduling and allocation. In this section, we implement a scheduling simulation tool on top of the cloud computing tool *CloudSim* [29] to utilize the similarities.

A typical *CloudSim* system compromises of a data center, a data broker, multiple cloudlets, virtual machines, and hosts. The *data center* is an abstract container that manages all the physical computing sources. Those computing sources are called *hosts*. The data center *broker* is responsible for sending multiple computing tasks to the data center and acknowledge its completion. Each computing task is called a *cloudlet*, and usually those cloudlets can be bound with computing resources designated by the cloud service provider. We call those computer resources a *virtual machine*. The broker sends all required virtual machines into the data center as well. The data center allocates virtual machines to different hosts according to the user-defined allocation policies.

Since both cloud computing and ER-based system perform task scheduling and allocation for some specific entities, an interesting question is that can the ER-based system be projected into cloud computing based platform? In this section, we provide a positive answer. We first abstract the ER along with all the power sources as a data center. Then, we divide a load demand demand_i into multiple piece-wise constant time windows. Each time window of demand_i is regarded as a set of cloudlets that starts computing at different specific time stamps. A virtual machine is directly bound to a cloudlet to guarantee that the cloudlet can finish computing within the scheduling interval. We label the virtual machine as vm_i^j representing for demand_i at a time window order j. Thus, $\text{demand}_i = \{\text{vm}_i^1, \text{vm}_i^2, ..., \text{vm}_i^N\}$. A host $host$ then becomes a power source provider naturally. Then the question for the ER-based system becomes for each demand_i at the beginning of each time window j, vm_i^j should be allocated to which host_k via the ER according to designated policy.

A. Implementation of ER-Based System Simulation

From the problem formulation, we need *CloudSim* to have the following three properties.

1) All virtual machines should be able to start at a user-specified time stamp.
2) A virtual machine should be freed from its host and destroyed once the task is completed.
3) A new host allocation policy is required to make load choose a feasible power service provider.

1) Modifying Virtual Machine Starting Time: We modified the interface of *CloudSim* to suit all the needs for ER-based system simulation. In *CloudSim*, all the cloudlets stored in an array are submitted by the broker at the beginning of the experiment via the function call $sendNow$. We created a new class $ERCloudlet$ specifying the start time the cloudlet to be scheduled, and replaced $sendNow$ with another API $send$ in the original source code that can specify the starting time of a cloudlet to be executed. Instead of submitting cloudlets by array at time 0, the broker in our implementation submits cloudlets one by one at their designated starting time.

2) Modifying Virtual Machine Destroy Time: The destroy process of all virtual machines in *CloudSim* happens when all the cloudlets completed their computing task. In the ER-based system, a cloudlet stands for the energy needed in a time window. Thus, it should be finished before the time window ends. Consequently, the virtual machine should be deallocated from the host before next time window comes. In our implementation, we destroy virtual machines one by one and shifted the destroy process to an earlier stage when the broker acknowledges that the virtual machine has finished computing.

3) Host Allocation Policy for the ER-Based System: Host allocation policy in the cloud computing defines which virtual machines are allocated to which host. Under the scenario of the ER-based system, this policy assigns vm_i^j representing for load demand to host_k that stands for a power service provider. To be more specific, a load requests power to all power providers through the ER during the start of each time window, and then each power provider sends back a price. We set the pricing scheme to be time-dependent, and also trust dependent if the trust system is enabled. The load chooses which power station to provide power supply according to the user-defined criteria and also chooses to pay or not to pay after receiving the desired power if the trust system is enabled. The load's trust increases if it pays for the demand generated, otherwise the trust decreases.

4) Trust-Based Power Transaction: The scheme for building a trust system is totally customized. Our implementation of the trust system is almost identical to Section IV. The trust evaluation for trust query scheme is identical to (4) and (5). The pricing scheme for a service provider is identical to (6). The only difference is that we disabled $\text{sp}x, r$ but setting a load's trust to the minimum instead. This guarantees every vm_i^j will receive power service.

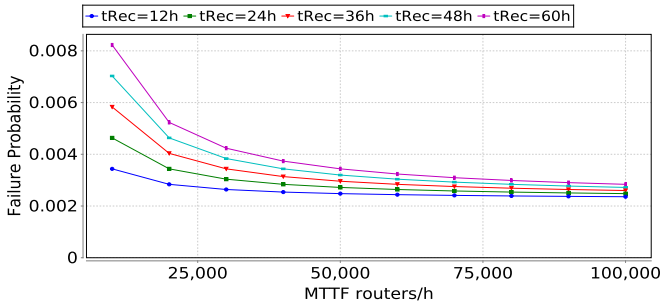


Fig. 3. Multiple ER system failure probability.

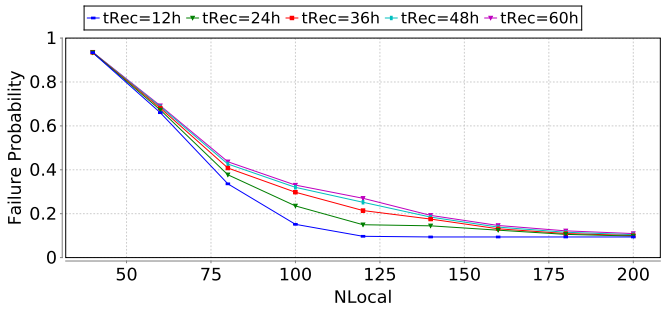


Fig. 4. Single ER-based subsystem failure probability.

A load will choose a power provider not only considering the electricity price, but also line loss. Line-loss cost loss_L defines the money wasted on the energy loss transmitted from a power plant to the load. Power service provider with longer distance incurs higher loss_L. For simplifying computation complexity, we define the total payment P_{total} to get the required amount of energy to be approximately $P_x + \text{loss}_L$. The load checks the cost from all the hosts, and then chooses the service provider that has enough power with the lowest cost. Each node has its estimated price P_{est}, and we relate P_{total} and P_{est} via a coefficient f_{exp}. If $P_{\text{total}} > f_{\text{exp}} * P_{\text{est}}$, the load will have a higher probability not to pay the service, otherwise the load will have a lower probability of not paying.

VI. Experiments and Results

We implemented all three models in the probabilistic model checking tool *PRISM* [30]. Prism is a probabilistic model checker supporting modeling of DTMCs, CTMCs, MDPs, and PTAs. We also implemented our ER-based simulation framework on top of *CloudSim*. All the evaluations and simulations are run on a desktop equipped with an Intel E5-2643@3.30G CPU and 128 GB memory. The PRISM's CUDD memory is set to be 4 GB, and java memory is set to be 16 GB.

A. Architecture Model Properties

The multiple-ER model contains 1 hub, 10 ERs, and 1000 ports. It has 0.8 k states and 2 k state transitions. Configurable parameters include MTTF and recovery time for the controller, the ERs, and the ports. Our main interest for this model is its reliability. We varied the MTTF and the recovery time for the ERs to verify the reliability property whose result is shown in Fig. 3.

We observe that the system failure probability goes down when the ERs have larger MTTF and faster recovery time, and an ER MTTF larger than 100 000 h provides limited contribution to the system reliability because of the probability convergence. Curves for different MTTF settings for each ER will be bounded in this set of curves.

The ER-based subsystem model contains one ER module, one local cell module, one service queue module, and three port modules. The number of channels in the local cells NLocal and ER's recovery time are set as experiment parameters, where NLocal varies from 40 to 200, and $t\text{Rec}_{\text{Router}}$ varies from

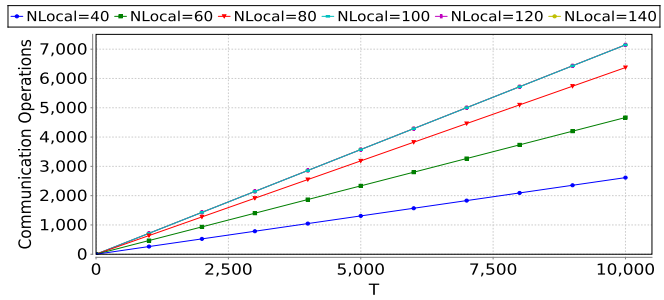


Fig. 5. Single ER-based subsystem communication count.

12 to 60 h. Average life time for the ER is 50 000 h. Average life time for all three ports is 1, 1.5, and 2 months. Request rate for all three ports, which are the inverse of the corresponding average request frequency, is 0.015, 0.02, and 0.025. In the service queue module, the maximum capacity for the service queue is 50. In the local cell module, the arriving rate of new calls and handoff calls is 49 and 21, and the departure base rate is 1. In total, the whole model contains approximately 14–54 millon states and 50–489 millon state transitions depending on the number of local channels.

The model checking result for the reliability property is shown in Fig. 4 where NLocal stands for the number of channels in the local cells. We observe that the reliability of the local cell contributes the major part of the failure. Higher cell channel capacity results in lower failure probability. Faster recovery time results in lower failure probability as well.

We also checked the total router related communication operations property shown in Fig. 5, where T varies from 0 to 10 000 s to get a taste of energy consumption under the current parameter setting. We observe a linear increment of number of operations over time, and an increment with more local channels at the same time. The operation number saturates when nLocal is large enough.

B. Electricity Trading Model Properties

In this model, we checked properties from both the service requester and the service providers perspective. This model contains one service requester and three service providers. The requester is required to consume energy for 15 times during 48 time slots (72 h). The trust level varies from 1 to 10. We set α to 0.5, $\beta \in \{1, 3, 5\}$, and γ varying from 0.5 to 1.5 with a step of

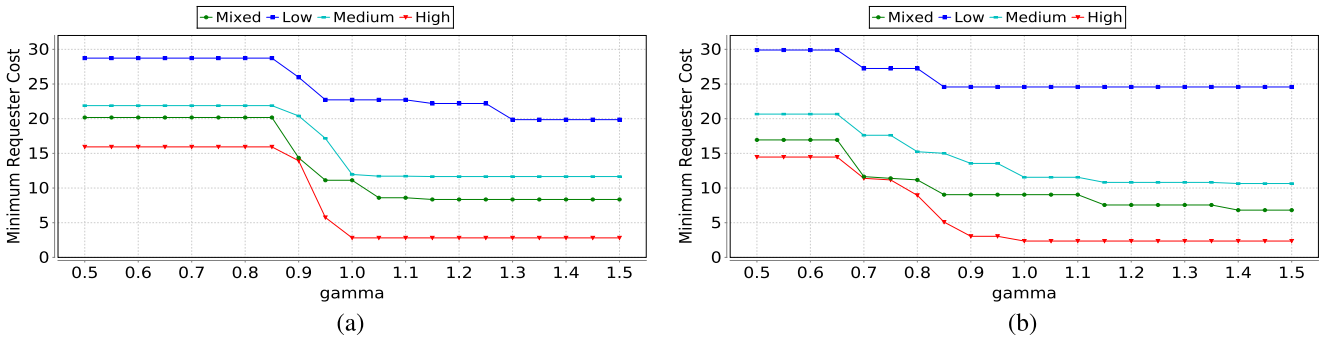


Fig. 6. Minimum cost for the requester to get the required amount of services. (a) $\beta = 1$. (b) $\beta = 3$.

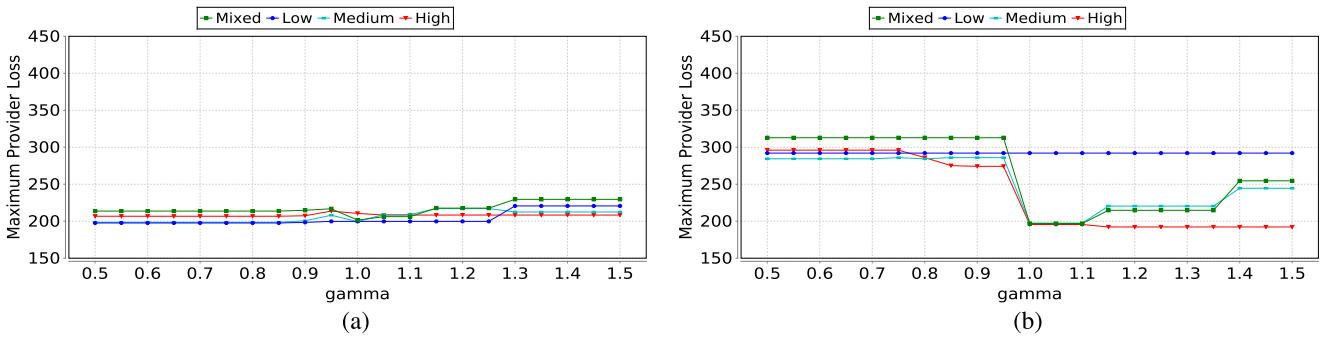


Fig. 7. Maximum loss for the provider to provide the required amount of services. (a) $\beta = 1$. (b) $\beta = 3$.

0.1. The full model contains 113 million states and 194 million transitions. We check the minimum cost from the service requester side, and the maximum loss property from the service provider side.

The minimum loss for the requester to get required services is shown in Fig. 6, where the mixed (2/5/8), low (2/2/2), medium (5/5/5), high (8/8/8) stands for the requester's initial trust levels for all three providers. The run time for the experiment is 14 h. We observe a higher γ results in smaller cost, and the cost starts to drop earlier when β increases. It is also clear that the requester's strategy to get lower cost is to increase its target price, and the cost starts to saturate when β is large.

The maximum loss for all the service providers is shown in Fig. 7, where the initial trust settings are identical to the minimum loss checking. We observe different behaviors on varying β and γ. When β is small, all four trust settings show similar increasing trends, which are insensitive to γ. The interpretation behind is that a small γ does not regulate the price tight enough; thus, under all situations, the requester will take a similar strategy to pay as less money as possible to get the service. The amount increases in high γ due to the decrement in the requester's cost. When β is large, low initial trust indicates the requester has only limited chances to get the service. Thus, it follows the same strategy regardless of γ. Under other initial trust levels, a small γ results that the issued price is always higher than the target price. To guarantee the load consumption with the minimum money, the user will perform unpaid actions as many as possible. Thus, the service providers suffer constant high loss. When γ is high, the issued price can be lower than the target price. This increases the choices for the requester to pay less, and the maximum loss gradually increases. However, all high initial trusts make the optimal choices fixed again because every transaction will be consented by the providers. Thus, the solution space becomes the same regardless of γ.

C. Scheduling Behavior of the ER-Based System

We built an ER-based system containing three loads and three power service providers controlled by an ER. The system contains three categories of input data, distance from each load to each power provider, a 5-day demand with a unit of MWh for each load with a time window of 1 h length, a 5-day time-dependent electricity price having a unit of \$/MWh from a power provider with a time window of 1 h length. The distance is generated by a random number generator, and the other two data come from running data of ISO-NE [31]. We set a 0.15 probability that a load will refuse to pay the service if the price is higher than expected, and 0.05 if lower than its expectations. Then we varied β from 1 to 3 in (6) and $ExpectPriceFactor$ from 1 to 2 to observe the service providers total revenue and total line loss in dollars within 5 days. Since the load actions are stochastic, for each parameter setting, we performed the experiment 500 times and returned their average outputs. We also measured the total revenue and total line loss using time-based pricing only with no trust system applied as a base scenario to compare.

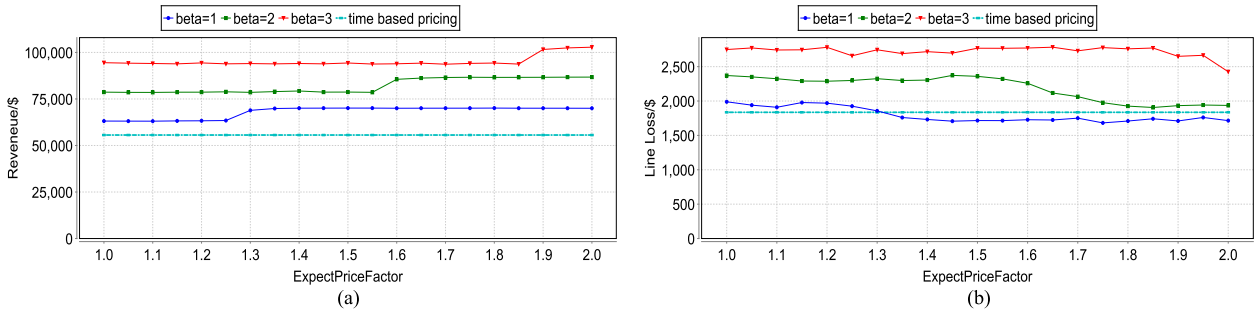


Fig. 8. Average total revenue and average total line loss for the ER-based system. (a) Average total revenue. (b) Average total line loss.

Fig. 8(a) shows the total revenue of all service providers. Comparing with the time-based pricing, we observe that introducing the trust system in general increases the total revenue since a higher price will be applied to the load with low trust score. A higher $beta$ value yields higher revenue since high β increases the price for low trust score load. We also observe that load's $ExpectPriceFactor$ has the effect on the total revenue. The total revenue increases when $ExpectPriceFactor$ is around $1.3, 1.6, 1.9$ when β equals to $1, 2, 3$, respectively. This indicates that below those factors, the prices issued by the service providers are in general satisfy loads' expectations, and thus, less loads refuse to pay the price. When the factor is high, more loads refuse to pay for the power they got from the power provider, and thus, bad trusts result in higher revenue for the providers.

Total line loss of the experiment is shown in Fig. 8(b). We observe that the line loss increases while β is increasing. This is because the higher price with higher β yields higher loss in terms of money even with the same loss in terms of energy. Higher $ExpectPriceFactor$ helps reduce the line loss in general. Comparing to the no trust time-based pricing case, $\beta = 1$ yields even lower line loss when $ExpectPriceFactor$ is high.

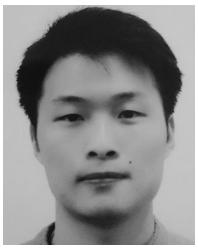
VII. Conclusion and Future Work

In this paper, we applied probabilistic model checking to verify the ER system design in EI for green cities. We proposed two CTMC models to describe an operation of a multiple ER system and a single ER subsystem and presented their reliabilities and expected communication operations through probabilistic model checking. To apply ER system functions into a real scenario for green cities, we selected electricity trading as an example and proposed an MDP model to describe the trading behavior, and quantitatively model checked the minimum cost of the service requester and the maximum loss of the service providers. We also introduced an ER-based system scheduling monitoring tool to observe the system's scheduling behavior. We projected the power demand and supply transaction process to computing cloudlets in cloud computing, and then modified *CloudSim* to suit the need of ER-based system transaction. As the future work, we will perform probabilistic model checking on more real functions in the ER system, apply other formal verification techniques to verify the system design, and improve the monitoring tool to model more complex ER-based system design for the future development of green cities.

References

[1] S. Cherry, "How to build a green city," *IEEE Spectr.*, vol. 44, no. 6, pp. 26–29, Jan. 2007.
[2] A. Becerra, W. Zeng, M. Chow, and J. Rodriguez-Andina, "Green city: A low-cost testbed for distributed control algorithms in smart grid," in *Proc. 41st 2015 Annu. Conf. IEEE Ind. Electron. Soc.*, vol. 68, 2015, pp. 1948–1953.
[3] W. Ejaz, M. Naeem, A. Shahid, A. Anpalagan, and M. Jo, "Efficient energy management for the internet of things in smart cities," *IEEE Commun. Mag.*, vol. 55, no. 1, pp. 84–91, Jan. 2017.
[4] K. Wang *et al.*, "A survey on energy internet: Architecture, approach, and emerging technologies," *IEEE Syst. J.*, vol. PP, no. 99, pp. 1–14, Jan. 2017.
[5] W. Zhong, R. Yu, S. Xie, Y. Zhang, and D. Tsang, "Software defined networking for flexible and green energy internet," *IEEE Commun. Mag.*, vol. 54, no. 12, pp. 68–75, Dec. 2016.
[6] K. Wang, X. Hu, H. Li, P. Li, D. Zeng, and S. Guo, "A survey on energy internet communications for sustainability," *IEEE Trans. Sustain. Comput.*, vol. 2, no. 3, pp. 231–254, Jul./Sep. 2017.
[7] H. Jiang, K. Wang, Y. Wang, M. Gao, and Y. Zhang, "Energy big data: A survey," *IEEE Access*, vol. 4, pp. 3844–3861, 2016.
[8] O. Ojala and T. Ferm, "Building green city with green choices in traffic," in *Proc. eChallenges e-2015 Conf.*, vol. 68, 2015, pp. 1–8.
[9] M. Moreno *et al.*, "Applicability of big data techniques to smart cities deployments," *IEEE Trans. Ind. Informat.*, vol. 13, no. 2, pp. 800–809, Apr. 2000.
[10] J. He, J. Wei, K. Chen, Z. Tang, Y. Zhou, and Y. Zhang, "Multi-tier fog computing with large-scale IoT data analytics for smart cities," *IEEE Internet Things J.*, vol. PP, no. 99, pp. 1–10, Jul. 2017.
[11] K. Wang, H. Li, Y. Feng, and G. Tian, "Big data analytics for system stability evaluation strategy in the energy internet," *IEEE Trans. Ind. Informat.*, vol. 13, no. 4, pp. 1969–1978, Aug. 2017.
[12] W. Zhong, R. Yu, S. Xie, Y. Zhang, and D. K. Y. Yau, "On stability and robustness of demand response in V2G mobile energy networks," *IEEE Trans. Smart Grid*, vol. PP, no. 99, pp. 1–10, Nov. 2016.
[13] Y. Zhang, R. Yu, S. Xie, W. Yao, Y. Xiao, and M. Guizani, "Home M2M networks: Architectures, standards, and QoS improvement," *IEEE Commun. Mag.*, vol. 49, no. 4, pp. 44–52, Apr. 2011.
[14] K. Wang, Y. Wang, X. Hu, Y. Sun, D. J. Deng, A. Vinel, and Y. Zhang, "Wireless big data computing in smart grid," *IEEE Wireless Commun.*, vol. 24, no. 2, pp. 58–64, Apr. 2017.
[15] L. Xin, Z. Dong, Y. Sun, J. Hou, and J. Liu, "Design and application of energy router to realise energy internet," in *Proc. 10th Int. Conf. Adv. Power Syst. Control, Oper. Manage.*, 2015, pp. 1–6.
[16] P. Yi, T. Zhu, B. Jiang, R. Jin, and B. Wang, "Deploying energy routers in an energy internet based on electric vehicles," *IEEE Trans. Veh. Technol.*, vol. 65, no. 6, pp. 4714–4725, May 2016.
[17] M. Behl, M. Aneja, H. Jain, and R. Mangharam, "Enroute: An energy router for energy-efficient buildings," in *Proc. 10th ACM/IEEE Int. Conf. Inf. Process. Sens. Netw.*, 2011, pp. 125–126.
[18] A. Sanchez-Squella, R. Ortega, R. Grino, and S. Malo, "Dynamic energy router," *IEEE Control Syst.*, vol. 30, no. 6, pp. 72–80, Nov. 2010.
[19] J. Miao, N. Zhang, C. Kang, J. Wang, Y. Wang, and Q. Xia, "Steady-state power flow model of energy router embedded ac network and its application in optimizing power system operation," *IEEE Trans. Smart Grid*, vol. PP, no. 99, pp. 1–10, Feb. 2017.
[20] S. Hambridge, A. Huang, and R. Yu, "Solid state transformer (SST) as an energy router: Economic dispatch based energy routing strategy," in *Proc. 2015 IEEE Energy Convers. Congr. Expo.*, 2015, pp. 2355–2360.

[21] M. Gao, L. He, R. Majumdar, and Z. Wang, "Llsplat: Improving concolic testing by bounded model checking," in *Proc. 2016 IEEE 16th Int. Working Conf. Source Code Anal. Manipulation*, 2016, pp. 127–136.
[22] J. Burch, E. Clarke, K. McMillan, and D. Dill, "Sequential circuit verification using symbolic model checking," in *Proc. 27th ACM/IEEE Des. Autom. Conf.*, 1990, pp. 46–51.
[23] M. Kwiatkowska, G. Norman, and D. Parker, "Stochastic model checking," in *Proc. 7th Int. Conf. Formal Methods Perform. Eval.*, 2007, pp. 220–270.
[24] Y. Xu, J. Zhang, W. Wang, A. Juneja, and S. Bhattacharya, "Energy router: Architectures and functionalities toward energy internet," in *Proc. 2011 IEEE Int. Conf. Smart Grid Commun.*, 2011, pp. 31–36.
[25] Q. Qiu and M. Pedram, "Dynamic power management based on continuous-time Markov decision processes," in *Proc. 1999 Des. Autom. Conf.*, 1999, pp. 555–561.
[26] G. Haring, R. Marie, R. Puigjaner, and K. Trivedi, "Loss formulas and their application to optimization for cellular networks," *IEEE Trans. Veh. Technol.*, vol. 50, no. 3, pp. 664 –673, May 2001.
[27] A. Aldini and A. Bogliolo, "Model checking of trust-based user-centric cooperative networks," in *Proc. 4th Int. Conf. Adv. Future Internet*, 2012, pp. 32–41.
[28] H. Hildmann and F. Saffre, "Influence of variable supply and load flexibility on demand-side management," in *Proc. 8th Int. Conf. Eur. Energy Market*, 2011, pp. 63–68.
[29] R. Calheiros, R. Rajiv, A. Beloglazov, C. Rose, and R. Buyya, "Cloudsim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Softw. Pract. Exp.*, vol. 41, no. 1, pp. 23–50, Jan. 2011.
[30] M. Kwiatkowska, G. Norman, and D. Parker, "PRISM 4.0: Verification of probabilistic real-time systems," in *Proc. 23rd Int. Conf. Comput. Aided Verif.*, 2011, pp. 585–591.
[31] "ISO New England data." [Online]. Available: https://www.iso-ne.com/.



Min Gao received the B.Eng. degree in electrical engineering and automation from Nanjing University of Technology, Nanjing, China, in 2011, and the M.Sc. degree in electrical engineering in 2012 from the University of California, Los Angeles, CA, USA, where he is currently working toward the Ph.D. degree at the Electrical and Computer Engineering Department.

He was a Visiting Scholar with Max Planck Institute for Software Systems, Kaiserslautern, Germany, and the State Key Laboratory of VLSI and Systems, Fudan University, Shanghai, China. His research interests include energy Internet, machine learning, formal verification for software and hardware, cloud computing, IoT, and electronic design automation.

Kun Wang (M'13–SM'17) received the B.Eng. and Ph.D. degrees in computer science from the School of Computer, Nanjing University of Posts and Telecommunications, Nanjing, China, in 2004 and 2009, respectively.

From 2013 to 2015, he was a Postdoc Fellow with the Electrical Engineering Department, University of California, Los Angeles, CA, USA. In 2016, he was a Research Fellow with the School of Computer Science and Engineering, the University of Aizu, Aizu-Wakamatsu, Japan. He is currently a Research Fellow with the Department of Computing, the Hong Kong Polytechnic University, Hong Kong, SAR, and also a Full Professor with the School of Internet of Things, Nanjing University of Posts and Telecommunications, Nanjing, China. He has authored or co-authored more than 100 papers in refereed international conferences and journals. His current research interests mainly include the area of big data, wireless communications and networking, smart grid, energy Internet, and information security technologies.

Dr. Wang serves as an Associate Editor for the IEEE ACCESS, the Editor for the *Journal of Network and Computer Applications*, the *Journal of Communications and Information Networks*, *EAI Transactions on Industrial Networks and Intelligent Systems*, and a Guest Editor for the IEEE ACCESS, *Future Generation Computer Systems*, *Peer-to-Peer Networking and Applications*, and the *Journal of Internet Technology*. He was the Symposium Chair/Co-Chair of IEEE IECON16, IEEE EEEIC16, IEEE WCSP16, IEEE CNCC17, etc. He was the recipient of the Best Paper Award at IEEE GLOBECOM16. He is a Member of the ACM.

Lei He (M'99–SM'08) received the Ph.D. degree in computer science from the University of California, Los Angeles (UCLA), CA, USA, in 1999.

He is a Professor with the Electrical and Computer Engineering Department, UCLA, and a Guest Chair Professor with Fudan University, Shanghai, China. He was a Faculty Member with the University of Wisconsin, Madison, WI, between 1999 and 2002, consulted Cadence Design Systems, Cisco, Empyrean Soft, Hewlett-Package, Intel, and Synopsys, and was a Founding Technical Advisory Board Member for Apache Design Solutions and Rio Design Automation. He is a Co-Founder of Silicon Cloud International and the Founder of NxEco, Inc. He has authored or co-authored one book and more than 200 technical papers. His research interests include modeling and simulation, very-large-scale integration circuits and systems, Internet-of-things, and artificial intelligence.

Dr. He was the recipient of many best paper nominations and awards including the 2010 ACM Transactions on Electronic System Design Automation Best Paper Award and the 2011 IEEE Circuit and System Society Darlington Best Paper Award.