DPBRE: Demand Prediction and Bidding Responsiveness Enhancement in Smart Grids with Edge and Cloud Computing

Linna Ruan, Shaoyong Guo, Fushuan Wen, Senior Member, IEEE, Yong Yan, Xuesong Qiu, Senior Member, IEEE, and Rajkumar Buyya, Fellow, IEEE

Abstract—Demand Response (DR) in a smart grid is envisioned as a potential method to optimize energy management. However, due to limited capability for attaining dynamic electricity price and implementing demand side bidding, many customers do not participate in any DR program such as reducing or even interrupting some load demand in the peak load period. To ameliorate this status, this paper addresses demand prediction and bidding responsiveness enhancement (DPBRE) from the perspective of customers. First, an edge and cloud computing-based demand response (ECDR) architecture for a smart grid is proposed to achieve powerful computation capability and good real-time performance. Under the proposed architecture, the DPBRE solution is presented to break DR barriers, which includes a DE (Differential Evolution)*-CNN-LSTM based prediction (DCLP) algorithm, a demand bidding algorithm and a responsiveness oriented virtual resource management (RVRM) algorithm. These algorithms are implemented using iFogSim. Simulation results demonstrate the effectiveness of DPBRE on reducing more than 39% response delay during demand bidding compared to single mode of cloud computing or edge computing for the sample system. Besides, participation keeps increasing subject to acceptable response delay.

Index Terms—Demand Response (DR), Prediction and Bidding Responsiveness, DR Participation Rate, Edge and Cloud Computing, Virtual Resource Management.

I. INTRODUCTION

GLOBAL warming has aroused extensive concerns about environmental and energy issues. With the melting of glaciers, potential virus may be released, causing disaster to human being [1]. To address environmental problems, there are two effective ways, i.e. cutting off energy usage and enhancing energy efficiency. However, balancing the relationship between curtailing energy usage and the increased demand presents a big challenge. Demand response (DR) is viewed as a potential solution to address this challenge in a power system.

The two-way communication between the supply-side and demand-side proposed by Smart Grids facilitates a better DR. For suppliers, energy generation can be optimized with accurate knowledge of demands; For consumers, reward can be obtained by cutting off or shifting energy at peak periods. However, the DR is less attractive to users than expected, which is reflected by the low participation rate. One of the widely-admitted barriers is the low DR technology readiness levels [2], which is limited by physical infrastructures and information technologies.

There are three participation-related problems that can be alleviated with advanced technologies and well-designed mechanisms. 1) Load prediction accuracy. It directly affects price setting and users’ trust in DR information. 2) Prediction and bidding responsiveness. Intolerable delay generated will reduce users’ satisfaction with the DR process. 3) Knowledge of how benefit affects participation. Participation modeling guides real-time pricing and computing resource management.

The existing research mostly focused on strategy design, such as pricing methods [3]-[4], while the responsiveness and participation problems are less discussed. Compared with these ones, this paper discussed the above three problems and solved them with demand prediction and bidding responsiveness enhancement (DPBRE) solution. The main contributions of the paper can be summarized as follows:

1) A demand response participation rate (DRPR) model is proposed to give an exact evaluation of participation rate with benefit offered. Consumers’ electricity savings and satisfaction are combined as benefits and taken as participation affector.

2) A DE*-CNN-LSTM based prediction (DCLP) algorithm is designed to predict load hourly. It is the first time that the CNN-LSTM is optimized with DE, which shows better convergence performance with higher accuracy.

3) A responsiveness based virtual resource management (RVRM) algorithm is designed to realize containers scheduling among edge nodes. It aims to create and allocate containers to meet response delay limit with minimum energy consumption, which gives an edge-side virtual resource allocation strategy based on demand responsiveness for the first time.

The rest of the paper is organized as follows. Section II discusses the related work. Section III constructs the edge and cloud computing-based demand response architecture. Section IV describes system models. Section V depicts the DPBRE solution. Section VI shows simulation results. Section VII concludes the paper and provides future research prospects.

The authors would like to thank China Scholarship Council for supporting the visit to the first author to CLOUDS Lab at the University of Melbourne where this work is carried out.

Linna Ruan, Shaoyong Guo and Xuesong Qiu are with the Beijing University of Posts and Telecommunications, State Key Laboratory of Networking & Switching Technology, China (e-mail: linruan, syguo, xqiu)@bupt.edu.cn). Shaoyong Guo and Xuesong Qiu are the corresponding authors.

Yong Yan is with the State Grid Zhejiang Electric Power Research Institute, China (e-mail: 55682381@qq.com).

Fushuan Wen is with the School of Electrical Engineering, Zhejiang University, China (e-mail: fushuan.wen@gmail.com).

Rajkumar Buyya and also Linna Ruan are with the Cloud Computing and Distributed Systems Laboratory, School of Computing and Information Systems, the University of Melbourne, Australia. The email of Rajkumar Buyya is: rbuyya@unimelb.edu.au.
II. RELATED WORK

A large amount of research has been done for demand response process in smart grids. Most of them focuses on pricing strategies [3]-[4], load regulation [5], model and system design [6]-[8]. Efficient methods are proposed to improve DR performance, but it is still not clear what causes low participation rate and how to optimize these methods to address this problem.

Fortunately, there are some articles try to figure out the reasons, the challenges, and then provide DR development trend [9]-[12]. These articles indeed give some proper explanation based on internal and socio-economic factors, but do not provide an exact model of how these factors affect participation rate. They are able to provide limited help to guide DR development.

As for prediction, a function expected to be added in the future DR, has rarely been discussed in the existing research. [13] developed an artificial neural network based methodolgy for day-ahead prediction and shaping of dynamic demand. [14] proposed PSO-based CNN-LSTM method and explored the optimal energy prediction structure with various time series. But both of them only discuss prediction process, without a consideration of motivating customers with demand bidding. They also lack discussion on response delay which affects user satisfaction.

Demand bidding has received much attention recently. It encourages customers to participate in peak electricity reduction bids, compete for DR opportunities, and realize cost reduction. In [15], customers were divided into two types and then day-ahead demand bidding was formulated as a MILP problem. Mohsenian-Rad et al. [16] focused on day-ahead optimal bids for time-shiftable loads. However, most of the existing bidding strategies based on prediction result are implemented one day ahead. They indeed reduced the requirement for delay, but to some extent they lost the response accuracy. Real-time interaction is undoubtedly a better way.

The few research about participation are listed as [17]-[21]. Lee et al. [17] overviewed load participation in Electric Reliability Council of Texas (ERCOT) market. But they only discussed the status of ‘participation’ and did not give a method to evaluate the participation rate. In [18], participation rate has been mentioned, but it was taken as a factor for real-time pricing. [19] and [20] grouped users with participation capabilities evaluated with historical load profiles. But these methods missed flexibility when DR policy such as pricing strategy changed. In [21], an online dashboard was developed to enhance and persuade customers to participate in DR programs. A summary of the comparison between our work and other closely related works is given in Table I.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Related Work</th>
<th>DPBRE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Demand prediction</td>
<td>[9]-[12]</td>
<td>✔️</td>
</tr>
<tr>
<td></td>
<td>[13]</td>
<td>✔️</td>
</tr>
<tr>
<td></td>
<td>[14]</td>
<td>✔️</td>
</tr>
<tr>
<td></td>
<td>[15],[16]</td>
<td>✔️</td>
</tr>
<tr>
<td></td>
<td>[17]-[21]</td>
<td>✔️</td>
</tr>
<tr>
<td>Demand bidding</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Resource allocation for prediction</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Participation</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Level discussion</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Evolution prediction</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>
III. SYSTEM ARCHITECTURE

An edge and cloud computing-based demand response architecture (ECDR) for smart grids is shown in Fig. 1. The demand side uploads energy demand and bidding price to edge nodes with metering facilities such as AMI (Advanced Metering Infrastructure). Edge nodes collect and process these information. Cloud assists suppliers to make pricing strategy and supplement computing capability. Details of the main components are as follows.

A. Sensors and Actuators

Sensors and actuators assist smart grids in realizing intelligence, energy saving, safety and reliability [22]. The commonly used physical entities of sensors and actuators are AMI and breaker. They present a good solution to cope with some specific requirements of demand response, such as capabilities of real-time load monitoring, two-way data exchanging.

B. Edge Nodes

In ECDR architecture, edge nodes cooperate with cloud to realize demand collection, prediction, as well as bidding. Besides, algorithm complexity and computing capability which have the greatest impact on the response delay, are also considered. Virtual resource management also be implemented at the edge side, so as to reduce energy consumption and enhance resource efficiency.

C. Cloud

In traditional grids, data are sent to cloud for processing. In ECDR architecture, we plan to keep both the cloud computing and edge computing. Since edge computing outperforms on real-time response, we prefer edge nodes to collect and process demand information. Cloud computing, on the other hand, offers sufficient computing capability and a larger coverage, and can assist suppliers in formulating pricing strategy.

IV. SYSTEM MODEL

Notations used in this paper are summarized in Table II.

A. Demand and Electricity Price Interaction (DEPI) Model

In smart grids, demand prediction is the basis for price setting, while demand curtailment determines the adjustment of electricity price. In this model, we assume that each cell facilitates one aggregator, which issues load curtailment profile and a highest bidding price to customers. If customers choose to participate, they would submit hourly load curtailment bids and bidding prices. Edge and cloud computing are adopted to enhance the responsiveness. As load shedding is the major case in this paper, and it usually happens during the peak period, we choose 9am-23pm, the commonly admitted high demand period, to execute our model. This model is shown in Fig. 2.

![Fig. 2: Demand and electricity price interaction (DEPI) model.](image-url)

<table>
<thead>
<tr>
<th>TABLE II: NOTATIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Model</strong></td>
</tr>
<tr>
<td>$m$</td>
</tr>
<tr>
<td>$n$</td>
</tr>
<tr>
<td>$p_u^n$</td>
</tr>
<tr>
<td>$J_n$</td>
</tr>
<tr>
<td>$\beta$</td>
</tr>
<tr>
<td>$t$</td>
</tr>
<tr>
<td>$T$</td>
</tr>
<tr>
<td>$v_m^T$</td>
</tr>
<tr>
<td>$n_t^m$</td>
</tr>
<tr>
<td>$D_{m,T}$</td>
</tr>
<tr>
<td>$p_o$</td>
</tr>
<tr>
<td>$p_{m,T}$</td>
</tr>
<tr>
<td>$T_{m,p}$</td>
</tr>
<tr>
<td>$\varphi$</td>
</tr>
<tr>
<td>$\delta$</td>
</tr>
<tr>
<td>$u_{m,T}$</td>
</tr>
<tr>
<td>$u_{m,s}$</td>
</tr>
<tr>
<td>$T_m$</td>
</tr>
<tr>
<td>$U_{m,Y,k}^T$</td>
</tr>
<tr>
<td>$U_{m,N,k}^T$</td>
</tr>
<tr>
<td>$\nu_{m,k}^T$</td>
</tr>
<tr>
<td>$\rho_{m,N,k}^T$</td>
</tr>
<tr>
<td>$\Omega_{m,Y}^T$</td>
</tr>
<tr>
<td>$\Omega_{m,N}^T$</td>
</tr>
<tr>
<td>$\Omega_{m,n}^T$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Design</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M$</td>
<td>Population (Number of groups)</td>
</tr>
<tr>
<td>$D$</td>
<td>Dimension of solution space</td>
</tr>
<tr>
<td>$g$</td>
<td>Current evolutionary algebra</td>
</tr>
<tr>
<td>$G$</td>
<td>Maximum evolutionary algebra</td>
</tr>
<tr>
<td>$x_{j,min}^T$</td>
<td>Lower Bound of the $j^{th}$ component</td>
</tr>
<tr>
<td>$x_{j,max}^T$</td>
<td>Upper Bound of the $j^{th}$ component</td>
</tr>
<tr>
<td>$F$</td>
<td>Scaling factor</td>
</tr>
<tr>
<td>$z_i^g$</td>
<td>The $i^{th}$ individual of the $g^{th}$ generation population</td>
</tr>
<tr>
<td>$c_{i,j}$</td>
<td>Intermediates produced after mutation</td>
</tr>
<tr>
<td>$CR$</td>
<td>Crossover probability</td>
</tr>
<tr>
<td>$Time_{co}$</td>
<td>Time complexity of CNN-LSTM</td>
</tr>
<tr>
<td>$Time_{m,T}$</td>
<td>Prediction complexity of cell $m$ in $T^{th}$ DR process</td>
</tr>
<tr>
<td>$A_{m,n,T}^T$</td>
<td>Load curtailment submitted by user $n$ in $T^{th}$ DR process</td>
</tr>
<tr>
<td>$\mu_{m,n}^T$</td>
<td>Bidding price submitted by user $n$ in $T^{th}$ DR process</td>
</tr>
<tr>
<td>$\pi_{m,n}^T$</td>
<td>Decision of whether to choose user $n$</td>
</tr>
<tr>
<td>$Q_{m,T}^T$</td>
<td>Target load curtailment issued by aggregator $m$ in $T^{th}$ DR process</td>
</tr>
<tr>
<td>$T_{m,U}$</td>
<td>Upper limit response time of group $m$</td>
</tr>
<tr>
<td>$c_{i,j}$</td>
<td>Number of containers migrate from $j$ to $i$</td>
</tr>
<tr>
<td>$t_{m,n}^{\text{migr,max}}$</td>
<td>Maximum migration delay of node $i$</td>
</tr>
<tr>
<td>$C_T^j$</td>
<td>Maximum number of migration containers of $j$ in $T^{th}$ DR process</td>
</tr>
</tbody>
</table>
B. Level and pattern-based user grouping (LPUP) model

There is a great difference in users’ electricity using habits and capability, which affect the participation rate evolution. Hence, a grouping method is needed to gather users with high similarity. Using consumption level to reflect capability, and pattern to indicate using habits, a level and pattern-based user grouping (LPUP) model is built. Where the input is two values set according to the environment: the number of total groups (cells) and the capacity of each group. The model is formulated as below and executed monthly.

1) Demand level

\[ \ell_{n}^{v} \] indicates the electricity consumption of user \( n \) in \( v \)th day of one month. \( \max \sum_{v=1,...,30} \ell_{n}^{v} \) indicates the monthly consumption of the user who consumes the most. The normalized value of consumption per user is

\[ \frac{\sum_{v=1,...,30} \ell_{n}^{v}}{\max \sum_{v=1,...,30} \ell_{n}^{v}}. \]

2) Electricity usage pattern

\[ \ell_{n}^{v,u} \] indicates the hourly consumption of each day, and \( \max \ell_{n}^{v,u} \) means the daily peak load. We can calculate a load factor to represent the load pattern [23]. It is defined as

\[ \frac{\sum_{v=1,...,30} (\ell_{n}^{v,u} / \max \ell_{n}^{v,u})}{30}, \]

the average daily electrical energy consumption divided by the daily peak load. A high result means that the power usage is relatively constant, while a low one means that sometimes high power is demanded.

3) Demand judgement factor

The demand judgement factor can be calculated by combining the above normalized consumption and load factor, \( \beta \) is a weighting factor that indicates the relative importance of the two parts and is set according to the specific situation.

\[ J_{n} = \beta \left( \frac{\sum_{v=1,...,30} \ell_{n}^{v,u}}{\max \sum_{v=1,...,30} \ell_{n}^{v,u}} \right) + (1-\beta) \left( \frac{\sum_{v=1,...,30} (\ell_{n}^{v,u} / \max \ell_{n}^{v,u})}{30} \right). \] (1)

In this way, we can get the similarity difference of each pair of users, calculated as \( |J_{i} - J_{n}|, (i \in N, n \in N/i) \).

The grouping process is shown in Fig. 3. The core idea is that: First, obtain the similarity differences between each pair of users and then sort the pairs from small to large based on the values. Second, the two users with the smallest similarity difference are grouped and then they are deleted from the set that waiting to be processed. Third, select the pair with the highest similarity, and then judge whether one of the users belong to the existing groups. If so, judge whether there are groups that still has the capacity, if so, the other user of the pair will join the group that hold the smallest variance of \( \{J_{n}\} \) after including the new user; otherwise, make it another group. If both the users don’t belong to existing groups, judge whether there are groups that still have the capacity. If the answer is yes, hold both the plans, which means putting the users into an existing group that hold the smallest variance after including it or putting the users into a new group, then delete the pair from the set. If all the existing groups reach the maximum capacity or the number of users waiting for grouping is smaller than the groups that can be established, the pair will follow a new group. Further, cyclic from the third step. When one group reaches the capacity, judge the existing plans. Calculate the variance of each group and select the plan with the smallest total variance of all groups. This is an effective way to reduce the complexity of the model. When all users are grouped or the number of remaining users is greater than the groups that can be established, this process terminates.

C. DR Participation Rate (DRPR) Model

Participation is modeled based on benefit function, which consists of consumers’ electricity savings and satisfaction with DR. Users are grouped by adopting LPUP model. Consider that each group matches an edge node, so the group is the same as the edge node, denoted as \( m \).

1) Cost saving model

We adopt a widely used pricing method, that is, charging based on the response of the groups to DR. Taking load shedding as the major discussed DR strategy, if one group refuses to participate in DR, it will be charged at the original price \( P_{o} \). Otherwise, a lower price \( P_{m,T} \) is set according to the customers’ participation rate \( r_{m,T} \), with an insurance of higher than the generation cost. \( t \) indicates hour and \( T \) is...
iteration times (accumulated from the first day of DR). \( D_{m}^{\tau,T} \) represents demand prediction result. Hence, the cost saving can be modeled as:

\[
u_{m,T}' = D_{m}^{\tau,T} \times (P_o - P_{m}^{\tau,T}).
\]

2) Satisfaction model

Prediction and bidding delay represents request responsiveness and is used to evaluate user’s satisfaction. Extended from [24], satisfaction can be represented as:

\[
u_{m,s}^{\tau,T} = \varphi(e^{-r_{m,s}^{\tau,T}} + \delta) \times D_{m}^{\tau,T},
\]

where \( r_{m,s}^{\tau,T} \) means response delay, \( \varphi \) is the comfort benefit function coefficient. \( \delta \in (0,1) \) is a random variable, which reflects the disturbance influence of environmental factors.

3) Benefit model

Benefit is modeled with the above saving and satisfaction functions, shown as:

\[
u_{m}^{T} = \alpha_{m} \times \sum_{i=9,10,..,23} u_{m,T}^{\tau} + (1 - \alpha_{m}) \times \sum_{i=9,10,..,23} u_{m,s}^{\tau},
\]

where \( \alpha_{m} \) and \( 1 - \alpha_{m} \) indicate the relative importance of saving and satisfaction respectively for group \( m \).

4) Dynamic evolutionary game interaction (DEGI) model

Evolutionary game has successfully explained many complexing and challenging aspects of biological and social phenomenons [25]-[26]. The problem of DR participation has the characteristics of profit-seeking, which is suitable to be solved with evolutionary game theory.

Considering that a group represents a class of users, the decision of the group in this model can be denoted as \( X_{m} = \{x_{Y}, x_{N}\} , x_{Y} \) means the group of users choose to participate DR, while \( x_{N} \) means refuse to. Accordingly, if group \( m \) participates, the benefit is set as \( u_{m,Y,k}^{T} \), otherwise, the benefit will be \( u_{m,N,k}^{T} \). \( k \) indicates the number of combinations of group decisions. Since each group has two choices, the total number of combinations will be \( K = 2^{M-1} \). The benefit matrix of users in group \( m \) can be obtained as follows:

\[
U_{m}^{T} = \begin{bmatrix}
u_{m,Y,1}^{T} & ... & \nu_{m,Y,k}^{T} & ... & \nu_{m,Y,2^{M-2}}^{T} \\
u_{m,N,1}^{T} & ... & \nu_{m,N,k}^{T} & ... & \nu_{m,N,2^{M-2}}^{T} \\
\end{bmatrix},
\]

where \( \nu_{m,Y/N,k}^{T} \) represents the relative participation rate. It can be calculated with formula (7). This evolutionary game process can be represented by a full binary tree with depth of \( M \), and the number of leaf nodes is \( K \) as shown in Fig. 4.

Fig. 4: General decision and benefit evolution.
It can be easier extended to the general situation, the benefit of group \( m \) when selects strategy \( x \) after \( T \) iterations is:

\[
\Omega_{m,Y}^T = U_{m,Y,1}^T \star r_{m,Y,1}^T \star r_{m,Y,2}^T \star \ldots \star r_{m,Y_{m-1}}^T \star r_{m,Y_k}^T \star \ldots \star r_{m,T}^T
\]

Similarly, we can get the benefit of group \( m \) when selects strategy \( x \). The average benefit function is:

\[
\Omega_{m,Y}^T = r_m^T \Omega_{m,Y}^T + (1 - r_m^T) \Omega_{m,N}^T.
\]

In this evolutionary game, the aim is to observe the changes of various users’ participation rate. We can construct replication dynamics, which follows: to guide DR strategies:

\[
\frac{dr_m^T}{dt} = r_m^T \left( \Omega_{m,Y}^T - \Omega_{m,N}^T \right).
\]

The equilibrium point is \( r_m^T \) equals to 0 or 1 obviously. Since this paper consider that each group has initial participation, the balance point will be \( r_m^T = 1 \) definitely.

The upper limit of response delay caused by prediction and bidding can be obtained. This theorem is proved in APPENDIX A.

V. DEMAND PREDICTION AND BIDDING RESPONSIVENESS ENHANCEMENT (DPBRE) SOLUTION

The flowchart of DPBRE is shown in Fig. 5, which mainly contains six steps. 1) Demand day-ahead prediction and hour-ahead update. This step is implemented by sequential execution of DEPI model, DCLP algorithm and DB algorithm. 2) Level and pattern-based user grouping. This step is performed once a month, and users with similar energy usage characteristics are grouped based on LPUP model. 3) DR participation rate prediction. DRPR model is used in this step to get the knowledge of participation dynamic evolution. 4) Obtain the upper limit response. Using the information obtained in the previous three steps, we can calculate the upper limit delay, as proved in the APPENDIX. 5) Responsiveness oriented virtual resource management. This step aims to generate and allocate containers in subject to response delay limit with minimum energy consumption. 6) Judge whether the participation rate of the group reaches 1, if yes, jump out of the cycle, otherwise, proceed to the next round from step 1. The details of the three algorithms are introduced as follows.

### A. DE*-CNN-LSTM based Prediction (DCLP) Algorithm

1) **DE*-CNN-LSTM:** CNN is suitable for extracting data features, while LSTM is commonly used to deal with time series problems. Hence, CNN-LSTM can express features in these two dimensions and is suitable for load prediction. The differential evolution (DE) is a common method for global optimization problems [27]. It exhibits better convergence and is easier to understand due to two components distinguished from other evolution algorithms: the trial vector generation strategies and the control parameters [28]. In DCLP, DE is used to obtain the number of layers and units in CNN and LSTM, while guaranteeing the prediction accuracy is higher than 99%. The DE*-CNN-LSTM architecture is illustrated in Fig. 6. The blue rectangles represent CNN part, including Convolutional layer, Pooling layer and Dense layer. The green ones belong to LSTM. We use DE* to optimize layer selection, where ‘*’ means that the scaling factor of DE is adjustable, not a constant. The process of DCLP is shown in algorithm 1.

#### Algorithm 1: DCLP Algorithm

- **Input:** (CNN-LSTM) dataset; (DE): Population, \( M \); Dimension, \( D \); Generation, \( G \); Scaling factor, \( F_0 \); Crossover probability, \( CR \)
- **Output:** Units of LSTM(int(X[0][j])); Units of Dense(int(X[1][j]))
- **Initialization:** \( g = 1; \)
- **for** \( i = 1 \) to \( M \) **do**
- **for** \( j = 1 \) to \( D \) **do**
  \( x^{g}_{i,j} = x_{i,j}^{g-1} + \text{rand}(0,1) \times (x_{i,j}^{g-1} - x_{j}^{g-1}) \)
- **end**
- **end**
- **while** \( (||f_{fit}\_fun(X)||) > \varepsilon \) or \( (g \leq G) \) **do**
- **for** \( i = 1 \) to \( M \) **do**
- **for** \( j = 1 \) to \( D \) **do**
  \( (\text{Mutation}) \)
  \( u_{i,j}^{g+1} = x_{i,j}^{g} + \text{rand}(0,1) \times \lambda \times (x_{r2,j}^{g} - x_{r3,j}^{g}) \)
  \( (r_1 \neq r_2 \neq r_3 \neq i) \)
  \( F = F_0 \times 2^G \)
  \( \lambda = e^{1 - \frac{g}{G}} \)
  \( (\text{Crossover}) \)
  \( \text{if rand}(0,1) < R \text{ or } j = j_{\text{rand}} \text{ then} \)
  \( x_{i,j}^{g+1} = u_{i,j}^{g+1} \)
  \( \text{else} \)
  \( x_{i,j}^{g+1} = x_{i,j}^{g} \)
- **end**
- **end**
- **end**
- **if** \( f_{fit}\_fun(u_{i,j}^{g}) < f_{fit}\_fun(x_{i,j}^{g}) \) **then**
- \( x_{i,j}^{g} \leftarrow u_{i,j}^{g+1} \)
- **end**
- **if** \( f_{fit}\_fun(x_{i,j}^{g}) < f_{fit}\_fun(X) \) **then**
- \( X \leftarrow x_{i,j}^{g} \)
- **end**
- **else**
- \( x_{i,j}^{g+1} \leftarrow x_{i,j}^{g} \)
- **end**
- **return** the best position \( X(X[0][j], X[1][j]) \).

The core idea of this algorithm is 1) Obtain initial parameter settings from DE*. 2) Use the parameters to train the data with CNN-LSTM. 3) Judge whether the result of fitness function satisfies requirement or the generation times is smaller than the input. 4) If so, follow the steps of DE* (Mutation, Crossover and Selection), and then back to stage 2) to continue; Other-
2) Complexity Analysis: We are concerned about the impact of complexity on response delay, so the time complexity of DCLP algorithms is analyzed. The time complexity of CNN can be expressed as:

$$Time_c \sim O \left( \sum_{l=1}^{D} M_l^2 \cdot K_l^2 \cdot C_{l-1} \cdot C_l \right).$$

(15)

$M$ is the side length of each convolution kernel output feature graph. $D$ is the number of convolution layers. $l$ represents the $l^{th}$ convolution layer. $C_{l-1}$ and $C_l$ indicates the number of corresponding channels. The time complexity of LSTM is:

$$Time_{lstm} \sim O \left( nm + n^2 + n \right).$$

(16)

where $m$ means input size, $n$ is the hidden size. The overall time complexity of CNN-LSTM is:

$$Time_{c} = Time_{c} + Time_{lstm} \sim O \left( \sum_{l=1}^{D} M_l^2 \cdot K_l^2 \cdot C_{l-1} \cdot C_l + nm + n^2 + n \right).$$

(17)

B. Demand Bidding (DB) Algorithm

Hourly demand curtailment and bids submitted by customers are denoted as $A_{m,n}^t$ and $\mu_{m,n}^t$ respectively. The decision on whether to select the customer to join the DR process is $\bar{z}_{m,n}^t \in \{0,1\}$. Target curtailment issued by aggregator is $Q_m^t$. Since this trading aims to obtain the best combination of bids $Z^*$ that can minimize the total cost while meet target load curtailment, the process can be modeled as an ILP problem, shown as:

$$\begin{align*}
\min & \sum_{n=1}^{N} z_{m,n}^t A_{m,n}^t \mu_{m,n}^t \\
\text{s.t.} & \sum_{n=1}^{N} z_{m,n}^t A_{m,n}^t \mu_{m,n}^t > Q_m^t \cdot P_{o} - P_{t,m,n}^t.
\end{align*}$$

(18)

We expand the existing plane graphic method to multi-dimensional space vertex (MDSV) method to give a solution. There are $n$ customers in group $m$, and each customer corresponds to two decisions, that is, whether to be selected. The combination of decision set will be $2^n$ points in $n$-dimensional space. Firstly, all the points are verified according to the constraint conditions. Then, the satisfied points are brought into the objective function, and finally $Z^*$ of this 0-1 ILP problem is obtained. Since the bids submitted by customers is an individual decision, the calculation pressure is much lower than the global bidding decision-making process performed by edge nodes, thus the load response delay on the end-use device is omitted. The time complexity of DB is $Time_{db}^t \sim 2^n$.

C. Responsiveness Oriented Virtual Resource Management (RVRM) Algorithm

The DCLP algorithm guarantees accuracy, DB algorithm encourages more customers to join, and then RVRM is applied to meet the response delay requirements of given complexity. It mainly contains the following two steps.
1) Containers required per edge node: According to experience, delay changes proportionally to complexity (proved in Fig. 10). Virtual resource is allocated based on the initial value of complexity and the proportion difference. Two sets of initial response delays $\tau_{m}^1$ and $\tau_{m}^2$ are established respectively. The optimal solution of ILP problem is obtained. The whole process is shown in algorithm 2.

Algorithm 2: RVRM Algorithm

<table>
<thead>
<tr>
<th>input</th>
<th>Complexity of the prediction tuple $\text{Time}<em>{i,m}^T$; Upper limit of its responsive time $\tau</em>{m}^U$; Percentage of response delay improvement $\omega$</th>
</tr>
</thead>
<tbody>
<tr>
<td>output</td>
<td>Container scheduling scheme.</td>
</tr>
</tbody>
</table>

1. (Containers required per edge node)
2. for $m = 1$ to $M$ do
3. import $\text{Time}_{i,m}^1$, $\text{Time}_{i,m}^2$, $\tau_{m}^1$, $\tau_{m}^2$
4. get $\tau_{m}^\ast$ with (18)
5. if $\frac{\tau_{m}^1}{\tau_{m}^U} < 1 - \omega$ then
6. There is no need to do container migration of node $m$
7. else
8. the number of other same containers should be allocated is $\left\lfloor \frac{\tau_{m}^1}{(1 - \omega) \cdot \tau_{m}^U} \right\rfloor - 1$
9. another container allocated is specialized with $\left\lfloor \frac{\tau_{m}^1}{\tau_{m}^U} \right\rfloor$ times the original size.
10. end
11. end
12. (Container migration optimization)
13. get the relaxation problem of (19) by removing the integer constraint
14. solve (19) and obtain the optimal value $N_1 \leq c_{i,j} \leq N_2$
15. (Get Integer result with 3DSV method)
16. obtain solution set $S_1(i,j, N_1)$, $S_2(i,j, N_2)$
17. verify vertex combination of all the same original node according to the constraint
18. select the set that satisfy the condition
19. bring these sets into objective function
20. get the optimal solution that best fits the MO-ILP.
21. return the best solution $c_{i,j}$.

VI. PERFORMANCE EVALUATION

A. Simulation environment

We used iFogSim for simulation of a smart grid environment. Compared with other general simulators, iFogSim is customized for edge computing, simulating the environment well with edge nodes and IoT devices. It contains three basic kinds of components: physical components, logical components and management components [29]. Their relationship and the topology adopted in this paper is shown in Fig. 7.

Fig. 7: Realization of DPBRE using iFogSim.

The real dataset in the continental European electricity system is employed to make the simulation results more convincing, including hourly energy demand inflows recorded during 2012-2014 [30]. Firstly, we use the whole dataset to evaluate performance of the proposed DE*-CNN-LSTM with compared algorithms. Then, users are grouped with LPUP model and 4 groups are selected as an example and numbered with group 1-4 from the highest demand level to the lowest. Follow the construction shown in Fig. 7, we create one cloud, four edge nodes, each holds 20 clients, and one client associates one sensor as well as one actuator. The
proposed algorithms are modeled as events and added to edge nodes, which decide to process these events at edge or cloud according to the response delay predicted hourly.

B. Experiments and results

1) DCLP algorithm simulation–Prediction: This paper takes electricity demand as the main feature, so only the number of units and layers of LSTM and Dense needs to be optimized. The first step is for units. We take PSO-CNN-LSTM Prediction [14] and DE-CNN-LSTM Prediction (the scaling factor is unadjustable) as contrast algorithms. The parameters setting is shown in Fig. 8.

These algorithms are modeled and the simulation result is shown in Fig. 9. In the process of multi-times experiments, we found that the prediction accuracy can reach the target of higher than 99% within 30 iterations. The PSO-CNN-LSTM takes minimum time to complete genetic evolution, but its best accuracy is 93.18%; DE*-CNN-LSTM takes 843.7s, and its accuracy can reach to 99.01%, the optimal number of units for LSTM and Dense are 34 and 2. For DE-CNN-LSTM, the corresponding results are 1100.7s, 99.28%, 6 and 1. It is obvious that the latter two both meet the accuracy requirements, while DE*-CNN-LSTM adopted in this paper takes less time. Follow the same way, we found that one layer of LSTM and Dense are enough to meet our accuracy requirements. Now the architecture of DE*-CNN-LSTM is determined and the complexity of DCLP is calculated as:

\[ \text{Time}_o \sim O(nm + n^2 + n) \sim O(m). \]  \hspace{1cm} (21)

2) DCLP algorithm simulation–Response time: To evaluate the relationship between complexities and response time, we set training datasets for 4 groups, and test the delay of 2 groups to obtain the gain ratio. Their input sizes are 8400 and 17640, and the corresponding delay are 14,0204s and 22,2327s. Delays for other input sizes can be predicted with:

\[ T_m = \frac{22.2327 - 14.0204}{17640 - 8400} \ast (\text{Time}_o - 8400) + 14.0204 \]  \hspace{1cm} (22)

Eight groups of data with various complexity are tested and the result is shown in Fig. 10. The average accurate probability reached 99.0186%, which proves that the response delay is positively correlated with the time complexity. Besides, we test the predicted delay with cloud, shown as the red points. For the eight kinds of dataset inputs, processing with edge nodes reduces the response delay by 79%-85%.

3) DB algorithm simulation–Demand bidding incentive: Load curtailment ratio is set proportional to demand. For example, we set the curtailment ratio of 100kw be 10%. The maximum savings of each group are shown in Fig. 11. It can be seen that with cost saving as an incentive, users’ participation rate will increase, and then saving will in turn further increase. When \( r^*_m = 1 \), a steady state is reached.

4) DB algorithm simulation–Response time: Bidding process also brings complexity, which mainly depends on how many customers join the transaction. Before this simulation, we set the historical datasets sizes for group 1-4 with 17640, 14112, 11088, 8400, chosen from 8 tests in Fig. 10. The response delay is tested and the results of group 1 and 4 are shown in Fig. 12 as an example. The conclusion is that the combination of edge and cloud can shorten delay by 39%-93% compared to single cloud and has a greater impact on only edge processing when complexity keep rising.

5) Response delay requirements with participation rate model: Before this simulation, we set the parameters for benefit and satisfaction model. According to experience, in the benefit model, we set \( P_o = 0.58, P_m^T = P_o / (0.45 \ast n^T + 1) \); Also we set satisfaction model as \( u^T_{m,s} = 1.1(e^{-r_m^T + 0.5} \ast D^T_{m,s}) \), \( a_m \) is set with 0.85. After getting the value of \( u^T_{m,Y,k} \), we select the random value of \( r^*_m \) from (0,1). Then with (9) and the predicted \( D^T_{m,p} \), it is easy to obtain \( \Pi_{m,Y} - \Pi_{m,N} \). Referring to APPENDIX A, we can get the upper limit of response time. The result is shown in Fig. 13 with the delay decrease ratio \( \omega = 0.1% \) as an example.
Fig. 12: Response delay comparison.

Fig. 13: Upper limit of response time.

6) RVRM algorithm simulation: With the predicted and expected response time, it is easy to get the required number of containers. Then, RVRM algorithm is used to realize the migration of containers between edge nodes. The energy cost for container creation is set to 10 cents and for migration is 1 cent [31]. The result is shown in Fig. 14. Compared with all creation (AC) method, RVRM algorithm saves energy cost about 10.25%.

Fig. 14: Energy cost of virtual resource allocation.

7) Prediction of participation evolution: The prediction of participation enhanced is shown in Fig. 15. The final prediction of participation evolution is shown in Fig. 16. Four groups reached equilibrium through 35, 51, 145 and 307 iterations respectively, corresponding to four subgraphs. Each graph shows the trend of participation rate evolution before equilibrium point. Results verified that the participation rate can keep increasing under DPBRE solution, and it shows a clear idea about how can factors affect participation.

Fig. 15: Prediction of participation enhanced.

Fig. 16: Prediction of participation evolution.

VII. CONCLUSIONS AND FUTURE WORK

A demand prediction and bidding responsiveness enhancement (DPBRE) solution is proposed to break demand response barriers in a smart grid. In this solution, the DEPI model is used to formulate demand and electricity interaction, the DPLP model groups users according to energy consumptions levels and habits, and the DRPR model describes the exact connection between participation rate and benefits. Besides, DCLP, DB, RVRM algorithms are used to implement demand prediction, bidding and virtual resource management respectively. Simulation results demonstrate that compared with only cloud computing or edge computing, the DPBRE solution can reduce response delay by more than 39% for the sample system. Energy cost is saved by about 10.25% during container scheduling and the participation rate keeps increasing with acceptable response delay.

In this paper, the focus is on response delay and energy efficiency, without considering renewable energy generation, which will be addressed in our future research efforts. Besides, edge computing will also be explored in power distribution and transformation applications.

APPENDIX A

PROOF OF THE UPPER LIMIT RESPONSE THEOREM

To motivate more customers to join DR process, it is needed to guarantee $\frac{dr_m}{dt} > 0$, which means the condition to keep participation increasing is $r_m^T(\Omega^T_m,Y - \Omega^T_m) = r_m^T(1 - r_m)(\Omega^T_m,Y - \Omega^T_m,N) > 0$. As $r_m \leq 0, 1$, so the condition can be simplified as $(\Omega^T_m,Y - \Omega^T_m,N) > 0$. As:

$$\Omega^T_m,Y - \Omega^T_m,N = \sum_{k=1}^{M-1} (U_{m,Y,k}^T \Pi R^T_{k}) - \sum_{k=1}^{M-1} (U_{m,N,k}^T \Pi R^T_{k})$$

$$= \sum_{k=1}^{M-1} \alpha d_{m,k}^T \beta \pi N_{m,k} [e^{-r_{m,k}^T} - e^{-r_{m,k}^T N_{m,k} + r_{m,k}^T}]$$

Finally, we can get a form like:

$$A(e^{-a_1 r_m^T} - e^{-a_2 r_m^T}) + B(e^{-b_1 r_m^T} - e^{-b_2 r_m^T}) + ... < Q$$

where, $a_1 < a_2, b_1 < b_2,...$. With such computation tool (Matlab), we can get the upper limit of response time $r_m^{T,U}$.

REFERENCES


Yong Yan received the B.S. and the M.S. degrees from Chongqing University, Chongqing, China, and South China University of Technology, Guangzhou, China, in 2006 and 2009, respectively. He is currently pursuing his Ph.D. degree in electrical engineering in the College of Energy and Electrical Engineering at Hohai University, and working in State Grid Zhejiang Electric Power Corporation Research Institute, China. His current research interests are blockchain, artificial intelligence, and power system.

Xuesong Qiu (Senior Member, IEEE) was born in 1973. He received the Ph.D. degree from the Beijing University of Posts and Telecommunications, Beijing, China, in 2000. He is currently a Professor and the Ph.D. Supervisor with the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications. He has authored about 100 SCI/EI index papers. He presides over a series of key research projects on network and service management, including the projects supported by the National Natural Science Foundation and the National High Tech Research and Development Program of China.

Rajkumar Buyya is a Redmond Barry Distinguished Professor and Director of the Cloud Computing and Distributed Systems (CLOUDS) Laboratory at the University of Melbourne, Australia. He is also serving as the founding CEO of Manjrasoft, a spin-off company of the University, commercializing its innovations in Cloud Computing. He has authored over 750 publications and seven text books including "Mastering Cloud Computing" published by McGraw Hill, China Machine Press, and Morgan Kaufmann for Indian, Chinese and international markets respectively. Dr. Buyya is one of the highly cited authors in computer science and software engineering worldwide (h-index=137, g-index=304, 100, 600+ citations). Dr. Buyya is recognised as Web of Science "Highly Cited Researcher" for four consecutive years since 2016, IEEE Fellow, Scopus Researcher of the Year 2017 with Excellence in Innovative Research Award by Elsevier, and the "Best of the World", in Computing Systems field, by The Australian 2019 Research Review. Software technologies for Grid, Cloud, and Fog computing developed under Dr.Buyya’s leadership have gained rapid acceptance and are in use at several academic institutions and commercial enterprises in 50 countries around the world. He served as founding Editor-in-Chief of the IEEE Transactions on Cloud Computing. He is currently serving as Editor-in-Chief of Software: Practice and Experience, a long standing journal in the field established 50 years ago. For further information on Dr.Buyya, please visit his cyberhome: www.buyya.com.