Network-Aware QoS Routing for Smart Grids Using Software Defined Networks

Jinjing Zhao[1(✉)], Eman Hammad[2], Abdallah Farraj[2], and Deepa Kundur[2]

[1] National Key Laboratory of Science and Technology on Information System Security, Beijing, China
jinjing.zhao@utoronto.ca
[2] Department of Electrical and Computer Engineering, University of Toronto, Toronto, Canada
{ehammad,abdallah,dkundur}@ece.utoronto.ca

Abstract. We consider the problem of quality of service (QoS) routing for smart grids using software defined networks (SDN). The SDN framework enables an efficient decoupled implementation of dynamic routing protocols that is aware of the communication network status. In this work we consider the varying delay status of the communication network along with other network parameters such as links throughput. The routing problem is formulated as a constrained shortest path problem. The results for a test case of the New England test power system are shown.

Keywords: Constrained shortest path · Quality of service · Routing · Software defined networks · Smart grid

1 Introduction

One of the coupled interactions in the smart grid is between communication network infrastructure and cyber-enabled control; in this context many works have considered the quality of service (QoS) of the communication infrastructure that is involved in the system control [1–3]. Previous works studied delay-sensitive control functions that require QoS guarantees or best effort and consequently formulated the routing problem in smart grid systems as a QoS routing problem. In [2] an enhanced genetic algorithm with ticket-based flooding discovery is proposed for a QoS routing in the smart grid. QoS routing problem is also considered in [1] in the context of demand-response; where the authors develop a QoS metric based on the impact of some constraints on the pricing-based control, and propose a greedy algorithm to solve for the shortest path under the defined QoS metric. In addition to the QoS studies, recent research work have tried to address denial of service (DoS) attacks on communication networks using different approaches including potential games, flocking based routing, and genetic algorithms in order to avoid the links that are under attack [4].

A. Leon-Garcia et al. (Eds.): Smart City 2015, LNICST 166, pp. 384–394, 2016.
DOI: 10.1007/978-3-319-33681-7_32

Dijkstra algorithm can be used for finding the shortest paths between nodes in a graph; it is widely used in network routing protocols, most notably in Intermediate System to Intermediate System (IS-IS) [5] and Open Shortest Path First (OSPF) [6]. In Dijkstra algorithm, each network link has a cost value to present its status, and this cost is used to calculate the shortest path. In practical implementations of the Dijkstra algorithm, link costs are assigned in a simple approach due to the hardship of obtaining useful link status updates automatically and dynamically for the whole network. For example, a link cost in OSPF is defined as the reference bandwidth divided by interface bandwidth which leads to a static cost value [6]; however, in other cases, link cost is defined simply as 1 reducing the shortest path weight to a hop count.

The software defined network (SDN) framework provides an approach to solve for the shortest path based on dynamic link statuses through SDN's high network monitoring capability. Many useful link information (for example, link type, ownership, bandwidth, delay and historical data) can be collected by the SDN controller and used in the routing algorithm enabling more reliable, safe and efficient paths.

Enabled by the SDN framework, the implementation of the double constrained QoS routing in the SDN framework takes advantage of the dynamic communication network state. A dynamic delay cost matrix is obtained using regularly-updated link delay statistics. The constrained shortest path (CSP) algorithm is evaluated by the SDN controller and corresponding routing entries are respectively pushed to related forwarding switches.

The main contributions of this work include the following:

1. We formulate a multi-constraint routing problem for network-status aware routing.
2. We model and simulate the smart grid communication network using software defined networks.

The remainder of this paper is organized as follows: the problem setting is presented in Sect. 2, double constrained shortest path problem is discussed and the derivation of QoS constraints and related cost metrics are presented in Sect. 3, a background on software defined networks is provided in Sect. 4, and implementation details are provided in Sect. 5. Section 6 investigates the performance of the proposed framework and some test cases are considered. Conclusions and final remarks are discussed in Sect. 7.

2 System Model

Let N denote the number of nodes in the power system; for this discussion let N refer to number of buses in the power grid. Then, without loss of generality, we can assume a communication network connecting the N buses in a topology that parallels that of the electrical grid. This assumption is justified based on a mix of fiber optic and Ethernet physical-media communication networks.

Consider a graph representation of the corresponding communication network. The weighted undirected graph model $G(V, E, w)$ describes an N-node and M-link network, where the node set $V = \{v_1, \ldots, v_N\}$ and the edge set $E = \{e_{ij}, i, j = 1, 2, \ldots, M\}$ denote the buses and communication links, respectively. The weight w on the edge between two nodes is defined as the cost of the corresponding communication link. Then, the adjacency matrix A can be defined as

$$A_{i,j} = \begin{cases} w_{ij} \; i \neq j, \text{ for } (i,j) \in E \\ 0 \quad \text{otherwise.} \end{cases} \tag{1}$$

Consider next the routing problem of communication data between a source node s and destination node t in the graph G. The shortest path route between the pair can be found using various algorithms. Due to its simplicity and optimality, Dijkstra-based routing algorithm has long been the most used algorithm to arrive at the shortest path.

Within the smart grid, cyber-enabled control systems require information delivery between relevant nodes with certain delay requirements; as an example, the IEC 61850 GOOSE messaging specifies the message delay constrains for performance class P2/3 to be within 3 ms [7]. Accordingly, if we define the delay cost matrix A_d, then the problem of finding paths that satisfy the delay constraints can be formulated as a CSP problem. When there are more than one constraint, the CSP problem is often called a multi-constraint shortest path (MCSP) problem. Further, CSP and MCSP problems are proved to be NP-hard; yet, many algorithms have been developed to find a feasible or a set of feasible solutions [8].

It is essential to distinguish between the nature of the constraints and their interdependencies, as this will dictate if the constraints can be combined (in an additive way) into a single constraint. To illustrate this idea, consider packet-drop and link congestion, where an interdependence between the two constrains can be observed; a similar relation could be observed between congestion and communication delay. This approach can be abstracted by considering only one constraint or by constructing an additive combination of the two constraints as one with proper scaling.

In the context of smart grid systems, networked sensory and control impose many constraints on data communications; nevertheless, we are interested with how SDN can facilitate an efficient dynamic delay aware protocol with other constraints. Additionally, the SDN framework allows us to obtain a dynamic delay cost matrix sampled from the network at pre-defined intervals. If T_{sd} is defined as the sampling time, then the dynamic delay cost matrix is defined as $A_d^i = A_d(T_{sd}.i)$.

3 Constrained Shortest Path Problem

Given a network $G(V, E)$, assume every link $L_{u,v} \in E$ has two weights $c_{uv} > 0$ and $d_{uv} > 0$ (denoting, cost and delay). For source and destination nodes (s, t)

and maximum delay $T_{max} > 0$, let $\mathbf{P}_{st}$ denote the set of paths from s to t. Further, for any path p define

$$\begin{aligned} c(p) &= \sum_{(u,v)\in p} c_{uv} \\ d(p) &= \sum_{(u,v)\in p} d_{uv}. \end{aligned} \tag{2}$$

CSP problem seeks to arrive at the shortest path between s and t nodes with a certain link cost c. However, when the path is constrained by more than one constraint, the problem is termed an MCP problem. Given that there are multiple paths between s and t, a modified MCP problem, often called the multi-constrained optimal path (MCOP) problem, is defined where the goal is to retrieve the shortest path among a set of feasible paths. Furthermore, restricted shortest path (RSP) problem is a special case of MCOP problems where the goal is to find the path with the least cost among the set of feasible paths that satisfy one constraints; for example, a constraint on T bounds the maximum path delay [9].

A feasible path $s \to t$ is defined as path p_{st} that satisfies $d(p_{st}) \leq T_{max}$. let $P_{st}(T_{max})$ be the set of all feasible paths from s to t. Then, the CSP problem can be formulated as an integer linear program (ILP) with a set of zero-one decision variables x_{uv}. For each link $(u, v) \in E$; define $x_{uv} = 1$ if the link is in path p, and $x_{uv} = 0$ otherwise. The problem of finding the minimum-cost feasible path can be formulated as an integer linear program as [8–10]

$$\begin{aligned} \text{minimize} \quad & \sum_{(u,v)\in E} c_{uv}\, x_{uv} \\ \text{subject to} \quad & \sum_{v\in V} x_{uv} = \sum_{v\in V} x_{vu}, \forall u \in V \setminus \{s,t\} \\ & \sum_{v\in V} x_{sv} = 1 \\ & \sum_{u\in V} x_{ut} = 1 \\ & x_{uv} \in \{0,1\}, \forall (u,v) \in E. \end{aligned} \tag{3}$$

If the integrity condition $x_{uv} \in \{0, 1\}$ is relaxed into $x_{uv} \geq 0$ then the dual of the relaxed problem (a Lagrangian dual problem) is constructed [8,10]. The dual will include $s \to t$ paths and a multiplier $\lambda \geq 0$. For a link (u, v), let the aggregated cost c_λ be defined as $c_{uv} + \lambda d_{uv}$. Additionally, for a given λ, aggregated cost of the p is annotated $c_\lambda(p)$. Then, $L(\lambda)$ is defined as

$$L(\lambda) = \min\{c_\lambda(p) | p \in \mathbf{P}_{st}\}. \tag{4}$$

Let p_λ denote the path from s to t with the minimum aggregated cost with respect to a given λ. Then, $L(\lambda) = c_\lambda(p_\lambda) - \lambda T_{max}$, and the dual of the relaxed problem can be described as

$$L^* = \max\{L(\lambda) | \lambda \geq 0\}. \tag{5}$$

Algorithm 1. LARAC Algorithm

PROCEDURE LARAC(s, t, d, T)
$p_c := \text{Dijkstra}(s, t, c)$
if $d(p_c) \leq T$ **then**
  return p_c
$p_d := \text{Dijkstra}(s, t, d)$
if $d(p_d) > T$ **then**
  return "there is no solution"
loop
  $\lambda := \frac{c(p_c)-c(p_d)}{d(p_d)-d(p_c)}$
  $r := \text{Dijkstra}(s, t, c_\lambda)$
  if $c_\lambda(r) = c_\lambda(p_c)$ **then**
    return p_d
  else if $d(r) \leq T$ **then**
    $p_d := r$
  else
    $p_c := r$
END PROCEDURE

As previously pointed out, this is an NP-hard problem, where usually algorithmic approaches have successfully arrived at feasible solutions. The Lagrangian Relaxation Based Aggregated Cost (LARAC) algorithm developed in [10] solves the integer relaxation of the CSP problem. The LARAC algorithm is proven in [8] to be equivalent to Minimum Cost Restricted Time Combinatorial Optimization (MCRT) problems; further, the authors establish the generality of the LARAC algorithm for solving combinatorial problems involving two metrics. As shown in Algorithm 1, the LARAC algorithm presents an efficient procedure to arrive at an optimal λ and to terminate the search.

4 Software Defined Networks

Software defined networking offers the potential to change the traditional way networks operate. Current communication networks are typically built from a large number of network devices, with many complex protocols implemented on them. Operators in traditional communication networks are responsible for configuring policies to respond to a wide range of network events and applications. Consequently, network management and performance tuning is quite challenging and error-prone [11].

SDN provides a new toolkit and perspective for approaching many problems in smart grid communication network. Recent works have started to explore the potential of SDN in smart grids. In [12], Sydney *et al.* present a prototype that integrates a 4-bus power grid testbed with an OpenFlow network. Further, Goodney *et al.* propose in [13] an efficient multicast SDN system that connects high-rate PMUs and data subscribers with different data rate requirements. Moreover, an integrated SDN with IEC-61850-based substation automation systems is proposed in [14,15] to facilitate and improve the networking of intelligent

electric devices (IEDs) in a substation. In addition, Kim *et al.* propose in [16] using OpenFlow switches to form virtual local area networks (VLANs) for multiple grid applications with different QoS requirements. Zhang *et al.* also discuss in [17] three use cases of SDN applications in smart grids; specifically, content-based data exchange, virtual networks for distributed energy resource (DER) aggregation, and smart building management. Furthermore, Molina *et al.* discuss in [14] an OpenFlow's fast failover mechanism upon the detection of node failures in the application of SDN to IEC-61850-based substations. Finally, Dong *et al.* explain in [18] some of the challenges in applying SDN to improve smart grid resilience.

4.1 SDN Architecture

SDN is an approach to networking that allows network administrators to manage network services through abstraction of lower-level functionality. This is done by decoupling the system that makes decisions about where traffic is sent (the control plane) from the underlying systems that forward traffic to the selected destination (the data plane). An SDN network comprises of two main components:

- SDN Controller: the controller is a logically centralized function that determines the forwarding path for each flow in the network. A network is typically controlled by one or a few controllers.
- SDN Switch: SDN switches constitute the network data plane. The logic for forwarding the packets is determined by the controller and is implemented in the forwarding table at the switches.

The SDN architecture is designed to provide dynamic, manageable, cost-effective and adaptable networks. Within the SDN framework, network applications can obtain detailed traffic statistics from network devices and thus construct an up-to-date global network view. One common standard for the implementation of software defined networks is OpenFlow [19]. The OpenFlow standard defines a communication protocol between network switches forming the data plane and one or multiple controllers forming the control plane.

4.2 Implementation

Our SDN system setup is built using free open source tools. We use Floodlight v1.0 [20] as the SDN controller and Mininet 2.2.0 [21] as the SDN switches. Floodlight is an Apache-licensed, Java-based OpenFlow SDN Controller. Mininet can create a realistic virtual networks. The SDN controller can communicate with the switches via the OpenFlow protocol through the abstraction layer present at the forwarding hardware.

The architecture of the test SDN network is illustrated in Fig. 1 and is comprised of Floodlight controller and Mininet switches, and an OpenFlow controller typically manages a number of switches. Every switch maintains one or more

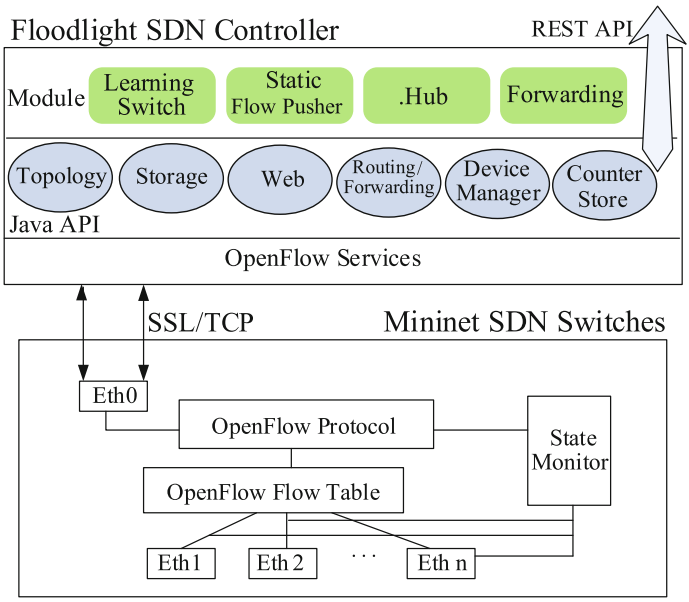

Fig. 1. Network architecture

flow tables that determine how packets belonging to a flow will be processed and forwarded. Communication between a controller and a switch happens via the OpenFlow protocol, which defines a set of messages that can be exchanged between these entities over a secure channel. The state monitor module can be used to collect switch state and transmit it to the controller.

5 Smart Grid

We consider the New England 10-generator 39-bus test power system; we develop a communication network that parallels that of the power grid. The New England power system and its candidate communication topology are shown in Fig. 2. The communication topology is based on the assumption of installing a network forwarding switch at every bus.

5.1 Network Topology

We consider a network where a centralized SDN controller computes the forwarding table for a set of SDN forwarding switches. Each bus can be considered as an SDN switch. In addition to forwarding packets, the SDN switches do some simple traffic measurement which they forward to the controller. The SDN controller uses this information along with routing rules defined by the protocol to dynamically change the forwarding tables at the switches in order to adapt to changing network link conditions.

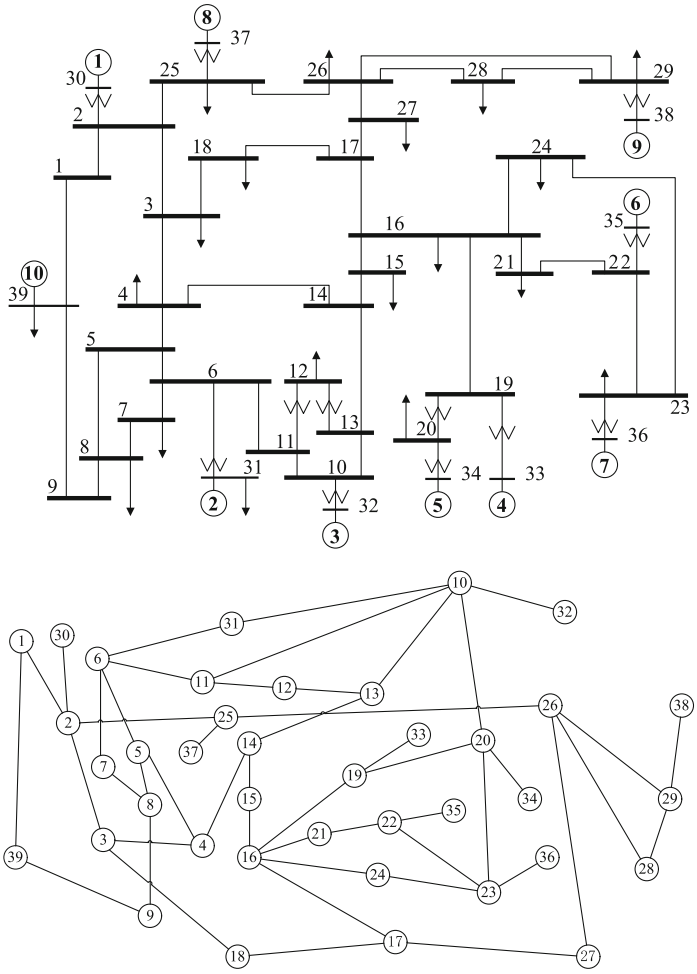

Fig. 2. New England test power system and its proposed communication network

5.2 Environment

The test environment is built using Floodlight v1.0 as the SDN controller [20], and Mininet 2.2.0 is used to implement the SDN switches network topology [21]. We utilize Iperf and jperf as test tools for collecting network performance statistics. Iperf [22] is a commonly-used network testing tool that can create Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) data streams and measure the throughput of a network that is carrying them. Iperf has a graphical user interface (GUI) frontend called jperf [23]. The simulation environment runs on a Windows 7 64-bit machine with a 2.53-GHz Intel Core i5 CPU and 8-GB RAM.

Table 1. Example of different paths between DCSP and Dijkstra algorithms

Algorithm	11 → 31	1 → 38	22 → 24
DCSP	{11, 6, 31}	{1, 2, 25, 26, 29, 38}	{22, 21, 16, 24}
Dijkstra	{11, 10, 31}	{1, 2, 25, 26, 29, 38}	{22, 23, 24}

Table 2. Processing time of random-chosen pairs (ms)

Algorithm	11 → 31	1 → 38	22 → 24
DCSP	20.149	20.932	20.233
Dijkstra	20.145	20.915	20.227

6 Simulation Results

We consider a Double Constrained Shortest Path (DCSP) algorithm, and we compare the network performance between DCSP and Dijkstra algorithms in a smart grid communication network with different link bandwidths. The DCSP algorithm tries to achieve the best throughput within the maximum delay restrictions. The Dijkstra algorithm in Floodlight calculates the shortest paths between source and destination based on one metric, and usually it is the hop count (number of links in the path). The end-to-end network bandwidth and delay are measured to compare the network performance using the two algorithms.

Link bandwidth is used to calculate the link cost, where the bandwidth of each link bw_{ij} is assigned randomly within a certain range say $[2 - 4.5]$ Gbps in this case. Further, the link cost c_{bw}^{ij} is defined as $c_{bw}^{ij} = 4.5 - bw_{ij}$; higher bandwidth links have lower link cost, and vice versa. Meanwhile, link delay cost d_{ij} is collected from the network dynamically, and a parametric maximum delay is defined $T_{max} = 1000$ ms.

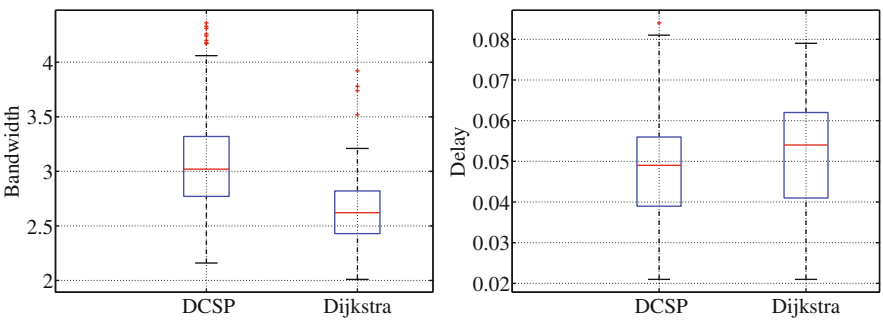

Fig. 3. Bandwidth and delay performance comparison of DCSP and Dijkstra algorithms

Random pairs of source and destination nodes (s, t) are selected (from a combination of 38×38), where for each pair both DCSP and Dijkstra algorithms are run. Statistics show that there are 228 different paths (119 undirected paths) between DCSP and Dijkstra among the total 1444 paths. Table 1 shows a random subset of 3 different (s, t) pairs with the calculated paths using DCSP and Dijkstra algorithms. In addition, Table 2 shows processing time for calculating the routes.

Finally, the set of affected paths was run for duration of 30 s during which bandwidth and delay was calculated using iperf and ping every 10 s. Numerical results are shown in Fig. 3 where DCSP algorithm has better performance both in network bandwidth and delay compared to that of Dijkstra algorithm.

7 Conclusions

In this work, we investigate the QoS routing problem with added constraints for smart grids using software defined networks. We consider the delay of the communication network of the smart grid along with other network parameters such as links throughput. The routing problem is formulated as a constrained shortest path problem. Numerical results for a test case of the New England test power system are shown.

References

1. Li, H., Zhang, W.: QoS routing in smart grid. In: IEEE Global Telecommunications Conference (GLOBECOM), pp. 1–6 (2010)
2. Zaballos, A., Vernet, D., Selga, J.M.: A genetic QoS-aware routing protocol for the smart electricity networks. Int. J. Distrib. Sens. Netw. **2013**, 1–12 (2013)
3. Pavlidou, F.-N., Koltsidas, G.: Game theory for routing modeling in communication networks a survey. J. Commun. Netw. **10**(3), 268–286 (2008)
4. Zhu, Q., Wei, D., Basar, T.: Secure routing in smart grids. In: Workshop on Foundations of Dependable and Secure Cyber-Physical Systems (FDSCPS), pp. 55–59 (2011)
5. Network Working Group, RFC 1195 use of OSI IS-IS for routing in tcp/ip and dual environments. Available at. https://www.ietf.org/rfc/rfc1195.txt (1990). Accessed 9 June 2015
6. Network Working Group, RFC 2328 OSPF version 2. Available at. https://www.ietf.org/rfc/rfc2328.txt (1998). Accessed 9 June 2015
7. Hohlbaum, F., Braendle, M., Alvarez, F.: Cyber security practical considerations for implementing iec 62351. ABB Technical Report (2010)
8. Xiao, Y., Thulasiraman, K., Xue, G., Jüttner, A.: The constrained shortest path problem: algorithmic approaches and an algebraic study with generalization. AKCE Int. J. Graphs Comb. **2**(2), 63–86 (2005)
9. Kuipers, F., Van Mieghem, P., Korkmaz, T., Krunz, M.: An overview of constraint-based path selection algorithms for QoS routing. IEEE Commun. Mag. **40**(12), 50–55 (2002)

10. Jüttner, A., Szviatovski, B., Mécs, I., Rajkó, Z.: Lagrange relaxation based method for the QoS routing problem. In: Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), vol. 2, pp. 859–868 (2001)
11. Nunes, B., Mendonca, M., Nguyen, X.-N., Obraczka, K., Turletti, T.: A survey of software-defined networking: past, present, and future of programmable networks. IEEE Commun. Surv. Tutorials **16**(3), 1617–1634 (2014)
12. Sydney, A., Ochs, D.S., Scoglio, C., Gruenbacher, D., Miller, R.: Using geni for experimental evaluation of software defined networking in smart grids. Comput. Netw. **63**, 5–16 (2014)
13. Goodney, A., Kumar, S., Ravi, A., Cho, Y.H.: Efficient PMU networking with software defined networks. In: IEEE International Conference on Smart Grid Communications (SmartGridComm), pp. 378–383 (2013)
14. Molina, E., Jacob, E., Matias, J., Moreira, N., Astarloa, A.: Using software defined networking to manage and control IEC 61850-based systems. Comput. Electr. Eng. **43**, 142–154 (2015)
15. Cahn, A., Hoyos, J., Hulse, M., Keller, E.: Software-defined energy communication networks: from substation automation to future smart grids. In: IEEE International Conference on Smart Grid Communications (SmartGridComm), pp. 558–563 (2013)
16. Kim, Y.-J., He, K., Thottan, M., Deshpande, J.G.: Virtualized and self-configurable utility communications enabled by software-defined networks. In: IEEE International Conference on Smart Grid Communications (SmartGridComm), pp. 416–421 (2014)
17. Zhang, J., Seet, B.-C., Lie, T.-T., Foh, C.H.: Opportunities for software-defined networking in smart grid. In: International Conference on Information, Communications and Signal Processing (ICICS), pp. 1–5 (2013)
18. Dong, X., Lin, H., Tan, R., Iyer, R.K., Kalbarczyk, Z.: Software-defined networking for smart grid resilience: opportunities and challenges. In: ACM Workshop on Cyber-Physical System Security, pp. 61–68 (2015)
19. Open networking foundation, software-defined networking: the new norm for networks. ONF White Paper (2012)
20. Project Floodlight. Available at. http://www.projectfloodlight.org/floodlight/ (2015). Accessed 9 June 2015
21. Mininet. Available at. http://mininet.org/ (2015). Accessed 9 June 2015
22. iperf.fr, Iperf. Available at. https://iperf.fr/ (2015). Accessed 9 June 2015
23. xjperf. Available at. https://code.google.com/p/xjperf/ (2015). Accessed 9 June 2015