# Demand Response Application as a Service: An SDN-based Management Framework

Ahmadreza Montazerolghaem, Mohammad Hossein Yaghmaee

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSG.2021.3139004, IEEE Transactions on Smart Grid



# Demand Response Application as a Service: An SDN-based Management Framework

Ahmadreza Montazerolghaem, *Student Member, IEEE,* Mohammad Hossein Yaghmaee, *Senior Member, IEEE*

***Abstract*—With an increase in the utilization of appliances, meeting the energy demand of consumers by traditional power grids is an important issue. The success of Demand Response (DR) depends conclusively on real-time data communication between the consumers and the suppliers. Hence, a scalable and programmable communication network is required to handle the data generated. We prove that the problem of DR global load balancing includes energy and data constraints is NP-hard. So, a dynamic and self-configurable network technology known as Software-defined Networking (SDN) can be an efficient solution. In order to handle DR communication challenges, an SDN-enabled framework for DR flow management is designed in this paper. This framework is based on two-tier cloud computing and manages energy and data traffic seamlessly. We also equip this framework with Network Functions Virtualization (NFV) technology. The proposed framework is implemented on a practical testbed, which includes Open vSwitch, Floodlight controller, and OpenStack. Its performance is appraised by comprehensive experiments and scenarios. Based on the results, it achieves low delay, a high throughput, and improves Peak to Average Ratio (PAR) by balancing the energy and data on the entire DR network.**

***Index Terms*—Two-tier cloud computing, DR softwarization, Software-defined networking, Demand response automation server (DRAS), Flow management, OpenFlow switches.**

## NOMENCLATURE

| | |
|---|---|
| $BDR(i)$ | Balance of DR in the period $i$ |
| $BT(i)$ | Balance of traffic in the period $i$ |
| $QoS(i)$ | Quality of service in the period $i$ |
| $G(i)$ | Total customers income in the period $i$ |
| $O(i)$ | Total incentive in the period $i$ |
| $Y(i)$ | Total penalty in the period $i$ |
| $F(i)$ | Total cost of electricity consumed in the period $i$ |
| $MB(i)$ | Maximum benefit in the period $i$ |
| $L$ | Numbers of customers |
| $K$ | Numbers of suppliers |
| $d^l(i)$ | Demand of applicant $l$ in the period $i$ |
| $s^k(i)$ | Supply of supplier $k$ in the period $i$ |
| $\varphi(i)$ | Customer income in the period $i$ |
| $\zeta(i)$ | Incentive in the period $i$ |
| $\omega(i)$ | Penalty in the period $i$ |
| $\eta(i)$ | Electricity prices in the period $i$ |
| $Ic^l(i)$ | Commitment rate in the period $i$ |
| $\bar{d}(i)$ | Number of demand message in the period $i$ |
| $\bar{s}(i)$ | Number of supply messages in the period $i$ |
| $p$ | Path $p$ |
| $T_p(i)$ | Traffic of path $p$ |
| $\delta_p^m(i)$ | Remaining resources of DRAS $m$ in path $p$ |
| $\delta_p^s(i)$ | Remaining resources of switch $s$ in path $p$ |
| $\delta_p^e(i)$ | Remaining resources of link $e$ in path $p$ |
| $\Theta^m(i)$ | Maximum resources of DRAS $m$ |
| $\Theta^s(i)$ | Maximum resources of switch $s$ |
| $\Theta^e(i)$ | Maximum resources of link $e$ |
| $r_p^m(i)$ | Resource consumption of DRAS $m$ in path $p$ |
| $r_p^s(i)$ | Resource consumption of switch $s$ in path $p$ |
| $r_p^e(i)$ | Resource consumption of link $e$ in path $p$ |
| $b_p$ | Bandwidth of path $p$ |
| $B_p$ | Maximum bandwidth of path $p$ |
| $d_p$ | Delay of path $p$ |
| $D_p$ | Maximum delay of path $p$ |
| $h_p$ | Length of path $p$ |
| $H_p$ | Maximum length of path $p$ |
| $\alpha, \beta, \gamma, \theta$ | DR balance coefficients |
| $\xi, \varsigma, \varrho$ | Traffic balance coefficients between DRASs, switches, and links |
| $\phi, \psi, \vartheta$ | Quality of service coefficients |

A. Montazerolghaem is with the Faculty of Computer Engineering, University of Isfahan, Isfahan, Iran. MH. Yaghmaee is with the Department of Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran. E-mail: a.montazerolghaem@comp.ui.ac.ir and hyaghmae@um.ac.ir.

## I. INTRODUCTION

SMART GRID (SG) is conceptualized as a combination of electrical network and communication infrastructure. Any SG infrastructure should support real-time, two-way communication between utilities and users. To manage millions of smart meters and massive data, in a reliable and scalable manner, utilities must develop this communication network management system [1]. In this respect, cloud computing is envisaged to play major roles of motivation in the design of the future SG. Cloud computing is an emerging computation pattern that provides on-demand facilities and shared resources [2]–[4]. The SG infrastructure needs to be deployed globally. A scalable software platform is needed in order to quickly integrate and analyze information streaming from multiple smart meters simultaneously, in order to balance the real-time demand and supply. Real-time energy utilization and pricing information can be shared [5].

DR is one of the vital applications of SG. In DR, supply and demand programs are utilized for energy balance [6]. Smart users can decide for themselves on how and when to use electricity. With the growth of Internet of Things (IoT) technology, it became possible to transfer power consumption information to Demand Response Automation Server (DRAS) through the existing communication infrastructure [7]. In DR, consumers change their consumption depending on the price of electricity. Using DR approaches, it is possible to decrease or shift energy consumption from peak hours to low demand periods (also called virtual power) [8]. Electricity consumption and price information is exchanged through the communication network. Therefore, it is essential to establish a reliable, scalable, and Quality of Service (QoS) communication network in DR [9], [10]. Such a network is very helpful from two perspectives:

1. Balance of data traffic in the DR communication network,
2. Balance of energy between suppliers and applicants.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSG.2021.3139004, IEEE Transactions on Smart Grid


In this respect, the present paper brings forward an SDN-based cloud communication infrastructure that achieves a simultaneous balance of DR energy and information.

To the best of our knowledge, there are no studies on a comprehensive approach combining DR energy balancing with DR traffic balancing. Therefore, the exploration of such an approach is crucial and timely, considering the quick expansion of DR applications and also the emergence of SDN and NFV.

Efficient use of cloud computing [11], SDN [12] and NFV [13] technologies could help solve the DR network management problem. Cloud computing can be a major breakthrough in the management of DR network by providing an integrated and shared platform of flexible and scalable resources of computation, communication, and storage. The SDN consists of three main sections called control plane, data plane and application plane. The control plane includes one or more controllers. The network brain is located in these SDN software controllers. These controllers manage the flow and make decisions. Also, the control plane is responsible for setting rules to manage forwarding devices located in the data plane. The data plane contains network devices such as routers and switches in such a way that they lack a control section or software for automated decisions. The application plane also has a set of applications such as routing, firewall, load balancing, monitoring, and more. Interface connection protocols are standard Open APIs, including OpenFlow protocol. SDN can provide a global view of network state, high flexibility, scalability, and easy and integrated management. In this case, network productivity increases dramatically [14]. NFV will take advantage of virtualization technologies to turn network functions and services (such as DR applications) into Virtualized Network Functions (VNFs). These will be implemented in software and executed as Virtual Machines (VMs) on commodity hardware and high-performance Physical Machines (PMs), namely Network Function Virtualization Infrastructures (NFVIs). NFVI leverages cloud computing technologies. The overall management of the structure is with NFV Management and Organization (MANO). A VNF consists of one or more virtual machines (VM) [15]. To utilize DR network functions there is no more requirement to provide special hardware (such as DRAS servers). VNF instances could rather be dynamically created, utilized or resized on demand.

### A. Motivations

SG is the combination of electrical and communication network. So, communication network has an important duty for reliable energy management. Most DR programs presume that there is a spotless two-way communication network between the utility company and customers. However, the real present communication network is not complete. Also, due to the intricate architecture of SG, resources (energy and data) optimization methods are challenging work. Similar to the energy scheduling, data traffic scheduling can also be the momentous one to maintain an appropriate data traffic rate in the SG. Massive information is generated from both the utility company and customers and passes through some intermediate nodes, such as switches. The handling of the vast amount of such data is challenging using the usual data management methods due to the different constraints (such as resources, routing, QoS requirements, etc.). Consequently, cloud computing and SDN are some of the best techniques to control such vast data to have a robust, reliable, and efficient large-scale DR communication network.

### B. Contributions

In this article, we prepare a systematic framework of integrating cloud computing applications in DR, in two aspects: energy management and data management in the SG architecture. In this regard, the most important contributions of this paper are the following:

- Mathematical modeling of DR global load balancing problem including energy and data traffic constraints, and proving that the problem is NP-hard in conditions of limited resources.
- Proposing three approaches for softwarization of DR cloud networks using SDN and NFV concepts:
  - The first approach: Cloud DR based on SDN and DRAS softwarization,
  - The second approach: Cloud DR based on SDN and physical DRAS,
  - The third approach: Cloud DR based on software-defined NFV and DRAS virtualization,
- Implementing and evaluating the function of the proposed frameworks in a real context and under various scenarios.

### C. Organization

Our article is organized in this fashion that we refer to recent related works in Section II. The system model and problem formulation are provided in Section III. Section IV describes the proposed frameworks and their details. Their implementation and performance evaluation are then discussed in Section V. Finally, the summary and future work are to be dealt with in Section VI.

## II. Related work

Bera *et.al* [2] provide an epitome of existing works integrating cloud computing in the existing SG architecture. From this survey, the use of cloud computing applications in SG is one of the useful methods to overcome challenges relevant to common power grid management. The coalition of cloud computing with SG is envisioned to be effective for evolving the SG architecture in terms of monitoring, computing, information management, and power management. In [16], the authors describe their experiences building a cloud-based platform for SG data-driven analytics. [17] has analyzed the packet loss of SG communication networks. To support novel DR programs, SG needs a wireless communication network with QoS. [18] studies the problem of providing QoS in terms of delay and packet error probability for wireless SG network. Also, the impact of packet losses during communication on DR has been studied in [19]. Yaghmaee *et.al* [20] propose a communication model to evaluate the communication efficiency of both the cloud-based DR and distributed DR. The authors prove that when the user datagram protocol (UDP) is leveraged to broadcast the DR messages, making the optimal solution unachievable. [21] tries to seek the relationship between packet loss and latency in the data communication and real-time match supply with demand. The outcomes display that packet loss and latency has influence on the matching between supply and demand. Therefore the mentioned parameters should be considered when developing DR methods. In [22], the authors summarize the basic requirements of communication infrastructures in SG. Scalability, efficiency, and reliability are critical to enabling SG communication infrastructures. A discussion of SG communication protocols is provided by Khalifa *et.al* [23]. [24] provides a method for performance evaluating of DR

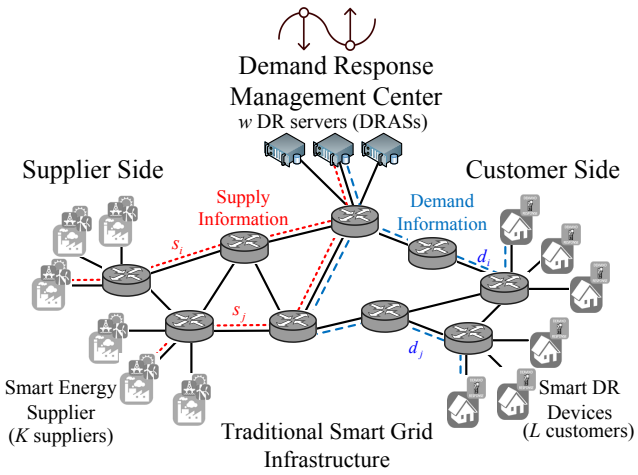



Figure 1. DR communication network (*energy* and *traffic* balancing).

communication protocols for the SG in combination with a DR strategy. A communication efficient distributed DR is proposed in [25]. While most of the existing distributed DR methods need iterative data exchange between the users and the utility company via two-way communications, the authors propose neighbor-wise communication between customers. A review of the application potential of 5G to DR is summarized in [7]. It provides guidelines for developing future 5G networks in the DR network. [26] provides an overview for SG improvement from the perspective of both power and communications. In addition, challenges in designing communications networks for the SG are figured out. A novel architecture of the power system network with an aim that information is aggregated and will be communicated between the users and utility company is proposed in [27]. Rinaldi *et.al* [28] introduce a communication architecture based on IP with customized SDN switches for SG. [29] has classified the routing protocols for SG communication networks also identified the pros and cons. This routing classification uses various vital metrics such as reliability, security, and QoS.

While much prior investigation has proposed the potential gains of applying SDN in communication networks in order to facilitate network management, there have only been few papers about the practical mechanisms of applying SDN in DR. Therefore, the exploration of such an approach is crucial and timely, especially considering the fast growth of DR applications and the advent of SDN.

## III. System Model and Problem Formulation

Consider a power network with some energy suppliers and customers and a communication network connecting them. As shown in Fig. 1, we suppose there are $K$ different numbers of suppliers in the system. Due to advances in Distributed Energy Resources (DER) and Distributed Generations (DG), these suppliers could be some DGs in the system without losing the generality. We also assume there are $L$ different numbers of customers in the power system. The amount of power supply generated by each supplier and the demand load of any customers are gathered through the existing communication network. This information is forwarded to the DR servers. As the number of customers increases, the traffic density is also increased. Although the primary responsibility of the DR program is to manage demand response solutions efficiently and smartly to achieve a balance in supply and demand, this will happen only when flexible and high-speed communication infrastructure is applied in the network. The DR program is successful when DR servers could deliver the supply and demand information ($s$, $d$) on time and lossless. So the main objective of the current work is to provide a balance not only in energy consumption (which is a goal of the traditional DR programs) but also in communication network traffic in the underlying communication infrastructure. This means that the DR objective (energy balancing) can only be achievable when the underlying network is not congested, and real-time traffic is delivered to the destination with minimum delay. Otherwise, when data traffic in the communication network is not balanced, it causes delays and loss of demand response necessary information.

According to the above description, our primary objectives can be summarized as follow:

- Balancing demand and supply in power network by utilizing peak shaving and load scheduling techniques.

- Traffic load balancing in communication infrastructure to provide quality of services for DR traffics. The goal is to transfer demand and supply information on time to the DR servers to make the DR program successful.

- Prevent any possible congestion and overload condition in DR servers by virtualizing DR servers in NFV and cloud computing environments to efficient use of resources.

Assume that a DR communication network can be modelled as a graph $G = (V, E)$, where $V$ illustrates the set of switches, and $E$ shows the set of links. Let $n = |V|$ and $m = |E|$, the number of switches and the number of links, respectively. This graph connects $K$ suppliers and $L$ customers to $w$ DR servers (DRASs) (Fig. 1). Each link uses $r^e$ units of resources. Resource consumption of each switch and DRAS is also denoted by $r^s$ and $r^m$, respectively. $\delta^m$, $\delta^s$, and $\delta^e$ represent the remaining resources of each DRAS, switch, and link.

Prior to proposing the approach, we prove that the problem of global load balancing in DR network is an Integer Linear Programming (ILP) problem and is, so, NP-hard.

**Proposition**: The global load balancing problem in DR network with limited resources is an ILP problem.

### A. DR Balancing

Suppose $d^l(i)$ represents the power load of each customer $l$ and at time period $i$. Lets $G(i)$, $O(i)$, $Y(i)$, and $F(i)$ represent the total customer's income, the incentive received by all customers, total penalty, and total cost of electricity consumption in period $i$, respectively. Then, the values of $G(i)$, $O(i)$, $Y(i)$, and $F(i)$ are calculated as follows:

$$G(i) = \varphi(i)(\sum_{l=1}^{L}(d^l(i) - d^l(i-1))) \quad (1)$$

$$O(i) = \zeta(i)(\sum_{l=1}^{L}(d^l(i-1) - d^l(i))) \quad (2)$$

$$Y(i) = \omega(i)(\sum_{l=1}^{L}(Ic^l(i) - (d^l(i-1) - d^l(i)))) \quad (3)$$

$$F(i) = \sum_{l=1}^{L}(\eta(i) \times d^l(i)) \quad (4)$$

where $\varphi(i)$, $\zeta(i)$, $\omega(i)$, and $\eta(i)$ are the revenue per one-unit change, the amount of incentive per one-unit change, penalty

and the cost of one-unit electricity consumption at period $i$, respectively. If the customer $l$ in period $i$ violates its obligations contrary to the contract ($Ic^l(i) - (d^l(i-1) - d^l(i))$) and does not reduce its consumption load, then the customer should pay penalties. $Ic^l(i)$ represents the commitment rate at period $i$. The following constraints make the demand not more or less than a certain maximum or minimum value.

$$\varphi_{min}(i) \leq \varphi(i) \leq \varphi_{max}(i) \tag{5}$$
$$\zeta_{min}(i) \leq \zeta(i) \leq \zeta_{max}(i) \tag{6}$$
$$\omega_{min}(i) \leq \omega(i) \leq \omega_{max}(i) \tag{7}$$
$$\eta_{min}(i) \leq \eta(i) \leq \eta_{max}(i) \tag{8}$$
$$d_{min}(i) \leq d(i) \leq d_{max}(i) \tag{9}$$

Suppose $MB(i)$ presents the maximum benefit in period $i$. The total benefit normalized in the period $i$, $BDR(i)$, is calculated as:

$$BDR(i) = \frac{\alpha G(i) + \beta O(i) + \gamma Y(i) + \theta F(i)}{MB(i)} \tag{10}$$

This benefit is the weighted sum of income, incentives, penalties, and electricity consumption costs. Using coefficients $\alpha$, $\beta$, $\gamma$, and $\theta$ one can have a combination of incentive and penalty programs with arbitrary weights.

### B. Traffic Balancing

Suppose path $p$ starts from the supplier or applicant and continues to DRAS while including a set of switches and links. Let $\delta_p^m(i)$, $\delta_p^s(i)$, and $\delta_p^e(i)$ represent the remaining resources of DRAS $m$ in path $p$, remaining resources of switch $s$ in path $p$ and the remaining resources of link $e$ in path $p$, respectively. The goal is to balance traffic throughout the DR communication network. To this end, the model seeks to find the best path $p$ for transmitting supply and demand messages over the network to DRAS. $P$ stands for all available paths ($p \in P$). The total normalized residual resources of period $i$ in the path $p$, $BT(i)$ are calculated as follows:

$$BT(i) = (\frac{\delta_p^m(i)}{\Theta^m(i)} + \frac{\delta_p^s(i)}{\Theta^s(i)} + \frac{\delta_p^e(i)}{\Theta^e(i)})u_p \tag{11}$$

where $\Theta_p^m(i)$, $\Theta_p^s(i)$, and $\Theta_p^e(i)$ are the maximum resources of DRAS $m$, the maximum resources of switch $s$, and the maximum resources of link $e$, respectively. The resources include DRASs, switches, and links resources. We are looking for a path with the maximum residual resources. $u_p$ is a binary variable that causes only one path to be selected from all paths. Suppose $r_p^m(i)$, $r_p^s(i)$ and $r_p^e(i)$ represent the resource consumption of DRAS $m$ in path $p$, resource consumption of switch $s$ in path $p$, and resource consumption of link $e$ in path $p$, respectively. Then, these variables are calculated as follows:

$$r_p^m(i) = \xi(T_p(i) \sum_{j \in p} |m_j|)u_p \tag{12}$$
$$r_p^s(i) = \varsigma(T_p(i) \sum_{j \in p} |s_j|)u_p \tag{13}$$
$$r_p^e(i) = \varrho(T_p(i) \sum_{j \in p} |e_j|)u_p \tag{14}$$

where $T_p(i)$, $\sum_{j \in p} |m_j|$, $\sum_{j \in p} |s_j|$, and $\sum_{j \in p} |e_j|$ represents the traffic in path $p$ at period $i$, the total number of DRASs, switches, and links in path $p$, respectively. The parameters $\xi$, $\varsigma$, and $\varrho$ are the traffic balance coefficients between DRASs, switches, and links.

Suppose $\bar{d}^l(i)$ and $\bar{s}^k(i)$ represent demand and supply messages, respectively. Then, the traffic in path $p$, $T_p(i)$, is calculated as follow:

$$T_p(i) = \sum_{l=1}^{L} \bar{d}^l(i) + \sum_{k=1}^{K} \bar{s}^k(i) \tag{15}$$

The following equations limit the resource consumption to remaining resources and the maximum resources of DRASs, switches, and links:

$$r_p^m(i) \leq \delta_p^m(i)u_p \tag{16}$$
$$r_p^s(i) \leq \delta_p^s(i)u_p \tag{17}$$
$$r_p^e(i) \leq \delta_p^e(i)u_p \tag{18}$$
$$\delta_p^m(i) \leq \Theta^m(i)u_p \tag{19}$$
$$\delta_p^s(i) \leq \Theta^s(i)u_p \tag{20}$$
$$\delta_p^e(i) \leq \Theta^e(i)u_p \tag{21}$$

### C. Quality of Service

As mentioned earlier, finding a path with low delay and high bandwidth is too essential to deliver demand and supply messages on time to the DR servers. Therefore, in addition to resource consumption, QoS is influential in route selection in the proposed system. Normalized QoS in period $i$ ($QoS(i)$) is calculated as the weighted sum of bandwidth ($b_p$), delay ($d_p$), and path length ($h_p$) as follows:

$$QoS(i) = \phi(\frac{b_p}{B_p}) - \psi(\frac{d_p}{D_p}) - \vartheta(\frac{h_p}{H_p}) \tag{22}$$

where $B_p$, $D_p$, and $H_p$ are the maximum bandwidth, maximum delay, and maximum length (hop count) of path $p$, respectively. The constant parameters $\phi$, $\psi$, and $\vartheta$ are the quality of service coefficients. The value of $b_p$ is set to the minimum link bandwidth in the path $p$:

$$b_p \leq b_{(i,j)}u_p \tag{23}$$

The path delay, $d_p$, is the summation of all link delays in the path $p$:

$$d_p = \sum_{(i,j) \in p} d_{(i,j)}u_p \tag{24}$$

The $h_p$ is calculated as the total number of links in the path $p$:

$$h_p = \sum_{(i,j) \in p} |\{i,j\}|u_p \tag{25}$$

Note that the following constraints should always be satisfied:

$$b_p \leq B_p, d_p \leq D_p, h_p \leq H_p \tag{26}$$
$$\sum_p u_p = 1 \tag{27}$$

According to the definition of $QoS(i)$, the paths with high bandwidth, low delay, and short length are more eligible to be selected. The $QoS(i)$ is always between zero and one and as mentioned earlier, constraint (27) guarantees that only one route will be selected.

### D. Optimization Problem

As mentioned earlier, although the primary responsibility of the DR program is to manage demand response solutions efficiently and smartly to achieve a balance in supply and demand, this

will happen only when the traffic is balanced in communication infrastructure and quality of service is provided. In this case, the demand and supply messages can be delivered to the DR servers efficiently. The following optimization problem is proposed to provide both *DR* and *traffic* balance in the network. The outputs of this problem are variable terms (e.g., $\varphi(i)$, $\zeta(i)$, $\omega(i)$, $\eta(i)$ as the customer income, incentive, penalty, and electricity prices in the period $i$). The rest of the terms are problem inputs (e.g., $\alpha, \beta, \gamma, \theta$ as the DR balance coefficients). $Ic^l(i)$ (commitment rate in the period $i$) and $s^k(i)$ (supply of supplier $k$ in the period $i$) are also inputs related to DR balance. $d^l(i)$ (demand of applicant $l$ in the period $i$) is the corresponding output. Resource consumption of DRAS $m$, switch $s$, and link $e$ in path $p$ ($r_p^m(i)$, $r_p^s(i)$, and $r_p^e(i)$) are outputs and maximum resources ($\Theta(i)$) are inputs related to traffic balance. About QoS, bandwidth, delay and length of path $p$ ($b_p$, $d_p$, and $h_p$) are the outputs and their maximum ($B_p$, $D_p$, and $H_p$) are the inputs.

$$\textbf{\textit{Maximize}} \quad BDR(i) + BT(i) + QoS(i) \tag{28}$$

***Subject to:***

$$\sum_{l=1}^{L} d^l(i) \leq \sum_{k=1}^{K} s^k(i) \tag{29}$$

$$\textit{Eqs.} \quad (1) - (27) \tag{30}$$

***Variables:*** $u_p \in \{0,1\}, 0 \leq BDR(i), BT(i), QoS(i) \leq 1,$
$G(i), O(i), Y(i), F(i), d(i), \varphi(i), \zeta(i), \omega(i), \eta(i) \geq 0,$
$\delta_p^m(i), \delta_p^s(i), \delta_p^e(i), r_p^m(i), r_p^s(i), r_p^e(i), T_p(i), \bar{d}^l(i), \bar{s}^k(i) \geq 0,$
$b_p, d_p, h_p \geq 0.$

**Proof**: The objective function (Eq. (28)) is to maximize the energy balance ($BDR(i)$) and traffic balance ($BT(i)$) as well as the quality of service ($QoS(i)$). Constraint (29) limits total demand to total supply. Although the objective function and constraints are linear, the binary variable $u_p$ converts the proposed model an ILP. Hence, the problem is generally NP-hard [30], [31].

To dominate this obstacle and reduce the complexity, an SDN-based framework is proposed and provides DR services in the cloud. We decompose the problem into subproblems as SDN applications in the application plane and benefit from SDN ability to cater a global view of the entire network and solve them.

## IV. Proposed Frameworks

Our purpose is to move toward *Software-defined DR Networking* based on the cloud computing in three approaches. Currently, the communication infrastructure between the supplier and the energy applicant is made up of a network of traditional switches. As the number of applicants increases, imbalances occur for both load (energy) and network traffic. As a result, neither energy nor network traffic is balanced over time. In the case of energy, this unnecessarily decreases the consumption or unreasonably increases the price of electricity. In the case of network traffic, it causes delays, loss of supply and demand information, as well as a lack of optimal use of DR server (DRAS) resources.

In the following, in the first proposed approach, we transfer the DRAS hardware function from the data plane over to the control plane and softwarize the DR balance. In the second approach, we use SDN in order to balance traffic between physical DRASs. In the third approach, DRASs are provided virtually under the control of NFV MANO. Details of these three approaches are given below.

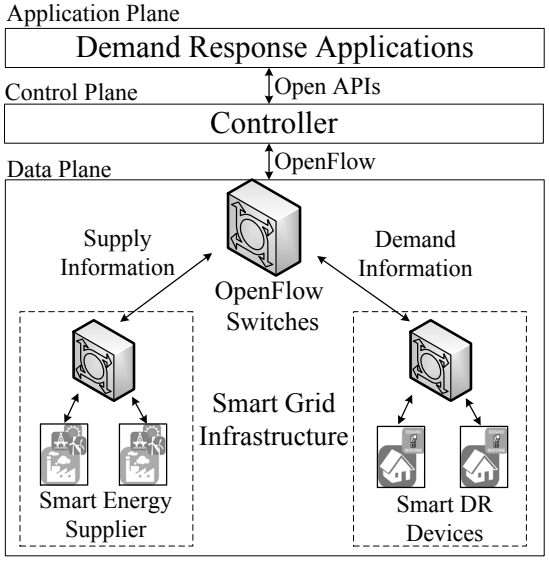



Figure 2. Proposed architecture for SDN-based DR communication network.

### A. SDN-based cloud DR and DRAS softwarization

In this paper, we present a communication network for DR based on SDN (Fig. 2). The infrastructure level includes energy suppliers, smart DR devices (such as home appliances) and OpenFlow switches. Their generated data, such as supply and demand information, is transmitted to the controller via OpenFlow switches for decision making. DR is composed of a large number of suppliers and DR devices that generate large amounts of data. Therefore, in order to integrate energy and information management, a centralized level of control is needed. In addition to DR data, the controller uses OpenFlow messages to gather updated information about network switches, such as the available capacity of switches and links. For this purpose, it demands diverse statistics from the network switches by transmitting `FEATURE-REQUEST` packets, and in return, the network switches return `FEATURE-REPLY` packets containing the requested statistics. Also, the network topology is taken using the Link Layer Discovery Protocol (LLDP). Different DR applications could be handled at the application plane. OpenAPIs such as the OpenFlow protocol makes interface connections. The controller has a global view of the entire DR network with OpenFlow. The controller is accountable for managing the DR network. Its timing diagram is shown in Fig. 3. At the beginning of each time interval $\tau$, each DR and supplier device transmits supply and demand data to the controller via the OpenFlow switches (data collection phase at time $t_g$). Depending on the data received and the network state, the controller calculates and notifies the switches of the optimal values of load (energy) and network traffic at time $t_c$ (calculation phase) and at time $t_n$ (notification phase). The controller then enters idle mode. In

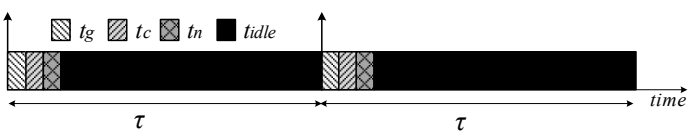



Figure 3. The time sequence of the proposed framework.

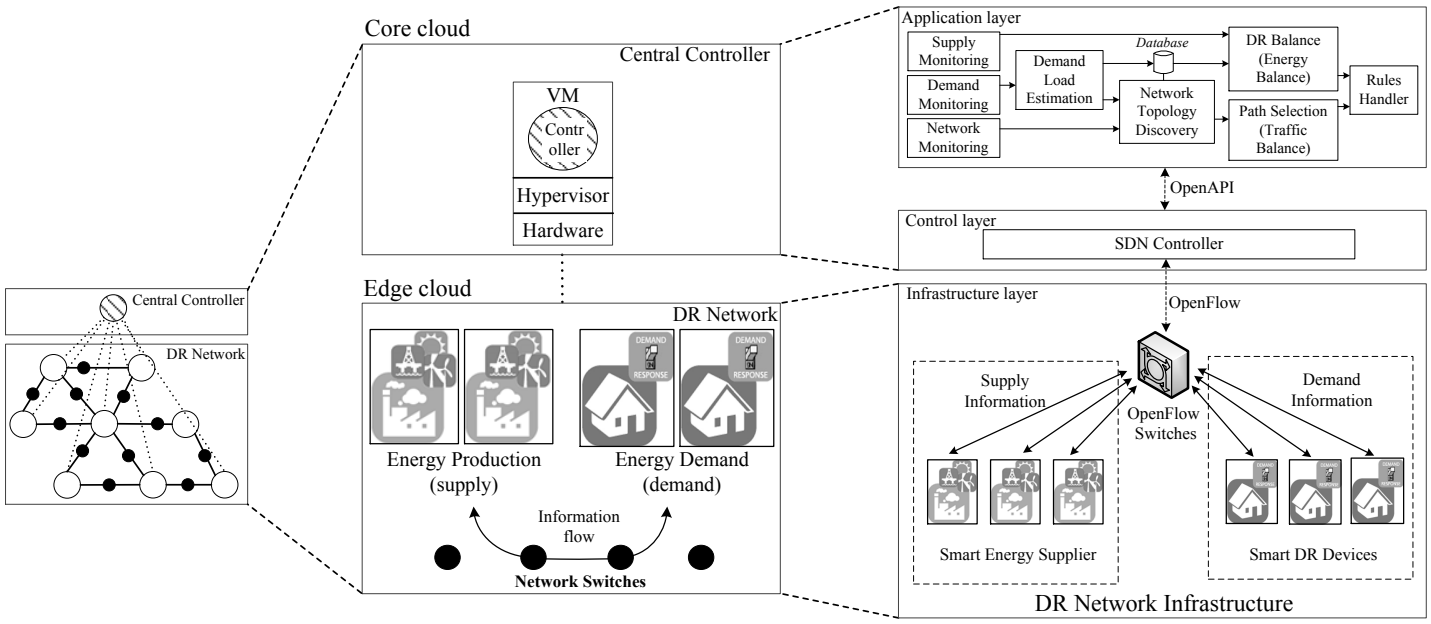



Figure 4. Details of SDN-based cloud DR architecture.

this method, the times $t_n$ and $t_g$ could be ignored. DR devices and energy supplier are certainly operating non-stop at every single time intervals $\tau$ (service phase). DR requests that reach the controller at time $\tau$ must wait for admitting and processing until the beginning of the next $\tau$ time. This time could be ignored relative to the total time and could also be reduced by shortening the time $\tau$.

The details and modules of the proposed framework are shown in Fig. 4. This approach is based on two-tier cloud computing: edge cloud and core cloud. The edge cloud includes all DR equipment and its communication network (infrastructure plane). The core cloud includes a central controller to control and decide on energy balance and traffic balance (control plane). Supply and demand information is collected from the infrastructure plane and provided to the control plane. The control plane with a global view could run all kinds of DR applications as software. The application plane has modules that we describe in continue. *Network Monitoring*, *Demand Monitoring*, and *Supply Monitoring* modules monitor and collect statistics on network, demand, and supply, respectively. Using this data, the *Demand Load Estimation* module estimates the demand in the next $\tau$ and stores them in a database. The *Network Topology Discovery* module obtains the new state as well as the map of the DR network using the Link Layer Discovery Protocol (LLDP) and provides it to subsequent modules. Given the estimated demand, the *DR Balance (Energy Balance)* module creates a trade-off between supply and demand. The *Path Selection (Traffic Balance)* module also balances traffic between network switches and links by selecting the appropriate paths for supply and demand information flows in the DR network. Finally, the *Rules Handler* module is responsible for creating the appropriate rules. This allows the controller to transmit rules to the switches via `Flow-Mod` packets. We will discuss the most important details of the modules in the following.

*1) Demand and Supply Monitoring:* In the SG, monitoring the total power consumption is crucial for determining the capacity needed and for the success of demand response programs. Furthermore, by the advances in DER and DG, DERs are widely used as energy suppliers in the SG. There are many dynamic data related to the DER systems that change frequently depending on the system and grid operating conditions. Some of these data are as follows: total power consumed by the customer, total power exported from the DER sit, total power imported from the grid to the DER site, active/reactive power generated by each DER, and the total power consumed/stored by each DER. In the proposed architecture, the amount of power supply generated by each supplier and the demand load of any customers are gathered through the existing SDN-based communication network. This information is used for DR balancing in the system.

*2) Network Monitoring:* The main objective of the current work is to provide a balance in energy consumption and communication network traffic in the underlying communication infrastructure. This means that energy balancing can only be achievable when the underlying network is not congested, and demand and supply-related data is delivered to the destination with minimum delay. Network monitoring is an essential component in network management. Using this module, we can determine the behavior of the network and the status of its components. It is a critical component because other services such as traffic engineering, quality of service, and routing also depend on it. This module constantly monitors the underlying communication network to check if it is running correctly. It can also optimize data flow and check the network availability. In the proposed architecture, using the OpenFlow protocol, we can provide the controller with the capability to know the network configuration, the paths, and the DR endpoints (customers and suppliers).

*3) Demand Load Estimation:* In the SG, demand load estimation is critical due to its applications in the planning of demand-side management, storage maintenance and scheduling, and renewable energy sources integration. Many factors such as weather conditions, time of the day, electricity prices, demand response, renewable energy sources, and storage cells affect the electricity demand in the SG environment. In the proposed architecture, we

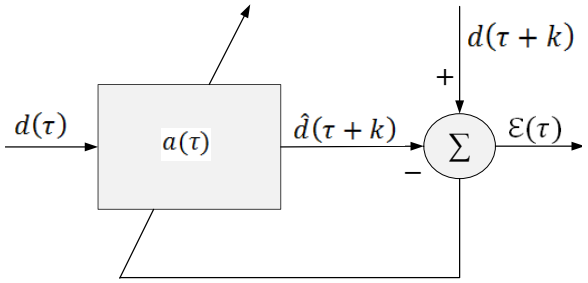


Figure 5. NLMS-based DR forecasting system.

use time-series analysis to find the iterator pattern in demand load and estimate future values. To apply time-series analysis methods, demand values must be sampled at specified time intervals. The outcome is a time series containing a sequence of recent demands. So, we focus on a proactive method that predicts future demand load (in next $\tau$) with respect to past demand load (in previous $\tau s$): Time-series analysis method. As a result, electricity pricing is already on the agenda. The Normalized Least Mean Square (NLMS) algorithm (Fig. 5) is one of the best methods in this regard, as it could make a trade-off between complexity, accuracy and responsiveness. The formulation of the algorithm for DR is as follows.

$d(\tau)$ is the amount sampled from demand at time $\tau$. Suppose we have a vector of $\upsilon$ demand, in the form $\bar{D} = [d(\tau), d(\tau - 1), ..., d(\tau - \upsilon + 1)]$. The estimated value of $d$ which is $\hat{d}$, is then obtained by the predictor function $\Gamma(\bar{D})$. To reduce complexity, we use linear predictors such as Eq. (31). $a(i)$ is a weight vector in this equation.

$$\hat{d}(\tau + k) = \Gamma(\bar{D}) = \Gamma(d(\tau), d(\tau - 1), ..., d(\tau - \upsilon + 1)) = \sum_{i=0}^{\upsilon - 1} a(i) d(\tau - i) \tag{31}$$

To calculate $a(i)$, the mean square error or the mean statistical error ($\mathbb{E}[\epsilon^2(\tau)]$) must be minimized. This mean is calculated in accordance with the Eq. (32). According to Eq. (33), the values of $a$ are determined in such a way that the said mean is minimized.

$$\mathbb{E}[\epsilon^2(\tau)] = \mathbb{E}[(d(\tau + k) - \hat{d}(\tau + k))^2] \tag{32}$$

$$\frac{\partial \mathbb{E}[\epsilon^2(\tau)]}{\partial a} = 0 \tag{33}$$

Because the data is constantly changing, the values of $a$ must also be constantly updated. To do this, we use the recursive Eq. (34) or its normalized Eq. (35).

$$a(\tau + 1) = a(\tau) + \mu \epsilon(\tau) d(\tau) \tag{34}$$

$$a(\tau + 1) = a(\tau) + \mu \frac{\epsilon(\tau) d(\tau)}{||d(\tau)||^2} \tag{35}$$

In these equations, $\mu$ is the step size and constant ($0 < \mu < 2$). To tune the parameters $\upsilon$ and $\mu$, we carried out various experiments. Values 30 and 0.8 were respectively considered for them. With those values in hand, the forecasting system reacted faster to changes in the demand load.

*4) Network Topology Discovery:* Managing networks in SDN requires the SDN controller to have fresh information about the network condition, especially topology. The SDN controller should have information about every device in the networks and the communication channels which link them. The OpenFlow discovery protocol, which facilitates the topology discovery, provides a vision about the network topology in the SDN network. In the proposed architecture, the network topology discovery module obtains the new state and the map of the DR network using the Link Layer Discovery Protocol (LLDP) and provides it to subsequent modules. As the proposed framework tries to balance energy and data traffic in the network, this module is crucial to provide the necessary information on the network topology, links, and switches.

*5) DR Balance (Energy Balance):* This module is used for balancing supply and demand in the power grid. Customers engage in demand response programs by different approaches such as time of use pricing, critical peak pricing, variable peak pricing, real-time pricing, and critical peak rebates. It also includes direct load control programs. The proposed DR approach is based on time of use pricing and load scheduling. It allows consumers to participate in the DR program by reducing or shifting their electricity usage during peak periods in response to time-based rates. So, we design the DR balance module. According to the output of the previous module, three states could be considered for demand: on-peak, off-peak and mid-peak. Pursuant to Fig. 6, in order to balance the DR, we have designed a Finite State Machine (FSM) that tries to adjust the next $\tau$ price of electricity in accordance with the load. Detection of each of these three states is based on the threshold. The maximum demand load is $\mathbb{D}$. If the predicted demand load ($\hat{d}(\tau + 1)$) is greater than the $\lambda\mathbb{D}$ threshold, then it is in on-peak state and consequently the price of electricity should increase. If the load is less than $\Upsilon\mathbb{D}$ then it is in off-peak state and the price of electricity should decrease. Finally, if the load is between $\Upsilon\mathbb{D}$ and $\lambda\mathbb{D}$, it is in mid-peak state and the price of electricity does not change. In this fashion, with the appropriate change in the electricity price of next $\tau$, demand is also balanced over time ($0 \leq \lambda, \Upsilon \leq 1$).

*6) Path Selection (Traffic Balance):* The main objective of this module is to choose an appropriate path for the traffic flow. With the emergence of SDN, path selection moves from the distributed network nodes to a centralized controller. Unlike the legacy network, where each network node computes the shortest path from the node to all destinations, in SDN networks, the controller knows the complete network topology and sets up paths for all possible source and destination pairs. Note that the information required for path selection is already provided by the network topology discovery module, as explained earlier. In this paper, in addition to balancing DR energy, we seek to balance DR network traffic, too. We pursue this in such a way that supply and demand messages are properly routed between switches and network links. In this regard, OpenFlow switches are configured in such a way that the flows pass through the routes with the least traffic.

*7) Rules Handler:* In the SDN architecture, the OpenFlow switches are responsible for matching packets with one or more flow tables. A flow table contains flow entries, and packets are matched based on the matching precedence of flow entries. The SDN controller computes and installs the flow rules in the flow table at switches. When a switch receives the data packet and does not have the flow rule in its flow table, the switch contacts controller to set up a new rule. This module is responsible for creating the appropriate rules. This allows the controller to transmit rules to the switches via `Flow-Mod` packets.

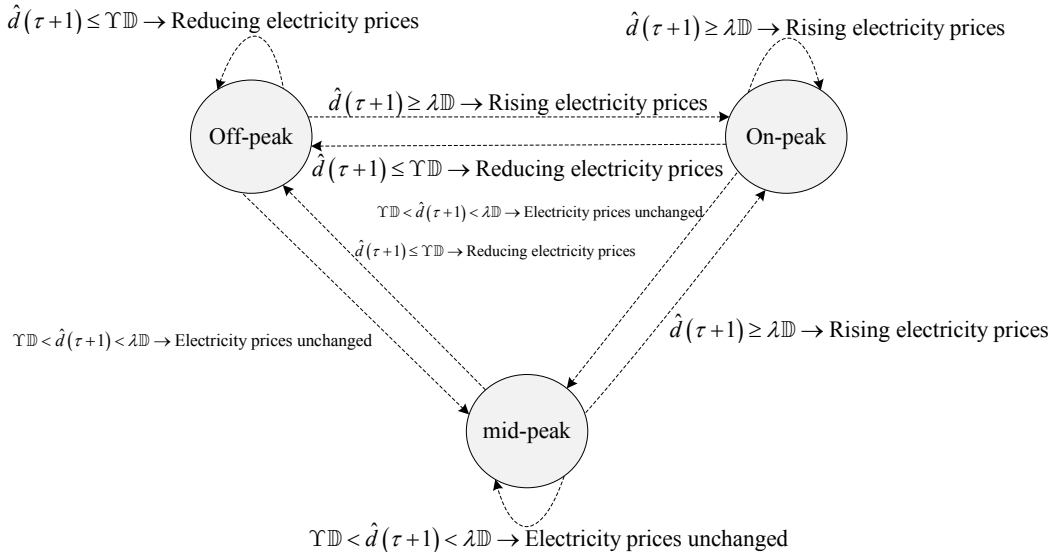



Figure 6. *DR Balance* finite state machine (dynamic pricing).

### *B. SDN-based cloud DR and physical DRAS*

In this section, we develop the previous framework assuming that DRASs are physical (Fig. 7). The DR Balance operation in this case, as mentioned in the previous section, is performed by physical DRAS (and no more softwarization) using the global information contained in the controller. In addition, network traffic (supply and demand messages) must be distributed among DRASs. For this reason, the three modules: *Admission*, *DRAS Selection* and *Path Selection* are added to the application plane. The *Admission* application decides on whether new demands will be admitted or ignored, given the overall capacity of DRAS. The *DRAS Selection* module uses response time as the load measure of each DRAS. This module selects a DRAS with the minimum response time to process the admitted demand messages. The role of the *Path Selection* module is to discover the shortest path to the chosen DRAS that satisfies the resource constraints of the switches and links. In the following, we examine their most important details.

*1) Admission:* The *Admission* module, as shown in Fig. 8, is implemented on the basis of a FSM. Assume that the capacity of each DRAS is $c$ and the whole capacity of the DRASs is $\mathbb{C}$. $c$ means how many requests could be processed by a DRAS per unit of time. Also, the total number of all DRAS requests in a specified unit of time is indicated by $\mathbb{N}$. In other words, $\mathbb{N}$ is a counter which increases by one unit when a demand message is admitted. $\mathbb{C}$ and $\mathbb{N}$ are the inputs of this FSM; and the output is one of these three actions: accept, drop, or drop with a specified probability. In addition, this FSM has three states: normal, overload and underload.

If $\mathbb{N}$ is less than $\Omega\mathbb{C}$, DRASs are in an underload state and all new demands are admitted. If $\mathbb{N}$ is greater than $\Omega\mathbb{C}$ and less than $\Psi\mathbb{C}$, it is in a normal state. In that case, the physical DRASs are not overloaded. However, to eschew an overload, a percentage of demand is dropped with a probability of $\frac{\mathbb{N}-\Omega\mathbb{C}}{\Psi\mathbb{C}-\Omega\mathbb{C}}$. Finally, if $\mathbb{N}$ is greater than $\Psi\mathbb{C}$, it is in an overload state. As a result, all received demands are dropped in order to stop saturation of DRAS resources ($0 \le \Psi, \Omega \le 1$).

*2) DRAS Selection:* This module is responsible for selecting the best DRAS to process the admitted demand messages; so that the workload is fairly distributed among the DRAS. For this purpose, it uses the response time measure. New demand messages are routed to DRAS with the shortest response time. This balances the traffic between DRASs.

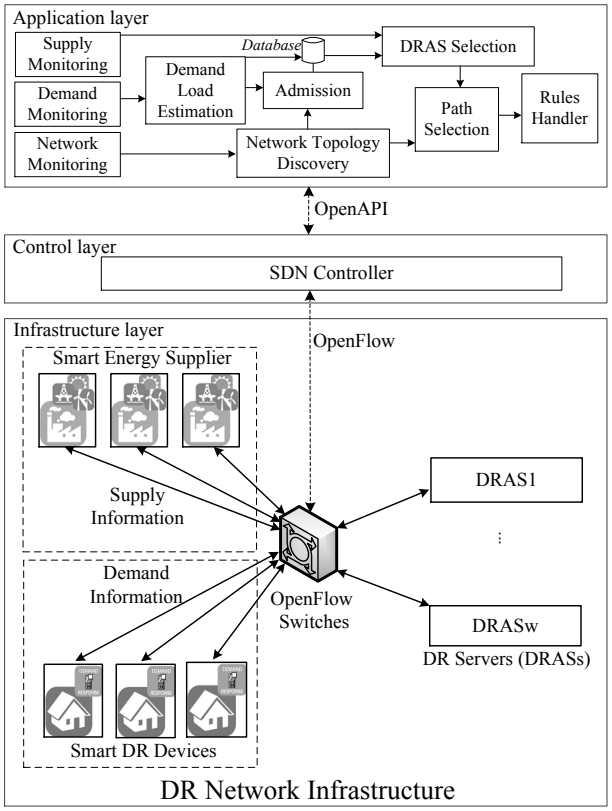



Figure 7. Cloud DR architecture based on SDN and physical DRAS.

*3) Path Selection:* Selected DRAS and network topology statistics (including switches and links) are the inputs to this module. In this module, the path with the least traffic to the selected DRAS is obtained using the *Least-load* algorithm.

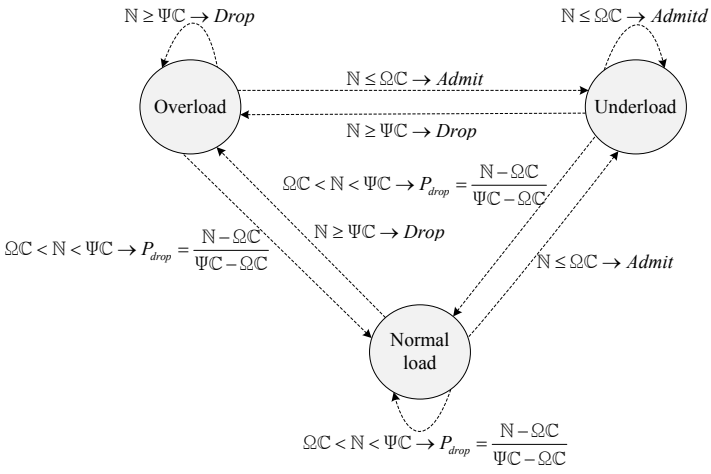



Figure 8. The finite state machine of the *Admission* module.

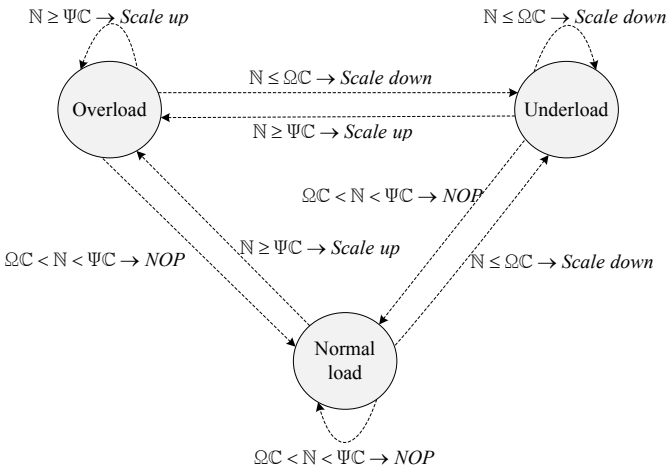



Figure 10. *Virtual DRAS Scaling* module finite state machine.

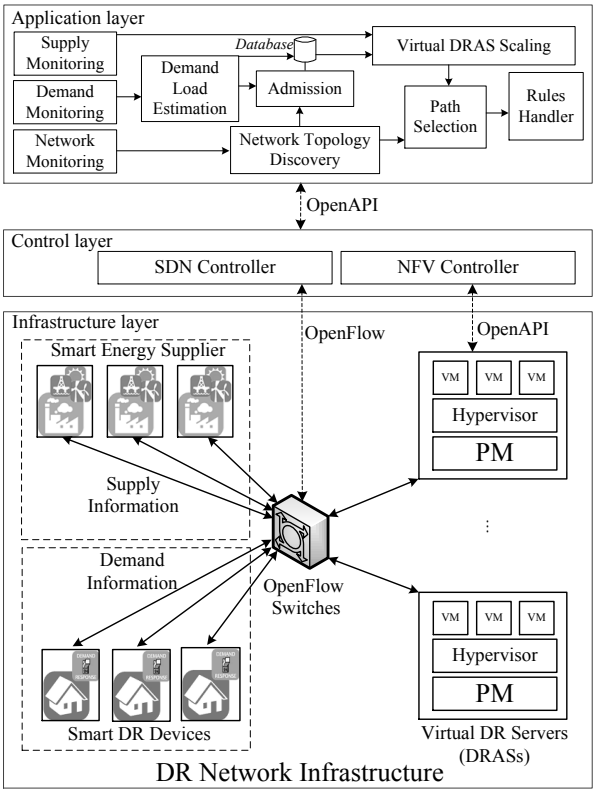



Figure 9. Cloud DR architecture based on SDN and DRAS virtualizations.

### C. NFV-based cloud DR and DRAS virtualization

Network function virtualization could help SDN overcome hardware constraints by virtualizing network equipment and functions. There is centralized management in the software-defined NFV architecture which not only focuses on OpenFlow switches but also on VNFs. In this architecture, due to the integrated control and monitoring systems it is possible to make decisions about DR balance and traffic balance on demand and according to changes in the network input load. Resources could also be scaled up or down in VM format.

Next, the framework proposed in the previous section is developed in such a way that DRAS virtualizations and their resources could be scaled. Resource scaling is a manner for adjusting resources in response to changing demands. As demonstrated in Fig. 9, in this proposed framework all network resources (such as switches) are controlled by the SDN controller. While virtual DRASs and their resources are controlled by the NFV controller, the controller stores all the essential information about those virtualizations. At the application plane, we have the new *Virtual DRAS Scaling* application. Its task is to decide on whether to scale virtual DRASs with a new FSM based on the input load.

*1) Virtual DRAS Scaling:* Fig. 10 shows the proposed FSM of this module. $\mathbb{C}$ and $\mathbb{N}$ are inputs of this state machine and its output are one of the three actions scale up, scale down or no operation. The state machine also has three states: overload, underload and normal load.

If $\mathbb{N} \leq \Omega\mathbb{C}$, then it is in underload state and the virtual DRAS with the minimum response time is scaled down. If $\mathbb{N} \geq \Psi\mathbb{C}$, then it is in overload state and the virtual DRAS with the maximum response time is scaled up. Finally, if $\Omega\mathbb{C} \leq \mathbb{N} \leq \Psi\mathbb{C}$, then it is in normal load state and no operation (NOP) is launched.

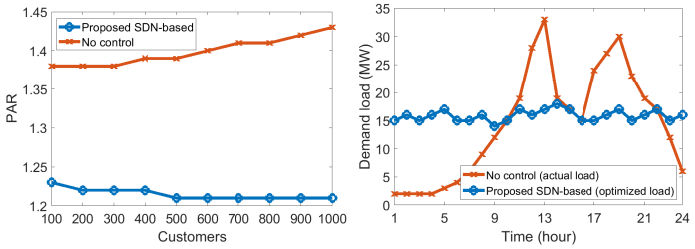
## V. Implementation and performance evaluation

Topology and test platform include eight PMs for DRASs, three PCs for Open vSwitch (OVS) switches [32], and one PM for the proposed controller (Fig. 11). PMs include three HP G9s, two G8s, three G7s, and one G6 server. In the testbed, Floodlight vl.2 [33] and OpenStack [34] are the controllers of the switches and PMs resources. Moreover, we utilize the Open vSwitch v2.4.1 for OpenFlow switches implementation. Floodlight is a powerful Java-based controller. Open vSwitch is a software and virtual switch supporting the OpenFlow. Floodlight has been run on an HP G9 DL580 server, and the modules designed in the previous section have been implemented on it. This server is equipped with VMware ESXi hypervisor v6.0. Also, the OVSs have been implemented on three PCs armed with an Intel Xeon 4-core 3.70GHz CPU and 16 GB RAM. The Oprofile [35] software is employed to monitor the PMs' consumption resources. We employ iPref [36] and StarTrinity [37] to generate DR traffic and measure network parameters, respectively. The bandwidth of the links is 10 Mbps. Each experiment was implemented at least three times and the average of which is considered as the outcome.

### A. DR Energy Balance (softwarization DRAS)

In this section, we are to assess the DR energy (DR load) balance employing the proposed framework. PAR (peak to average


Figure 11. The overview of the testbed of the proposed frameworks.



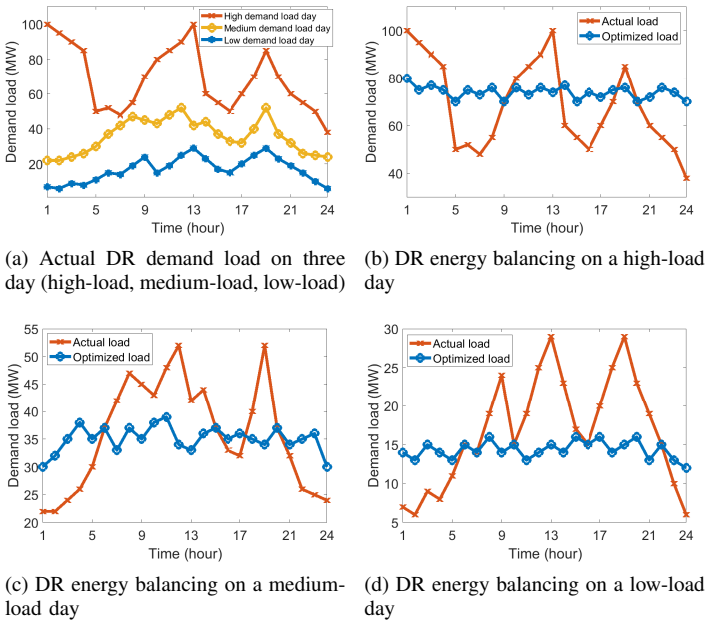

(a) PAR (peak to average ratio) (b) DR energy balancing

Figure 12. No control vs. proposed SDN-based

ratio)[1] index "proposed SDN-based framework" and "no control" for different number of customers are displayed in Fig. 12a. No control, high peak load, and low average load occur. Therefore, PAR is high. Moreover, by increasing the number of users, the peak load further increases and thus, PAR increases. By applying the SDN-based proposed framework, the peak is shaved and PAR decreases even for the high number of users. As a result, the proposed framework can shave the peak DR load and shift the loads to off-peak periods effectually.

Fig. 12b shows the reduction in the DR energy consumption. The result shows that the consumption is moved to off-peak time by the proposed controller (1:00 to 9:00 a.m.). The time period 10:00 to 13:00 when the real load demand is higher (on-peak time) displays a reduction in order to encounter the generation profile. From 14:00 to 16:00, demand is in mid-peak state and the load does not change. Also, in the time period 17:00 to 21:00, a reduction occurs and load is optimized. Finally, the load increases in off-peak time (22:00 to 24:00). Overall, the proposed SDN-based framework achieves a 6.81% reduction in the energy consumption. Therefore, the proposed SDN-based framework acts satisfactorily and demonstrates a significant load reduction in the DR testbed.

[1]The Peak to Average Ratio (PAR) is a momentous power network metric that is defined as the maximum daily load divided by the average load. The DR proposed framework is used to shift parts of power consumption from high peak hours to off-peak hours so as to cut down the PAR.




(a) Actual DR demand load on three day (high-load, medium-load, low-load)

(b) DR energy balancing on a high-load day

(c) DR energy balancing on a medium-load day

(d) DR energy balancing on a low-load day

Figure 13. DR energy balancing.

Note that $\tau$ is considered 15 minutes. If we consider $\tau$ less (e.g., 5 minutes), the optimized load curve will be flatter, meaning that the distance between the peak and the valley will decrease. However, it lay a foundation for network traffic (network becomes more crowded). Diminishing the $\tau$ value is more effective for great and industrial customers than home customers.

In the next experiment, the real load demand prepared from the Quchan city power grid is used. For probing, three days depicting 15 November 2019 (low-load day), 12 May 2019 (medium-load day), and 20 August 2019 (high-load day) load demands are considered. The demand curves for three days are shown in Fig. 13a. Using the SDN-based proposed framework, these demand curves are optimized to reduce the consumers' load demand. Using the proposed method, the load demand of 20 August 2019 (high) is optimized (Fig. 13b). It displays that the load demand is reduced from 1:00 to 4:00. Then, the load demand remains is increased (5:00 to 8:00). From 9:00 to 13:00, the load demand is again reduced. The load curve from 14:00 to 24:00 (except at 19:00) is increased again. Similarly, the load demand of 12 May 2019 and 15 November 2019 (medium and low) are optimized (Figs. 13c and 13d).

The result indicates a reduction in the energy load of 3.78% (low-load demand case), 4.69% (medium-load demand case), and 11.26% (high-load demand case). Therefore, it demonstrates the effectiveness of the proposed SDN-based framework.

The mapping of the generation profile with the optimized load demand is depicted in Fig. 14. The results indicate that the optimized load in all three days is almost equal to the generation profile. Consequently, the proposed SDN-based framework is efficient in balancing the DR energy.

### *B. Traffic Balance among DRAS Servers (Physical DRAS)*

In this section, the DR traffic balance among DRAS servers will be evaluated. In that regard, evaluation criteria include throughput, delay, and the average usage of the CPU and memory of the servers. This experiment consists of two scenarios with different DR background traffic. In the first scenario, the background

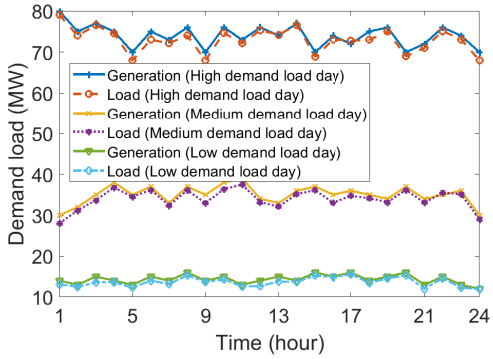



Figure 14. Mapping of the generation and optimized DR energy.

traffics of each DRAS are identical and equal to 50 supply and demand messages per second (mps). On the other hand, in the second scenario, the background traffics of DRASs are different: The first DRAS; 50 mps, the second DRAS; 100 mps, and so on. Then, for 100 seconds, a constant offered load (300 supply and demand mps) is introduced to the network. In this experiment, the three following methods are employed to balance DR traffic:

1. Proposed framework (based on response time), 2. Random, 3. Round-robin.

Fig. 15 demonstrates the results obtained in this experiment. In both of the scenarios, the proposed framework have obtained better results rather than Round-robin and Random. Moreover, even though the proposed framework's results are independent of the scenario, the Round-robin and Random techniques' outcomes in the second scenario are worse than the first scenario. Since the traffic of DRASs background is different in the second scenario. The proposed framework's throughput is pretty close to the offered load (Figs. 15a and 15b). Due to the fact that it has a correct estimation of the DRASs load.

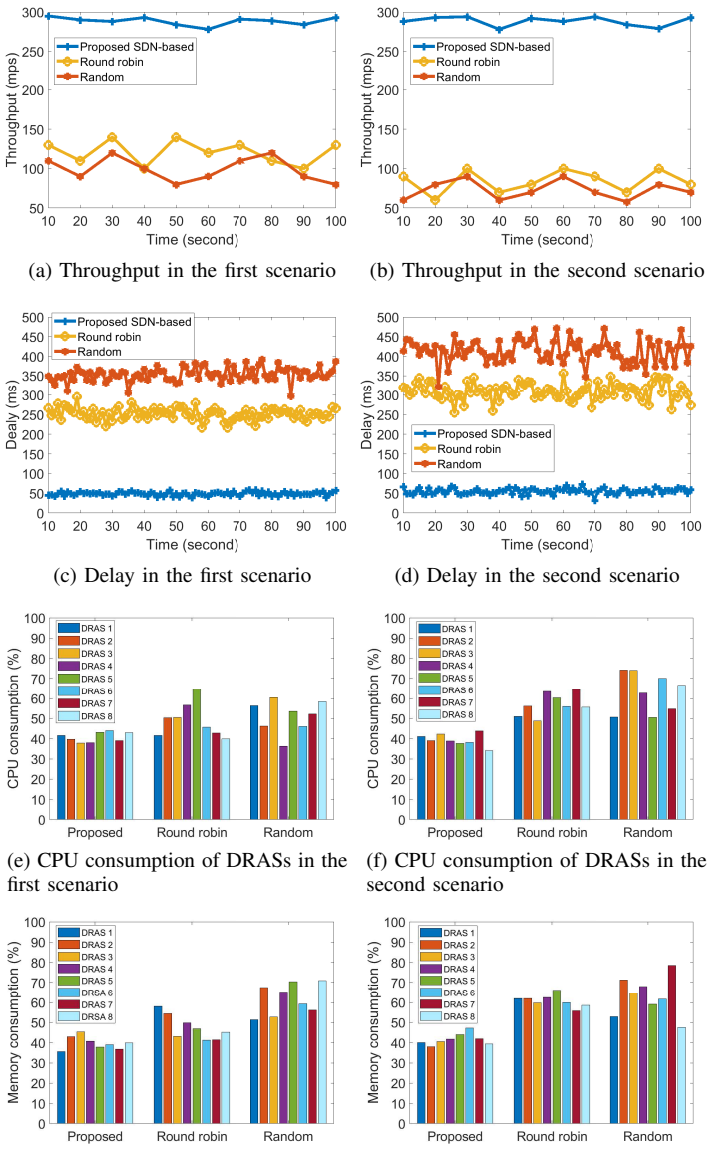
Despite more usage of Round-robin and Random resources than the proposed method, their average throughput is lower, and their delay is higher (Figs. 15c and 15d). In the proposed framework, the DRAS resource usage is approximately equal, indicating the network traffic's deliberate distribution by the mentioned SDN-based framework (Figs. 15e to 15h).

### C. The virtual DR based on software-defined NFV (Virtual DRAS)

In the previous experiment, the physical DRAS resources were constant. Accordingly, they are likely either to be saturated in peak times or useless at low-load times. Virtual DRAS solves this issue, and DRAS resources can either be enhanced in the course of peak times or released in idle times. The OpenStack platform is utilized to provide virtual DRASs in the testing platform and its management by NFVO. OpenStack is comprised of the components, such as Nova, responsible for the establishment and management of VMs. In this investigation, each of the virtual DRASs can have one of the four flavors indicated in Table I. The initial flavor of the DRASs is small. Running the command below, the virtual DRASs size can be changed:

$$Novaresize < VMInstanceName > < NewFlavor >$$

Table II demonstrates the throughput, admission rate, and the flavor of each DRAS through time. As can be observed, the virtual DRASs size has been changed at a duration of time following the enhancement or reduction in the load (scale-up or scale-down). In the optimal employment of the resources, this issue is of significant importance.




(a) Throughput in the first scenario
(b) Throughput in the second scenario
(c) Delay in the first scenario
(d) Delay in the second scenario
(e) CPU consumption of DRASs in the first scenario
(f) CPU consumption of DRASs in the second scenario
(g) Memory consumption of DRASs in the first scenario
(h) Memory consumption of DRASs in the second scenario

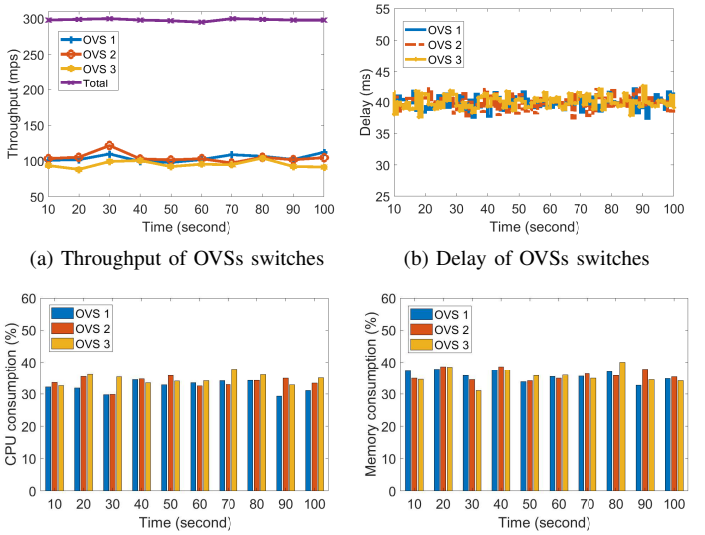
Figure 15. Traffic balance among DRAS servers (Physical DRAS).

### D. Traffic balance among the switches of the DR network

This experiment investigates the throughput, the delay, the average CPU, and memory usage of the three OVS switches. The results are provided in Fig. 16. The traffic of 300 mps flows in 100 seconds. As demonstrated, each of the three switches has an equal throughput and delay (Figs. 16a and 16b). Moreover, all three switches' usage resources are approximately equal, demonstrating that the network traffic between each of the three ones is balanced by the proposed SDN-based controller (Figs. 16c and 16d).

Table I
VM FLAVOR SPECIFICATIONS

| Flavor | Memory (MB) | vCPUs | Disk (GB) |
|---|---|---|---|
| small | 2048 | 1 | 20 |
| medium | 4096 | 2 | 40 |
| large | 8192 | 4 | 80 |
| xlarge | 16384 | 8 | 169 |




(a) Throughput of OVSs switches (b) Delay of OVSs switches

(c) CPU consumption of OVSs switches (d) Memory consumption of OVSs switches

Figure 16. Traffic balance among the switches of the DR network.

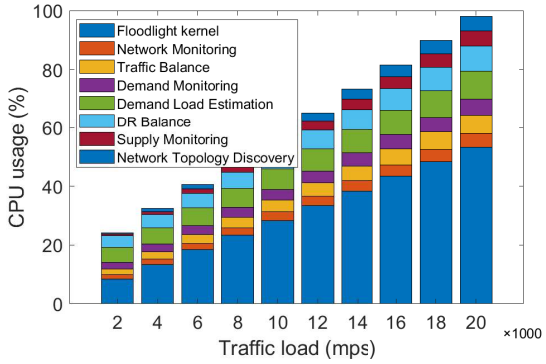



Figure 17. CPU consumption by the designed modules of the controller.

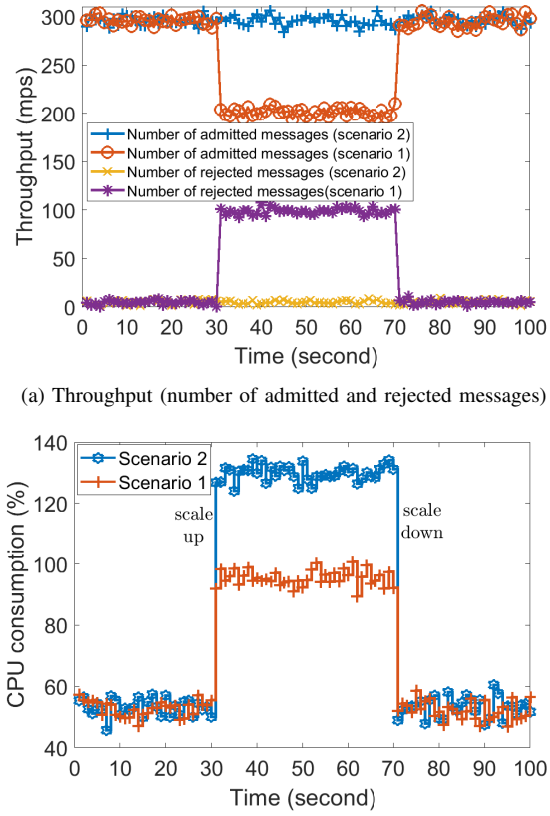
### *E. Scalability of the controller*

In this section, the evaluation of the proposed controller function is carried out. Fig. 17 indicates the details of the controller's CPU usage employing the obtained Oprofile software. As can be seen, the Floodlight Kernel occupies most of the CPU. However, the designed modules have a lesser CPU usage. This matter confirms the scalability of the controller. Among the designed modules, the *Demand Load Estimation* has more CPU usage due to the NLMS algorithm's implementation. Moreover, this figure indicates that the controller CPU is not saturated until the input DR traffic of 20000 mps.

Table III also indicates that controller throughput is so close to the input traffic or offered load. It has handled almost all messages of the OpenFlow. Besides, the delay time is less than 1000 ms. Also, the maximum occupied memory is 90.76%. Hence, the recommended modules do not put a strain on the controller.

### *F. Effect of DRASs failure on the service quality*

In this experiment, the examination of some of the DRASs' failure is examined, finding out whether the focused controller is capable of managing DR under these circumstances. The outcomes of this experiment are given in Fig. 18. The first DRAS comes across a failure at the 30th second, then starting to provide service in the 70th second. Two scenarios are investigated:




(a) Throughput (number of admitted and rejected messages)

(b) CPU consumption (resources scaling in Scenario 2)

Figure 18. Effect of DRASs failure on the throughput and CPU consumption in two scenario (Scenario 1: none of the DRASs are virtual, Scenario 2: all of the DRASs are virtual).

- The first scenario: none of the DRASs are virtual,
- The second scenario: all of the DRASs are virtual.

As shown in Fig. 18a, before the 30th second, the average throughput in both scenarios is equal to 296 mps. At the 30th second, the first DRAS faces a failure. In the first scenario, the resource consumption increases (Fig. 18b), while the average throughput diminishes to 192 mps, and the rejection of messages increases to 96 mps. In the second scenario, the resources are scaled up, since the DRASs are virtual. In this case, the throughput does not decline and remain approximately equal to 297 mps. When the first DRAS is reactivated at the 70th second, the traffic is distributed among all DRASs, and throughput is maximized.

### *G. Assessment of ILP solution*

In this part, we measure the optimal solution (ILP model) for a limited size. Concerning that the NP-hard is the subject of concern, the optimal solution cannot be achieved by the time complication of the polynomial. Accordingly, raising the size of the problem leads to the enhancement in the ILP time exponentially. We employed the CPLEX optimizer [38]. The results are indicated in Fig. 19. An SDN-based framework can obtain a result close to the outcome of an ILP (Fig. 19a). Nevertheless, with the enhancement in the size of the ILP method, the time rises significantly. In contrast, the proposed framework can accomplish a pseudo-optimal solution in a rational duration of time (Fig. 19b).

### *H. Effect of the time duration $\tau$*

In this part, we investigate the influence of $\tau$ on the proposed framework's performance. The results are indicated in Fig. 20.

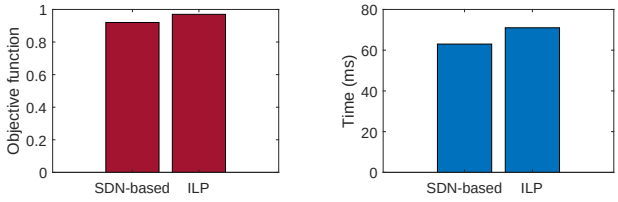
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSG.2021.3139004, IEEE Transactions on Smart Grid


Table II
THE EFFICIENCY OF THE PROPOSED FRAMEWORK BASED ON THE SOFTWARE-DEFINED NFV

| Time (s) | 0-100 | 100-200 | 200-300 | 300-400 | 400-500 | 500-600 |
|---|---|---|---|---|---|---|
| Offered load (mps) ∼ | 150 | 300 | 450 | 150 | 450 | 300 |
| Total throughput of DRASs (mps) ∼ | 146 | 296 | 446 | 149 | 445 | 297 |
| Admission rate (%) ∼ | 97.3 | 98.66 | 99.11 | 99.33 | 98.88 | 99 |
| DRAS 1 flavor | small | **medium** | **high** | small | **high** | **medium** |
| DRAS 2 flavor | small | **medium** | **medium** | small | **high** | **medium** |
| DRAS 3 flavor | small | **medium** | **high** | small | **medium** | **medium** |
| DRAS 4 flavor | small | **small** | **high** | small | **high** | **medium** |
| DRAS 5 flavor | small | **medium** | **high** | small | **high** | **small** |
| DRAS 6 flavor | small | **medium** | **high** | small | **medium** | **medium** |
| DRAS 7 flavor | small | **medium** | **high** | small | **high** | **medium** |
| DRAS 8 flavor | small | **medium** | **high** | small | **high** | **small** |

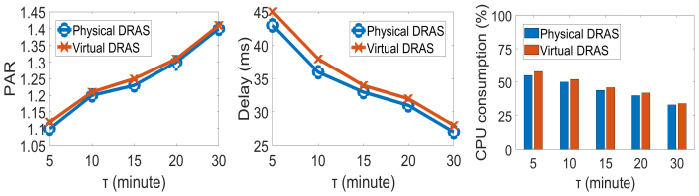
Table III
THE EFFICIENCY OF THE PROPOSED CONTROLLER

| Offered load (mps) | 2000 | 4000 | 6000 | 8000 | 10000 | 12000 | 14000 | 16000 | 18000 | 20000 |
|---|---|---|---|---|---|---|---|---|---|---|
| Throughput (mps) ∼ | 1991 | 3874 | 5911 | 7856 | 9865 | 11898 | 13874 | 15659 | 17762 | 19785 |
| Response time (ms) ∼ | 48 | 185 | 236 | 375 | 418 | 567 | 675 | 759 | 876 | 965 |
| Memory usage (%) ∼ | 17.74 | 28.34 | 37.87 | 49.23 | 54.98 | 61.45 | 67.63 | 77.64 | 85.75 | 90.76 |




(a) Objective function (b) Time to answer (ms)

Figure 19. Comparison between objective function and time to answer in the SDN-based framework and ILP model. The ILP model gives a better answer but in more time (it becomes exponential if expanded). But the SDN-based method gives a very close to the optimal answer in polynomial time.




(a) PAR (peak to average ratio) (b) Network delay (c) Resource consumption of switches

Figure 20. Effect of the time duration $\tau$ in two modes (1. Physical DRAS, 2. Virtual DRAS). More $\tau$ more PAR and less delay. If the DRASs are physical then the results are a little better.

As obvious, the more $\tau$, the higher the PAR will be (Fig. 20a). It demonstrates that the distance between the peak and the trough rises. Rather, the network becomes more secluded since lower control messages and commands are exchanged among the controller and the data plane. Consequently, both the delay (Fig. 20b) and the switches' resource usage are declined (Fig. 20c).

## VI. CONCLUSIONS AND FUTURE WORKS

In this study, we have addressed the simultaneous energy and data balance problem in the DR network. This problem comes across three types of constraints: The limitations related to the energy balance, traffic balance, and service quality. We demonstrated that this issue is an ILP, and subsequently, NP-hard. The requirements of DR energy and data balancing fit well with cloud applications. Accordingly, we provided three cloud and SDN-based frameworks. We decompose the problem into subproblems as SDN applications in the application plane and benefit from SDN capability to provide a global view of the entire network and solve them.

We designed a predictive controller, equipped with the *DR Balance (Energy Balance)*, as well as the *Path Selection (Traffic Balance)* modules. In the first framework, we transmitted the hardware function of DRAS from the data to the control plane, softwarizing the DR balance. In the second one, employing the SDN, we were looking for the traffic balance between physical DRASs. In the third framework, DRASs were provided in a virtual form and under the NFV MANO control. In these designs, the finite state machine was employed for the electricity dynamic pricing and the resources' dynamic adjustment. This is the first paper executing widespread experiments practically under the different SDN-based DR frameworks, providing the obtained results. It was shown that utilizing two-tier cloud computing based on SDN can provide a high level of reliability and scalability. In addition, it improves the performance of both the power network and communication network. Implementation results confirmed that the proposed cloud-based DR framework reduces the PAR while utilizing the resources of the communication network more effectively. One of the subsequent works is to utilize fog computing in order to develop the suggested frameworks. We are going to put the dual decomposition or lagrangian relaxation into use to mitigate the complexity of the problem.

## REFERENCES

[1] Y. Kabalci, "A survey on smart metering and smart grid communication," *Renewable and Sustainable Energy Reviews*, vol. 57, pp. 302–318, 2016.
[2] S. Bera, S. Misra, and J. J. Rodrigues, "Cloud computing applications for smart grid: A survey," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 5, pp. 1477–1494, 2014.
[3] K. Kaur, S. Garg, G. Kaddoum, S. H. Ahmed, F. Gagnon, and M. Atiquzzaman, "Demand-response management using a fleet of electric vehicles: an opportunistic-sdn-based edge-cloud framework for smart grids," *IEEE Network*, vol. 33, no. 5, pp. 46–53, 2019.
[4] A. Sheikhi, M. Rayati, S. Bahrami, A. M. Ranjbar, and S. Sattari, "A cloud computing framework on demand side management game in smart energy hubs," *International Journal of Electrical Power & Energy Systems*, vol. 64, pp. 1007–1016, 2015.

[5] H. Kim, Y.-J. Kim, K. Yang, and M. Thottan, "Cloud-based demand response for smart grid: Architecture and distributed algorithms," in *2011 IEEE international conference on smart grid communications (SmartGridComm)*. IEEE, 2011, pp. 398–403.

[6] J. S. Vardakas, N. Zorba, and C. V. Verikoukis, "A survey on demand response programs in smart grids: Pricing methods and optimization algorithms," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 152–178, 2014.

[7] H. Hui, Y. Ding, Q. Shi, F. Li, Y. Song, and J. Yan, "5g network-based internet of things for demand response in smart grid: A survey on application potential," *Applied Energy*, vol. 257, p. 113972, 2020.

[8] A. Elkasrawy and B. Venkatesh, "Demand response cooperative and demand charge," *IEEE Transactions on Smart Grid*, 2020.

[9] J. Zhou, R. Q. Hu, and Y. Qian, "Scalable distributed communication architectures to support advanced metering infrastructure in smart grid," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 9, pp. 1632–1642, 2012.

[10] A. Montazerolghaem, M. H. Y. Moghaddam, and A. Leon-Garcia, "Openami: Software-defined ami load balancing," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 206–218, 2017.

[11] B. Varghese and R. Buyya, "Next generation cloud computing: New trends and research directions," *Future Generation Computer Systems*, vol. 79, pp. 849–861, 2018.

[12] M. H. Rehmani, A. Davy, B. Jennings, and C. Assi, "Software defined networks-based smart grid communication: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2637–2670, 2019.

[13] H. Hawilo, M. Jammal, and A. Shami, "Network function virtualization-aware orchestrator for service function chaining placement in the cloud," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 3, pp. 643–655, 2019.

[14] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A survey on software-defined networking," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 27–51, 2015.

[15] M. Karimzadeh-Farshbafan, V. Shah-Mansouri, and D. Niyato, "A dynamic reliability-aware service placement for network function virtualization (nfv)," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 2, 2019.

[16] Y. Simmhan, S. Aman, A. Kumbhare, R. Liu, S. Stevens, Q. Zhou, and V. Prasanna, "Cloud-based software platform for big data analytics in smart grids," *Computing in Science & Engineering*, vol. 15, no. 4, pp. 38–47, 2013.

[17] L. Zheng, N. Lu, and L. Cai, "Reliable wireless communication networks for demand response control," *IEEE Transactions on Smart Grid*, vol. 4, no. 1, pp. 133–140, 2013.

[18] P.-Y. Kong, "Wireless neighborhood area networks with qos support for demand response in smart grid," *IEEE transactions on smart grid*, vol. 7, no. 4, pp. 1913–1923, 2015.

[19] L. Zheng, S. Parkinson, D. Wang, L. Cai, and C. Crawford, "Energy efficient communication networks design for demand response in smart grid," in *2011 International Conference on Wireless Communications and Signal Processing (WCSP)*. IEEE, 2011, pp. 1–6.

[20] M. H. Yaghmaee, A. Leon-Garcia, and M. Moghaddassian, "On the performance of distributed and cloud-based demand response in smart grid," *IEEE Transactions on Smart Grid*, vol. 9, no. 5, pp. 5403–5417, 2017.

[21] M. Pruckner, A. Awad, and R. German, "A study on the impact of packet loss and latency on real-time demand response in smart grid," in *2012 IEEE Globecom Workshops*. IEEE, 2012, pp. 1486–1490.

[22] Y. Yan, Y. Qian, H. Sharif, and D. Tipper, "A survey on smart grid communication infrastructures: Motivations, requirements and challenges," *IEEE communications surveys & tutorials*, vol. 15, no. 1, pp. 5–20, 2012.

[23] T. Khalifa, K. Naik, and A. Nayak, "A survey of communication protocols for automatic meter reading applications," *IEEE communications surveys & tutorials*, vol. 13, no. 2, pp. 168–182, 2010.

[24] E. Ebeid, S. Rotger-Griful, S. A. Mikkelsen, and R. H. Jacobsen, "A methodology to evaluate demand response communication protocols for the smart grid," in *2015 IEEE International Conference on Communication Workshop (ICCW)*. IEEE, 2015, pp. 2012–2017.

[25] S.-C. Tsai, Y.-H. Tseng, and T.-H. Chang, "Communication-efficient distributed demand response: A randomized admm approach," *IEEE Transactions on Smart Grid*, vol. 8, no. 3, pp. 1085–1095, 2015.

[26] C.-H. Lo and N. Ansari, "The progressive smart grid system from both power and communications aspects," *IEEE Communications Surveys & Tutorials*, vol. 14, no. 3, pp. 799–821, 2011.

[27] A. Jain, A. Mani, and A. S. Siddiqui, "Network architecture for demand response implementation in smart grid," *International Journal of System Assurance Engineering and Management*, vol. 10, no. 6, 2019.

[28] S. Rinaldi, F. Bonafini, P. Ferrari, A. Flammini, E. Sisinni, D. Di Cara, N. Panzavecchia, G. Tinè, A. Cataliotti, V. Cosentino *et al.*, "Characterization of ip-based communication for smart grid using software-defined networking," *IEEE Transactions on Instrumentation and Measurement*, vol. 67, no. 10, pp. 2410–2419, 2018.

[29] N. Saputro, K. Akkaya, and S. Uludag, "A survey of routing protocols for smart grid communications," *Computer Networks*, vol. 56, no. 11, 2012.

[30] F. Eisenbrand, C. Hunkenschröder, K.-M. Klein, M. Koutecký, A. Levin, and S. Onn, "An algorithmic theory of integer programming," *arXiv preprint arXiv:1904.01361*, 2019.

[31] C. A. Tovey, "A simplified np-complete satisfiability problem," *Discrete applied mathematics*, vol. 8, no. 1, pp. 85–89, 1984.

[32] "Open vswitch," www.openvswitch.org, accessed: 2021-01-30.

[33] "Floodlight controller," www.projectfloodlight.org, accessed: 2021-01-30.

[34] "Openstack," https://www.openstack.org/, accessed: 2021-01-30.

[35] "Oprofile," https://oprofile.sourceforge.io/, accessed: 2021-01-30.

[36] "Network monitoring tool," www.iperf.fr, accessed: 2021-01-30.

[37] "Startrinity," http://startrinity.com/, accessed: 2021-01-30.

[38] "Cplex optimizer," https://www.ibm.com/analytics/cplex-optimizer, accessed: 2021-01-30.


**Ahmadreza Montazerolghaem** received the Ph.D. degree in computer engineering from the computer department, Ferdowsi University of Mashhad (FUM), Iran, in 2017. He was a postdoctoral researcher at FUM in 2018. Also, he was IT Director at the Quchan University of Technology, Quchan, Iran in 2020. He is currently an Assistant Professor at the Faculty of Computer Engineering, University of Isfahan, Isfahan, Iran. He is an IEEE student member and quality manager at VoIP type approval laboratory in FUM. He is also a member of National Elites Foundation (Society of prominent students of the country). His research interests are in Software Defined Networking (SDN), Network Function Virtualization (NFV), Internet of Things (IoT) and, Multimedia Networking.

**Mohammad Hossein Yaghmaee** received his B.S. degree in communication engineering from Sharif University of Technology, Tehran, Iran in 1993, and M.S. degree in communication engineering from Tehran Polytechnic (Amirkabir) University of Technology in 1995. He received his Ph.D degree in communication engineering from Tehran Polytechnic (Amirkabir) University of Technology in 2000. He has been a computer network engineer with several networking projects in Iran Telecommunication Research Center (ITRC) since 1992. November 1998 to July1999, he was with Network Technology Group (NTG), C&C Media research labs., NEC corporation, Tokyo, Japan, as visiting research scholar. September 2007 to August 2008, he was with the Lane Department of Computer Science and Electrical Engineering, West Virginia University, Morgantown, USA as the visiting associate professor. July 2015 to September 2016, he was with the electrical and computer engineering department of the University of Toronto (UoT) as the visiting professor. Currently, he is a full professor at the Computer Engineering Department, Ferdowsi University of Mashhad (FUM). His research interests are in Smart Grid, Internet of Things (IoTs), Computer and Communication Networks, Quality of Services (QoS), Software Defined Networking (SDN) and Network Function Virtualization (NFV). He is an IEEE Senior member and head of the IP-PBX type approval lab in the Ferdowsi University of Mashhad. He is the author of some books on Smart Grid, TCP/IP and Smart City in Persian language.