# Fast Path Recovery for Single Link Failure in SDN-Enabled Wide Area Measurement System

Tong Duan, *Member, IEEE*, and Venkata Dinavahi, *Fellow, IEEE*

***Abstract*—In the wide area measurement system (WAMS), the end-to-end transmission delay between the phasor measurement unit (PMU) and phasor data concentrator (PDC) is strictly constrained for real-time monitoring and protection applications. When a communication link failure happens, fast path recovery is required to reduce the impact of measurement losses. In this work, the promising software-defined network (SDN) technique is leveraged to compute the re-routing path in a global view upon a single link failure. More specifically, a hybrid fast path recovery algorithm (HFPR-A) is proposed based on the principle of simplicity: in some cases, the shortest path or approximate shortest path between PMU and PDC can be recovered by *adding only one edge* to the original forwarding tree; while in the other cases, the shortest paths can be recovered with *lower computational complexity* than the traditional Dijkstra's algorithm. The proposed HFPR-A is implemented on the Ryu + Mininet testbed, and the simulation results on different IEEE benchmark test power systems show that the proposed HFPR-A could find shorter re-routing paths than the existing methods with a low-enough response time.**

***Index Terms*—Communication system, fault-tolerate, industrial network, smart grid, software-defined network, wide area measurement system.**

## I. Introduction

SINCE the concept of "Industry 4.0" was put forward, the information and communication technology (ICT) is increasingly integrated into the industrial infrastructure. In one of the most representative industry field, the power system is also in transition to an ICT-enabled grid, which is called "smart grid" (SG) conceptually: the ICT is utilized to transmit measurements and control commands between sampling/measurement devices, control center, and controllable equipment to achieve fast and automatic operation. In the transmission-level of power grid, the wide area measurement system (WAMS) is the representative ICT infrastructure that plays a crucial role in wide ranging monitoring, operation, and protection. In WAMS, the components mainly include phasor measurement unit (PMU), phasor data concentrator (PDC), control center, and controllable power equipment (such as controllable load, controllable circuit breaker, etc.) [1], [2], and the corresponding communication network is constructed to provide data connection between those WAMS components.

Typically, all the synchrophasor data from PMUs should be first gathered in PDCs, and then forwarded to the higher level control center for application level analysis. The transmission delays between a PDC and its monitored PMUs have stringent requirements for real-time monitoring and control based applications. All the synchrophasor data from different PMUs have a time-stamp that indicates the measurement time, and the PDC needs to wait for all the measurements with the same time-stamp before analysis or forwarding to upper level applications. If the wait time is longer than the pre-determined PDC timer, the PDC will forward the received measurement data without waiting for the rest of the data, which results in the measurement data being incomplete and may induce incorrect operation at system-level. Therefore, one main concern of WAMS is to achieve fast and reliable end-to-end data transmission between PMUs and PDCs. However, once a link failure happens due to the network attacker's malicious intrusion or unpredictable physical breakdown, the constructed data paths can be significantly impacted.

To deal with the link failure, most modern communication networks use fast-reroute (FRR) mechanisms to recover the data transmission, which leverage pre-computed alternative paths at any node towards any destination [3]. The loop free alternative (LFA) [4] is a representative FRR method to quickly recover the path, which simply finds the qualified neighbor of a specific router to re-route the data. However, such a qualified neighbor does not always exist, and thus several extensions of LFA have been proposed such as the remote LFA [5] and recursive LFA [6]. Another kind of FRR method popular in research is to add some tags into the packet header to carry the path information and guide the re-route path selection, as proposed in [7], [8]. The main goal of the above FRR based methods is to quickly find a feasible path to the destination, but cannot guarantee the performance of the new route, which is not suitable for the WAMS scenario with stringent transmission latency requirements. As for the fast path recovery specially designed for WAMS, the multi-path based methods were proposed in [9]–[11]. In [9] the resilient multicast tree is constructed based on robust flocking theory, while in [10] and [11] the multi-paths are computed by solving the elaborate linear programming optimization problems. The

Manuscript received June 22, 2021; revised October 17, 2021; accepted November 28, 2021. Date of publication December 1, 2021; date of current version February 21, 2022. This work was supported in part by the Natural Science and Engineering Research Council of Canada (NSERC) and in part by the National Natural Science Foundation of China under Grant 62101598. Paper no. TSG-00994-2021. *(Corresponding author: Tong Duan.)*

Tong Duan is with the National Digital Switching System Engineering & Technological Research Center, Zhengzhou, 450000, China (e-mail: tduan@ualberta.ca).

Venkata Dinavahi is with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB T6G 2V4, Canada (e-mail: dinavahi@ualberta.ca).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TSG.2021.3131682.

Digital Object Identifier 10.1109/TSG.2021.3131682

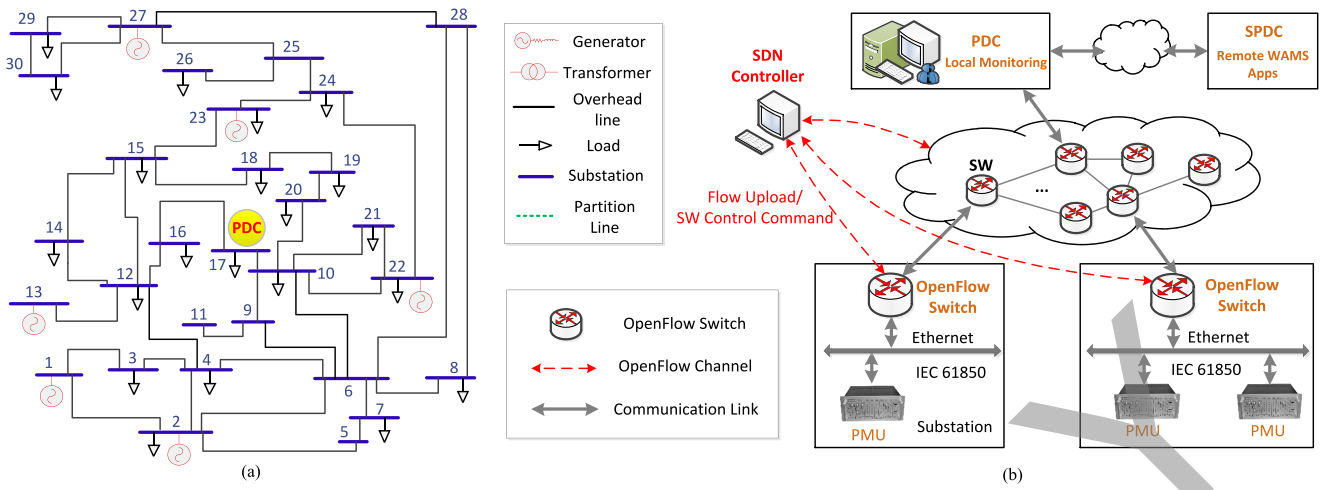



Fig. 1. Example of SDN-Enabled WAMS: (a) IEEE 30-bus test power system; (b) corresponding SDN-enabled communication network.

above works rely on complex meters and numerical solutions, which may be not applicable in practical WAMS.

Instead of re-routing the measurement data in a distributed manner, the centralized control for fast path recovery may be better since the new paths can be instantiated with a global view. The software-defined network (SDN) technology originating from [12] is just such an approach to achieve centralized network resource management. Through the separation of control plane and data plane, SDN can realize programmable data flow transmission. Therefore, the SDN-enabled smart grid has become a hot topic in power grid communication system design [13]. In SDN research, the fast link failure recovery is an old topic [14]. In recent works, the group table was used to quickly re-route the packet in data plane [15], but the recovery path needs to be pre-computed. In fact, the link failure may happen everywhere and thus pre-computing and pre-installing all the back-up paths for all possible link failures is unrealistic. The control plane based recovery methods need to re-compute a new path in the SDN controller, thus the problem formulation and solution method are important. In [16] and [17], the recovery problem is formulated as different linear programming problems based on different optimization goals and different algorithms are proposed. Unfortunately, these algorithms are not specially designed for WAMS. In fact, WAMS has some special features that can be leveraged for a fast path recovery design. For example, in WAMS the PMUs do not communicate with each other and they only need to send measurements to the PDC; thus only the forwarding tree rooted at the PDC node requires to be constructed, and only the path failure happening on the tree path may impact the end-to-end transmission. However, the recent works on the SDN-based WAMS [18], [19] follow the logic of taking concerned meters (such as resource consumption of each switch and the remaining link resources) into linear programming problem formulation and proposing an algorithm to solve the formulated problem, while not taking the special features of WAMS into consideration.

Based on the above discussion it can be observed that, the existing failure recovery algorithms for SDN-enabled WAMS are mainly based on complex meters and properly formulated linear programming problems, which have difficulties to be applied in real networks and have considerable computational complexities. Leveraging the special features of WAMS, in this work, the hybrid fast path recovery algorithm (HFPR-A) is proposed to respond to a single link failure in SDN-enabled WAMS. The HFPR-A is a hybrid method of one-edge adding algorithm and fast shortest path recovery algorithm that applies one of the two algorithms in different cases of link failure. The biggest advantage of HFPR-A is **simplicity**: it can quickly recover the route via very simple logic with low computational complexity and low weight of the new path. Because of its simplicity, it can be easily implemented and applied in a practical WAMS (a representative example is the OSPF routing protocol: although countless routing algorithms have been proposed, the shortest-path Dijkstra's algorithm based OSPF is still the dominant technique due to its simplicity). More specifically, the contributions of the proposed HFPR-A can be summarized as follows:

1) The one-edge adding algorithm can compute the shortest recovery path or $(1+2\alpha)$-approximate $(0 < \alpha < 1)$ shortest recovery path between PMU and PDC by only adding one edge to the original shortest path tree (SPT), which has lower computation time and is easier to implement compared with existing failure recovery methods;
2) The fast shortest path recovery algorithm can compute the shortest recovery path between PDC and its monitored PMUs with lower computational complexity than Dijkstra's algorithm, which can run faster than the traditional Dijkstra's algorithm.

The simulation testbed was constructed on the Ryu-Mininet based SDN environment, and the network topologies of IEEE 14, 24, 30, 39, 57 and 118 bus benchmark test power systems [24] were used. The simulation results on different test power systems show that the proposed HFPR-A could find shorter re-routing paths than the existing methods with lower response time.

## II. SDN-Enabled WAMS Data Transmission

In WAMS, the power grid refers to the transmission-level power system, which mainly contains synchronous generators, transmission lines, substations (buses) and loads, as shown in the example of IEEE 30-bus test power system in Fig. 1(a).

The corresponding ICT infrastructure is composed of the phasor measurement units (PMUs), phasor data concentrators (PDCs), global control center (or called "super PDC", SPDC), controllable power equipment and the communication system that connects those components. The PMUs are installed at bus nodes as the communication terminals to sample the electrical parameters and compute the phasor values as measurement data. Then, the data transmission path is as follows: the measurement data needs to be sent from PMUs to PDC for aggregation, and the aggregated measurement data is analyzed at PDC or directly sent to the SPDC; and the control instructions generated by SPDC need to be sent to the controllable power equipment.

Typically, the communication network connecting PMUs and PDC is separated from that connecting PDCs and SPDC, as shown in Fig. 1(b), because of the logically hierarchical functions of PMU, PDC and SPDC. In this work, only the communication network connecting PMUs and PDCs is of concern, because in this network the link failures easily induce the measurement incompleteness, which may significantly impact the system-level operation. The communication network topology is assumed to be the same as the power grid topology, i.e., each bus node is connected to a forwarding device (router or switch), and the data transmission lines are deployed in parallel with the power transmission lines. This simplification is a commonly used approach to study the power grid communication system [20], since in a real-world power system the substation may also be attached to a router for wide area connection, and the communication links can be deployed along with the power lines for convenient engineering construction. With SDN-enabled, the data transmission mode is changed to a manner different from the traditional networking. As shown in Fig. 1(b), all the forwarding devices are changed to the OpenFlow switches (called "SW" in this work), and the forwarding rules of those SWs are controlled by the SDN controller(s). With a global view of networking status and programmable flow tables in each SW, flexible forwarding strategies and fast link failure detection can be realized.

In WAMS, one PDC monitors many PMUs; therefore, the forwarding tree can be constructed rooting at the PDC-connected SW, and then the data transmission from PMUs to PDC can be established along the tree path. In general, the routing path can be obtained by the shortest path algorithm (Dijkstra's algorithm) that is commonly used in real-world networking. In this work, the forwarding tree is assumed to be the shortest path tree (SPT) rooted at the PDC-connected SW. For the IEEE 30-bus power grid shown in Fig. 1(a), let there have 10 PMUs installed at bus {1, 2, 6, 9, 10, 12, 15, 19, 25, 27} according to the PMU placement algorithm [21], and one PDC is deployed at bus 17. If the weight of each communication link is simply set as the link length for demonstration (in real-world, the link weight can be assigned based on several QoS meters), then the constructed SPT from source (vertex 17, the SW that connects with PDC) to the other SWs is shown in Fig. 2. The bold lines refer to the links that are critical to the PMU-PDC connection since if one of them fails then some PMUs may be unobservable by PDC.

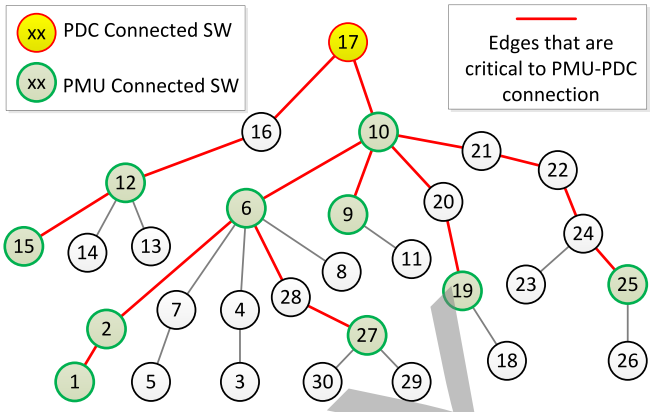



Fig. 2. Shortest path tree (SPT) of the communication network of IEEE 30-bus test power system: rooted at the PDC connected SW.

## III. Proposed Hybrid Fast Path Recovery Algorithm

In this section, two types of fast path recovery algorithms are proposed: one-edge adding algorithm and fast shortest path recovery algorithm, based on the core principle: *simplicity*. The proposed algorithms do not require complicated computation or complex meters, which can be easily implemented and applied. These algorithms make up the hybrid fast path recovery algorithm (HFPR-A) that applies one of these algorithms in different link failure cases.

### A. One-Edge Adding Algorithms

If a link failure happens on the link that is not in the SPT, then it will not impact the end-to-end communication between PMUs and PDC. Therefore, in this work only the case where a failure happens on one single edge of the SPT is analyzed. Let the link failure on the edge of SPT $T$ be denoted as $e_f = (p, q) \in T$ (be default in this paper, $p$ is closer to $s$ in $T$), then $T$ will be partitioned into two subtrees $T(up) = T_q$ and $T(down) = T - e_f - T_q$; in the view of $G$, $G$ will be partitioned into two subgraphs: $G(up)$ that only contains the vertices in $T(up)$ (denoted as $S(up)$) and the links connecting the vertices in $S(up)$, and $G(down)$ that only contains the vertices in $T(down)$ (denoted as $S(down)$) and the links connecting the vertices in $S(down)$. Assume that $G(up)$ and $G(down)$ also have some external links connecting each other, guaranteeing the existence of the alternative paths to recover the communication. The meanings of the related notations are listed in Table I. The derivation of the proposed algorithms begins from *Lemma 1*.

*Lemma 1:* If there is a link failure on the edge $e_f = (p, q) \in T$, the shortest path from $s$ to $q$ in $G - e_f$ can be obtained by adding only one edge to $T$.

*Proof:* Let $P$ be the shortest path from $s$ to $q$ in $G - e_f$, $a$ be the last vertex in $P$ that is also in $S(up)$, $b$ be the first successor of $a$ in $P$. Apparently, $a$ and $b$ must exist, and $P$ is $P(s, a)::(a, b)::P(b, q)$. All the vertices in $P(b, q)$ are in $S(down)$ due to the definition of $a$. And according to the definition of SPT, the shortest path from $s$ to $a$ must be $PATH_T(s, a)$, thus all the vertices in $P(s, a)$ also must be

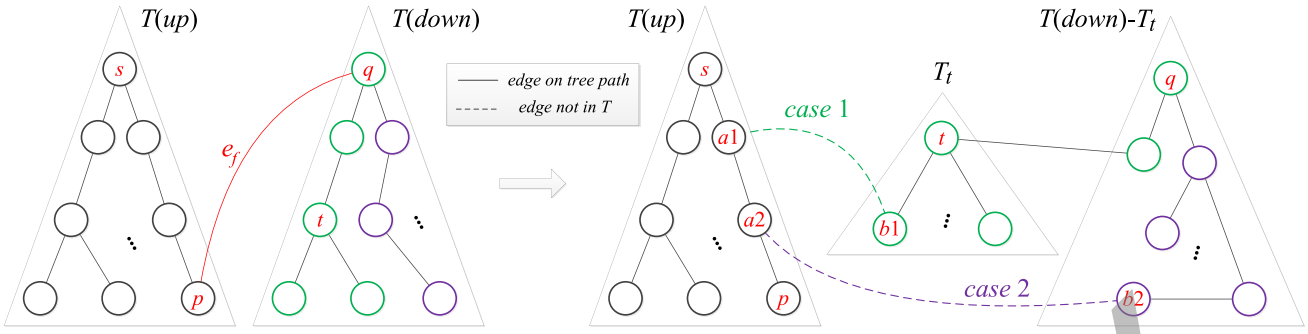



Fig. 3. Example of *Case 1* and *Case 2* of destination vertex $t$.

TABLE I
NOTATION LIST FOR SECTION III(A)

| Notation | Description |
|---|---|
| $G(V,E)$ | The SW undirected graph topology, $n=\|V\|, m=\|E\|$ |
| $e_f$ | The fault link $e_f=(p,q)\in T$ |
| $G-e_f$ | Graph obtained by deleting $e_f\in E$ in $G$ |
| $s\in V$ | Source node that connects with PDC |
| $T$ | Shortest path tree rooted from $s$ |
| $T_v$ | Subtree of $T$ rooted at a vertex $v$ |
| $T(up), T(down)$ | The two subtrees split by a fault edge in $T$ |
| $S(up), S(down)$ | The node set of $T(up), T(down)$ respectively |
| $G(up), G(down)$ | The two subgraphs split by a fault edge in $T$ |
| $m_{up}$ | Number of edges connecting the vertices in $S(up)$ |
| $m_{down}$ | Number of edges connecting the vertices in $S(down)$ |
| $m_{u,d}$ | Number of edges connecting $S(up)$ and $S(down)$ |
| $P$ | A path by default is an acyclic path, and no vertices on the path are visited twice |
| $wt(u,v)$ | Weight of edge $(u,v)$, $wt(u,v)>0$ |
| $wt(P)$ | Sum of the weights of all edges in $P$ |
| $dist_G(u,v)$ | Weight of shortest path from $u$ to $v$ in graph $G$ |
| $PATH_T(u,v)$ | The unique path from $u$ to $v$ on $T$ |
| $P1::P2$ | The path by concatenating paths $P1$ and $P2$, the last vertex in $P1$ is the first vertex of $P2$ |
| $\pi(v)$ | The predecessor vertex of $v$ on the shortest path to $v$ |
| $vertex_S(H)$ | The vertex $b$ in the edge $(a,b)$ returned by $FindMin(S,H)$ defined in *Algorithm 1* |

in $S(up)$, i.e., $P(s,a)=PATH_T(s,a)$. Since $G$ is undirected, $wt(P(b,q))=wt(P(q,b))$. Thus the shortest path from $b$ to $q$ also lies in $T(down)$, i.e., $P(b,q)=PATH_T(b,q)$. Then the shortest path from $s$ to $q$ in $G-e_f$ can be obtained by only adding $(a,b)$ to $T$.

If a single link failure happens on the edge $e_f=(p,q)$ of SPT, the location of the destination vertex $t$ in $S(down)$ may have two cases: (1) *Case 1*, $t$ is located on the tree path $PATH_T(vertex_{S(up)}(S(down)),q)$, or in the other words, $vertex_{S(up)}(S(down))$ is located on the subtree $T_t$ (as illustrated in Fig. 3), where $vertex_{S(up)}(S(down))$ denotes the first vertex on the shortest path from $s$ to $q$ in $G-e_f$ and is also in $S(down)$; (2) *Case 2*, $t$ is not located on the tree path $PATH_T(vertex_{S(up)}(S(down)),q)$, as illustrated in the example in Fig. 3. Note that in *Lemma 1*, the destination vertex $t=q$ belongs to *Case 1*. ■

*Lemma 2:* If there is a link failure on the edge $e_f=(p,q)\in T$, the shortest path from $s$ to $t$ in $G-e_f$ can be obtained by adding only one edge to $T$, if $t$ belongs to *Case 1*.

*Proof:* Let $b_1$ be $vertex_{S(up)}(S(down))$, and $a_1$ be $\pi(b_1)$. According to *Lemma 1*, the shortest path from $s$ to $q$ is $P=PATH_T(s,a_1)::(a_1,b_1)::PATH_T(b_1,q)$. According to the characteristic of shortest path, the sub-path of $P$ from $s$ to any vertex on the $P$ is also the shortest path. In *Case 1*, the destination vertex $t$ is located on $PATH_T(b_1,q)$, then the shortest path from $s$ to $t$ must be: $P(s,t)=PATH_T(s,a_1)::(a_1,b_1)::PATH_T(b_1,t)$, as also illustrated in Fig. 3. Based on *Lemma 1* and *Lemma 2*, the *Algorithm 1* for *Case 1* is proposed.

As shown in *Algorithm 1*, for the communication link recovery from the PMU connecting with the SW node $t$ to PDC, the only required computation in *Case 1* is to get the two vertices $a$ and $b$. The vertices $a$ and $b$ can be found by searching for all the edges connecting $S(down)$ with $S(up)$. Assume the results of the SPT construction running for the original graph is stored, i.e., $dist_G(s,x)$ and $dist_G(y,q)$ for any $\{(x,y)|x\in S(up), y\in S(down)\}$ can be obtained directly, then the computational complexity of *Algorithm 1* is $O(m_{u,d})$. Since

$$m_{up}+m_{down}+m_{u,d}=m, \tag{1}$$

and there are already $(N-1)$ edges in SPT, in the worst case the computational complexity of *Algorithm 1* is $O(m-(N-1-1))=O(m-N)$. ■

For the vertex $t$ that does not belong to *Case 1*, the following analysis is proposed to find the solution for *Case 2*.

*Lemma 3:* If there is a link failure on the edge $e_f=(p,q)\in T$, the $(1+2\alpha)-approximate$ $(0<\alpha<1)$ path from $s$ to $t$ in $G-e_f$ can be obtained by adding only one edge to $T$, if $t$ belongs to *Case 2*. (Note: the path is said to be a $\gamma$-approximate path, if for any instance of the recovery problem, the output path of the algorithm has a cost at most $\gamma$ times the cost of the optimal path.)

*Proof:* Let the path $P$ be the shortest path from $s$ to $t$, and let $Q$ be the constructed path by adding only one edge to $T$. In this case, let $b_2=vertex_{S(up)}(S(down))$, $a_2$ be $\pi(b_2)$. Then the constructed path is $Q=PATH_T(s,a_2)::(a_2,b_2)::PATH_T(b_2,t)$, and the weight of path $Q$ is:

$$\begin{aligned} wt(Q) &= dist_{G-e_f}(s,b_2)+wt(PATH_T(b_2,t)) \\ &\le \left[dist_{G-e_f}(s,b_2)+wt(PATH_T(b_2,q))\right] \\ &\quad + wt(PATH_T(q,t)) \\ &= dist_{G-e_f}(s,q)+dist_{G-e_f}(q,t) \\ &\le \left[dist_{G-e_f}(s,t)+dist_{G-e_f}(t,q)\right]+dist_{G-e_f}(q,t) \end{aligned}$$

**Algorithm 1** Path Recovery for the PMU Connecting With SW $t$ in *Case 1*

**Require:** The SPT $T$ and two sets $S(up)$ and $S(down)$ split by $e_f = (p, q)$;

$(a, b) = FindMin(S(up), S(down))$;
**if** $t$ is located on the tree path $PATH_T(q, b)$ **then**
  //*Case 1*
  $P = PATH_T(s, a)::(a, b)::PATH_T(b, t)$;
**end if**
**Function** $(a, b) = FindMin(S(up), S(down))$
{
$val_{min} = inf$;
**for** $e \in \{(x, y)|x \in S(up), y \in S(down)\}$ **do**
  **if** $dist_G(s, x) + wt(x, y) + dist_G(y, q) < val_{min}$ **then**
    $val_{min} = dist_G(s, x) + wt(x, y) + dist_G(y, q)$;
    $a = x, b = y$;
  **end if**
**end for**
}

**Algorithm 2** Path Recovery for the PMU Connecting With SW $t$ in *Case 2*

**Require:** The SPT $T$ and two sets $S(up)$ and $S(down)$ split by $e_f = (p, q)$;

$(a, b) = FindMin(S(up), S(down))$;
**if** $t$ is not located on the tree path $PATH_T(q, b)$ **then**
  //*Case 2*
  $P_{(1+2\alpha)-approximate} = PATH_T(s, a)::(a, b)::PATH_T(b, t)$.
**end if**

$$\begin{aligned} &= dist_{G-e_f}(s, t) + 2 \times dist_{G-e_f}(q, t) \\ &= dist_{G-e_f}(s, t) + 2\alpha \times dist_{G-e_f}(s, t) \\ &= (1 + 2\alpha) \times dist_{G-e_f}(s, t) \\ &= (1 + 2\alpha) \times wt(P) \end{aligned} \tag{2}$$

where $\alpha = dist_{G-e_f}(q, t)/dist_{G-e_f}(s, t) \in (0, 1)$. Therefore, the constructed path $Q$ is a $(1 + 2\alpha) - approximate$ path of the shortest path from $s$ to $t$. Based on *Lemma 3*, the path construction procedure is described in *Algorithm 2*. The computational complexity of *Algorithm 2* is also $O(m_{u,d})$, which is the same as *Algorithm 1*. ■

### B. Fast Shortest Path Recovery Algorithm

Although the fast path recovery can be achieved by adding only one edge to $T$, the constructed path in *Case 2* is not optimized, and its performance strongly depends on the location of the faulted edge that determines the value of $\alpha$:

$$\begin{aligned} \alpha &= \frac{dist_{G-e_f}(q, t)}{dist_{G-e_f}(s, t)} \\ &\le \frac{dist_{G-e_f}(q, t)}{dist_G(s, t)} = \frac{dist_G(q, t)}{dist_G(s, q) + dist_G(q, t)} \\ &= 1 - \frac{dist_G(s, q)}{dist_G(s, t)} = \alpha_{ub} \end{aligned} \tag{3}$$

where $\alpha_{ub}$ is the upper bound of $\alpha$ and easy to compute, since the shortest path weight $dist_G(s, v)$, $v \in G$ can be stored in the first construction of SPT. From (3) it can be observed that, if the failure happens on the link close to PDC (i.e., $q$ is close to $s$), $\alpha_{ub}$ will be close to 1. In general, it is preferred that $\alpha_{ub}$ is close to 0. For example, if the link failure happens on the edge (22,24) in Fig. 2, then $q$ is vertex-24 and $t$ can only be the vertex-25, and the large $dist_G(s, q)/dist_G(s, t)$ results in a small $\alpha_{ub}$ that is preferred. In practical cases, the threshold of $\alpha$ can be set (as $\alpha_\varepsilon$) to first determine the selection of used algorithms: if $\alpha_{ub} \le \alpha_\varepsilon$, the *Algorithm 2* constructed path is acceptable to make a reasonable re-connection; and if $\alpha_{ub} > \alpha_\varepsilon$, then the performance of the constructed path may be not good and a shortest path is required to be computed. In this part, the fast shortest path recovery algorithm is proposed for the case $\alpha_{ub} > \alpha_\varepsilon$.

**Algorithm 3** Path Recovery for the PMU Connecting With $t$ in *Case 2*

**Require:** The SPT $T$ and two sets $S(up)$ and $S(down)$ split by $e_f = (p, q)$, resulting in $\alpha_{ub} > \alpha_\varepsilon$;

$S = S(up)$, $Q = S(down)$;
**for** each vertex $u \in S$, $v \in Q$ **do**
  $d[u] = dist_G(s, u)$, $d[v] = \infty$;
**end for**
**for** each vertex $u \in Adj[v](v \in S)$ **do**
  RELAX$(v, u, wt)$;
**end for**
**while** $Q \neq \varnothing$ **do**
  $u \longleftarrow$ EXTRACT-MIN$(Q)$;
  $S = S \cup \{u\}$;
  **for** each vertex $v \in Adj[u]$ **do**
    RELAX$(u, v, wt)$;
  **end for**
**end while**

A natural idea is to re-run the Dijkstra's algorithm for $S$ and $Q$ (the two sets hold by the Dijkstra's algorithm [23]), and set $S = S(up)$ and $Q = S(down)$ to reuse the results of the original construction of SPT. However, the orders of adding vertices into $S$ in the original run of Dijkstra's algorithm may not result in $S = S(up)$, which makes the results for the original $G$ cannot be reused. In this work, the *Algorithm 3* is proposed to run Dijkstra's algorithm based on $S = S(up)$ and $Q = S(down)$: first update the distance from $s$ to every $v \in Q$ (denoted as $d[v]$), and then continue the Dijkstra's procedure for $S = S(up)$ and $Q = S(down)$. The update of $d[v]$ for every $v \in Q$ can be achieved by the RELAX() function [23]. Then the whole process is shown in *Algorithm 3*, where EXTRACT-MIN($Q$) is the function to find the minimum $d[u], u \in Q$ and remove $u$ from $Q$ [23].

Note that the *Algorithm 3* not only constructs the shortest path to $t$, but also reconstructs the whole SPT since it finds shortest paths for all the vertices in $S(down)$. It can be observed that step 5∼7 is to relax the edges connecting the vertices between $S(up)$ and $S(down)$, and then step 11∼13 is to relax the edges connecting the vertices in $S(down)$, therefore, the computational cost for all the relaxation operation is $O(m - m_{up})$, where $m_{up}$ is the number of edges connecting the vertices in $S(up)$. And assume the number of vertices in

TABLE II
NOTATION LIST FOR SECTION III(B)

| Notation | Description |
|---|---|
| $\alpha_{ub}$ | The upper bound of $\alpha$ |
| $\alpha_{\varepsilon}$ | The threshold set for $\alpha_{ub}$ |
| $S$ | The set hold by Dijkstra's algorithm, the shortest path from $s$ to $v \in S$ is already found |
| $Q$ | The set hold by Dijkstra's algorithm, $Q = V - S$ |
| $d[v]$ | The distance from $s$ to $v$, may be updated in each circulation |
| $Adj[v]$ | The vertices in $Q$ that directly connects with $v$ |
| $n_{up}$ | Number of vertices in $S(up)$ |
| $n_{down}$ | Number of vertices in $S(down)$, $n_{down} = n - n_{up}$ |

$S(up)$ is $n_{up}$, then the total number of required operations for step 9 is:

$$\begin{aligned} C &= O(n_{down}) + O(n_{down} - 1) + \cdots + O(1) \\ &= O\left[\frac{1}{2} n_{down}(n_{down} + 1)\right] \\ &= O\left[\frac{1}{2}(n - n_{up})(n - n_{up} + 1)\right]. \end{aligned} \tag{4}$$

Therefore, the total computational cost is $O[\frac{1}{2}(n-n_{up})(n-n_{up}+1)+(m-m_{up})] = O[(n-n_{up})^2+(m-m_{up})]$, which is related to the vertex and edge sets of $G(up)$. Although in the worst case ($n_{up} = 1, m_{up} = 0$), *Algorithm 3* and Dijkstra's algorithm have the same complexity, in other cases, the $\frac{1}{2}(n-n_{up})(n-n_{up}+1)+(m-m_{up})$ operations is always smaller than that in Dijkstra's algorithm ($\frac{1}{2}n(n+1)+m$ operations). Therefore, compared to re-running Dijkstra's algorithm for $G - e_f$, the computational cost of *Algorithm 3* is smaller in practice, especially if $G(up)$ is large (which means $G(up)$ contains large number of vertices and edges).

### C. HFPR-A Procedure

The one-edge adding algorithms are proposed for re-routing the data flow to a single vertex $t$, however, there may have multiple PMU-connected SWs required to be re-connected to PDC. The SWs may also have mix cases of *Case 1* and *Case 2*. Therefore, based on the three proposed algorithms, the hybrid fast path recovery algorithm (HFPR-A) is proposed to apply proper algorithms in different cases, and the processing flow of HFPR-A is shown in Fig. 4. It can be seen that HFPR-A executes three separate procedures depending on different cases:

1) *Procedure 1:* If all the SWs $t$ in $S(down)$ belong to *Case 1*, the shortest paths from $s$ to all the vertices $t$ can be obtained by adding only one edge to the SPT, with computational cost of $O(m_{u,d})$; (*Algorithm 1*)
2) *Procedure 2:* If all the SWs $t$ are either in *Case 1* or in *Case 2* with $\alpha_{ub} \leq \alpha_{\varepsilon}$, the shortest path or $(1+2\alpha)-$*approximate* $(0 < \alpha < 1)$ paths from $s$ to all the vertices $t$ can be obtained by adding only one edge to the SPT, with computational cost of $O(m_{u,d})$; (hybrid *Algorithm 1* and *Algorithm 2*)
3) *Procedure 3:* If there are SWs $t$ in *Case 2* with $\alpha_{ub} > \alpha_{\varepsilon}$, the shortest paths from $s$ to all the vertices $t$ are constructed with computational cost of $O[(n-n_{up})^2 + (m-m_{up})]$. (*Algorithm 3*).

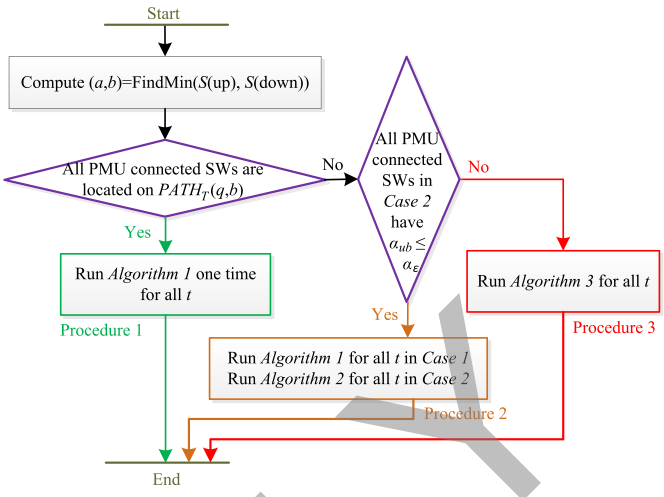



Fig. 4. Process flow of HFPR-A with three separate procedures.

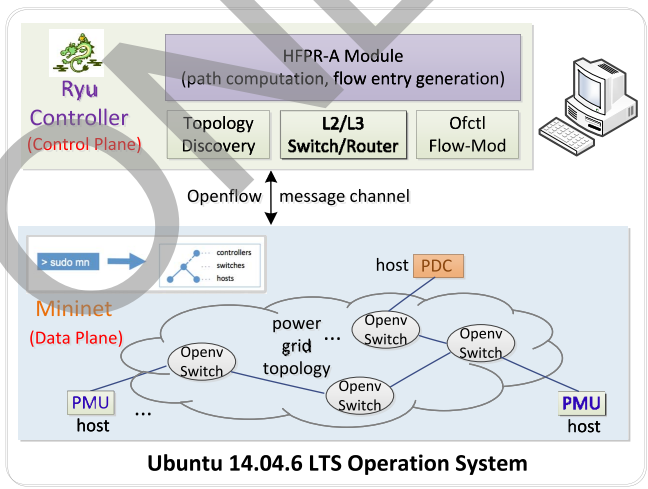



Fig. 5. The Ryu + Mininet simulation platform: the communication network is build in Mininet according to the test power system topology.

## IV. IMPLEMENTATION AND EVALUATION

In this section, the SDN-based testbed implementation is described to evaluate the proposed HFPR-A over the IEEE benchmark test power systems [24], and the comparison between different path recovery algorithms are performed to show the advantages of the proposed algorithms.

### A. Testbed Implementation

The SDN environment was constructed on the Ryu-Mininet platform. Mininet is a network emulator which can create a realistic virtual network of virtual hosts, switches (OpenvSwitch), and links. Ryu is a component-based SDN framework, entirely written in Python [25]. Ryu provides software components with well defined API that make it easy for developers to create new network management and control applications. The connection between Mininet and Ryu is based on the Openflow message channel. As shown in Fig. 5,

in this work, the Ryu controller and Mininet are installed and run on the Intel Core i7-10710U 1.61GHZ CPU with 16G RAM.

*Data Plane Configuration:* The PMUs and PDCs are instantiated as "hosts" in Mininet. The PMUs periodically generate data packets at the rate of 30Hz, which emulates the reporting operation. Since the data transmission is unidirectional from PMUs to PDC, the *ping* command (as bidirectional data transmission) is not suitable to generate the test data flow of WAMS. In this work, the Iperf tool is used to generate the data flow from PMUs to PDC. The PDC host acts as the *server* in Iperf, while the PMUs act as the multiple *clients* in Iperf and send packets at 30pps. However, Iperf could not measure the most concerned meter – end-to-end transmission delay between PMUs and PDC, so after the data flow is generated, the *ping* command is used to test the transmission delays for performance evaluation, which only contributes a small amount of data and does not impact the main data streams. After the topology is generated and the PMUs start to send packets to the PDC, the link failure should be configured. The "link.delete()" function is used to disconnect two SWs during run-time, which emulates a single link failure.

*Control Plane Programming:* The Ryu controller monitors the generated Mininet virtual network at run-time. The main implementation work in the Ryu controller was completed in the python based application layer. The Ryu project already provides some example applications for reference, such as the switching/router function, and topology discovery function etc., which is the foundation of the HFPR-A application in this work. Based on the global connecting information, the HFPR-A application first constructs the SPT rooted at the PDC-connected SW, and pushes required flow entries to corresponding SWs. Once a failure happens on the critical links of SPT, the HFPR-A application will receive the "packet-in" message from the ingress switch of the fault link. In this work, the HFPR-A application is specifically programmed for the single link failure, so once it receives the "packet-in" message it can directly know the location of the fault link and starts to compute the new forwarding rules without invoking the global link states. The exploited application may be too simple to be directly applied in real networks, however, it can demonstrate the proposed HFPR-A scheme and be integrated to the complete control applications.

### B. Performance Evaluation

To evaluate the proposed fast path recovery schemes, the IEEE 14, 24, 30, 39, 57 and 118 bus benchmark test power systems were used. As mentioned in Section II, each bus is assumed to have a SW installed, and the communication links are set as parallel with the transmission lines in the power grid. The length of a communication link is set to be the same as the length of the corresponding power transmission line; and if two buses are connected via a transformer, then in the communication network the link between the two corresponding SWs is set as 500m, which means the two SWs are regarded as deployed at neighbor places and have the minimum transmission delay. The PMU locations in the IEEE 14, 24, 30, 39 bus test power system topologies are assigned according to the work [21], and the PMU locations in the IEEE 57 and 118 bus test power systems are assigned according to [22], as shown in Table III. Note that in those communication networks, assume there is only one PDC deployed nearly at the "center" of the topology (may be not the optimal location for PDC placement), as shown in Table III. In real power systems, there may be several PDCs, and in that case, the HFPR-A application just needs to construct SPTs for different PDCs, and applies HFPR-A for each SPT.

TABLE III
PMU/PDC LOCATIONS IN DIFFERENT TEST POWER SYSTEMS

| Test System | PDC Location | PMU Location (Numbers of PMU) |
|---|---|---|
| 14-bus | Bus 11 | 2 6 7 9 (4) |
| 24-bus | Bus 11 | 2 3 8 10 16 21 23 (7) |
| 30-bus | Bus 17 | 1 2 6 9 10 12 15 19 25 27 (10) |
| 39-bus | Bus 16 | 2 6 9 10 11 14 17 20 22 23 25 29 (12) |
| 57-bus | Bus 22 | 1 4 8 10 20 21 24 28 31 32 36 41 44 46 49 52 55 57 (18) |
| 118-bus | Bus 69 | 2 5 10 11 12 17 20 23 25 29 34 37 40 45 49 50 51 52 59 65 66 71 75 77 80 85 87 91 94 101 105 110 114 116 (34) |

TABLE IV
PROBABILITIES OF HFPR-A PROCEDURE 1,2,3 IN EACH TOPOLOGY

| Topology | $m$ | $m_c$ | $m_c/m$ | $\alpha_\varepsilon$ | $m_1$ | $m_2$ | $m_3$ |
|---|---|---|---|---|---|---|---|
| 14-vertex | 20 | 6 | 0.3 | 0.25<br>0.5 | 3 | 1<br>2 | 2<br>1 |
| 24-vertex | 34 | 14 | 0.41 | 0.25<br>0.5 | 6 | 1<br>5 | 7<br>3 |
| 30-vertex | 41 | 16 | 0.39 | 0.25<br>0.5 | 7 | 3<br>6 | 6<br>3 |
| 39-vertex | 46 | 24 | 0.52 | 0.25<br>0.5 | 6 | 4<br>10 | 14<br>8 |
| 57-vertex | 79 | 37 | 0.47 | 0.25<br>0.5 | 15 | 9<br>17 | 13<br>5 |
| 118-vertex | 184 | 58 | 0.32 | 0.25<br>0.5 | 19 | 8<br>24 | 26<br>10 |

First, the randomized single link failure was generated in the six test power system topologies, and the probabilities of applying HFPR-A procedure 1, 2, 3 ($p_1, p_2, p_3$) were evaluated based on the constructed SPTs. The total number of edges ($m$), the number of critical edges for PMU-PDC connection ($m_c$) are listed in Table IV. It can be inferred that, the sum of $p_1$, $p_2$ and $p_3$ is nearly $m_c/m$, since the failure that happens on non-critical links does not impacts the communication between PMUs and PDC. Besides, probability of procedure 2 and 3 depends on the specific configuration of $\alpha_\varepsilon$; in this work, $\alpha_\varepsilon$ is set at 0.25 and 0.5 to test the performance respectively, which result in a 1.5- and 2-approximate path by HFPR-A procedure 2. Then, the number of links ($m_1, m_2, m_3$) on which the failure happens will induce the application of HFPR-A procedure 1, 2, 3 are listed in Table IV. For example, in the 14-vertex topology with $\alpha_\varepsilon = 0.25$, if a link failure happens on the $m_1 = 3$ specific edges, the HFPR-A procedure 1 will be applied. Therefore, in each topology with each $\alpha_\varepsilon$, $p_i = m_i/m$, $i \in \{1, 2, 3\}$. It can be seen that with the increase of $\alpha_\varepsilon$, $p_2 + p_3$ remains the same, but the probability of applying HFPR-A procedure 2 ($p_2$) increases while $p_3$ decreases. That means the one-edge adding algorithm is more likely used when $\alpha_\varepsilon$ is set at a relatively large value.

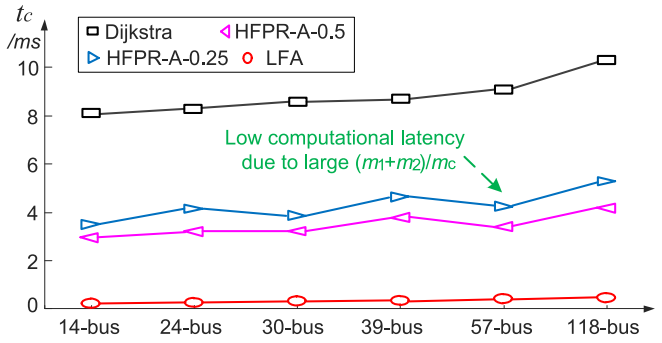



Fig. 6. Average computational latencies of LFA, HFPR-A and Dijkstra's algorithm upon a randomized link failure on the critical edges of the six test topologies.

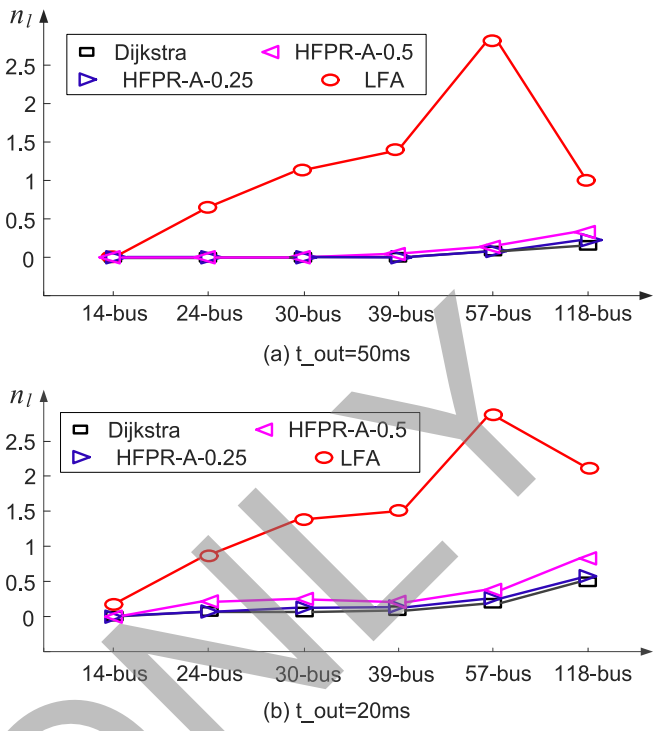



Fig. 7. Average number of PMUs with large PMU-PDC end-to-end transmission latencies resulting from LFA, HFPR-A and Dijkstra's algorithm.

Then, a randomized single link failure was generated on the $m_c$ critical edges of the six test topologies to apply the proposed HFPR-A, and the results were compared with the LFA and Dijkstra-based re-routing algorithms, because the LFA is a representative FRR-based method that is commonly used, while the Dijkstra's algorithm represents a fast (maybe fastest) solution method for the link cost based path selection problem. Two parameters were measured to show the performance: average computational latency of the applied algorithm ($t_c$) that determines the packet losses during path recovery, and average number of PMUs with large PMU-PDC end-to-end transmission latencies ($n_l$) that determines the packet losses after path recovery. Here, "large end-to-end transmission latency" means that the PMU-PDC transmission latency is larger than the predetermined PDC timer ($t_{out}$), and those packets with large transmission latencies will be dropped by PDC. A typical value of $t_{out}$ is 50ms, and in this work $t_{out}$ is set at 50ms and 20ms respectively to evaluate the performance under normal and stringent delay requirements. The results are shown in Fig. 6 and Fig. 7, where HFPR-A-0.25 and HFPR-A-0.5 denote the results under $\alpha_\varepsilon = 0.25$ and $\alpha_\varepsilon = 0.5$ respectively. Note that LFA is a distributed algorithm, which re-routes the packet to a neighbor SW of $s$, assuming that the neighbor SW knows how to route the packet to destination if possible. This work implemented LFA in the centralized SDN network environment, so the LFA application not only needs to find the proper neighbor SW, but also needs to find the new route from the neighbor SW to destinations. When evaluating the $t_c$ of LFA, only the latency of finding the neighbor SW and generating related flow entries is recorded to simulate the performance in distributed networks.

From the results in Fig. 6 it can be observed that, LFA always has the smallest computational latency, since it simply search the neighbor SWs of the source SW; and HFPR-A has much smaller computational latencies than Dijkstra's algorithm. When $\alpha_\varepsilon = 0.5$, the average computational latency of HFPR-A is smaller than that of $\alpha_\varepsilon = 0.25$, since with a larger $\alpha_\varepsilon$, the HFPR-A procedure 1 and 2 with lower computational complexities are more likely to be applied. It is noticed that the computational latencies of HFPR-A in the IEEE 57-bus topology are even smaller than those of IEEE 39-bus topology despite of the larger topology scale, because in the IEEE 57-bus topology the probability of applying HFPR-A procedure 1 and 2 is large due to the large $(m_1 + m_2)/m_c$ shown in Table IV.

As for the average number of PMUs with large PMU-PDC transmission latencies ($n_l$) shown in Fig. 7, it can be seen that LFA results in a much larger $n_l$, which means that in many cases LFA could neither find the qualified neighbor SW nor find the re-routing paths with low transmission delays. Thus LFA is not a good solution for path recovery in WAMS. In the topology of IEEE 57-bus test power system, LFA could not find qualified neighbor SWs in almost every failure location, which results in a large $n_l$ compared to other topologies. When the PDC timer reduces to 20ms, more PMUs cannot reach to the PDC within the timer. Especially in the IEEE 118-bus topology, the increase of $n_l$ is obvious. However, although HFPR-A may not find the shortest path, the results show that the 1.5-approximate paths when $\alpha_\varepsilon = 0.25$ somehow guarantee pretty good paths that can achieve similar performance with Dijkstra's algorithm. When $\alpha_\varepsilon = 0.5$, $n_l$ increases, which means that for delay-sensitive WAMS applications, $\alpha_\varepsilon$ should be set at a small value. Combining the computational latency and transmission latency, it can be concluded that HFPR-A can re-route the data packets with less packet losses during path recovery than Dijkstra' algorithm, while guaranteeing small enough PMU-PDC transmission latencies of the new route.

According to the evaluation results, when HFPR-A is applied in real WAMS networks, $\alpha_\varepsilon$ needs to be configured specifically for different topologies. For example, in a small-scale network, $\alpha_\varepsilon$ can be set at a large value because

even a 3-approximate path may have a low enough end-to-end transmission latency that can meet the requirements of WAMS applications. But for a larger network (such as the 118-bus system), the configuration of $\alpha_\varepsilon$ should also consider the performance of output recovery path and a small $\alpha_\varepsilon$ may be preferred. Fortunately, since the proposed HFPR-A runs in the SDN controller, the impacts of $\alpha_\varepsilon$ on $t_c$ and $n_l$ can be easily pre-evaluated for a specific topology and then $\alpha_\varepsilon$ can be set at a proper value according to the latency requirements of WAMS applications.

## V. Conclusion

In this work, the hybrid fast path recovery algorithm (HFPR-A) is proposed for fast path recovery of single link failure in the WAMS. The software-defined network (SDN) enabled WAMS is first described, which can facilitate the computation of the re-routing path in a global view. Then, the one-edge adding and fast path recovery algorithms are proposed with low computational complexities and simple processing procedures. HFPR-A is the combination of the proposed algorithms. The experimental results show that in the six IEEE benchmark test power system topologies the HFPR-A can achieve much lower recovery time than the traditional Dijkstra's algorithm while obtaining similar re-routing path properties. Due to its simplicity, the SDN-based framework and the proposed HFPR-A can be practically used by communication infrastructure providers of the WAMS system to construct resilient networks with a centralized control manner. Corresponding solutions for the cases with multiple link failures or with limited link capacities will be investigated in future work.

## References


[1] A. G. Phadke, "Synchronized phasor measurements in power systems," *IEEE Comput. Appl. Power*, vol. 6, no. 2, pp. 10–15, Apr. 1993.

[2] J. D. L. Ree, V. Centeno, J. S. Thorp, and A. G. Phadke, "Synchronized phasor measurement applications in power systems," *IEEE Trans. Smart Grid*, vol. 1, no. 1, pp. 20–27, Jun. 2010.

[3] M. Chiesa, A. Kamisiński, J. Rak, G. Rétvári, and S. Schmid, "A survey of fast-recovery mechanisms in packet-switched networks," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 2, pp. 1253–1301, 2nd Quart., 2021.

[4] A. Atlas and A. Zinin, Eds., "Basic specification for IP fast reroute: Loop-free alternates," IETF, RFC 5286, Sep. 2008.

[5] S. Bryant, C. Filsfils, and M. Shand, "Remote LFA FRR," IETF, Internet Draft, draft-shand-remote-lfa-00, Oct. 2011.

[6] S. S. Lor, R. Ali, R. Landa, and M. Rio, "Recursive loop-free alternates for full protection against transient link failures," in *Proc. IEEE Symp. Comput. Commun.*, Riccione, Italy, 2010, pp. 44–49.

[7] E. Rosenberg and J. Uttaro, "A fast re-route method," *IEEE Commun. Lett.*, vol. 77, no. 8, pp. 1656–1659, Aug. 2013.

[8] J. Liu, A. Panda, A. Singla, B. Godfrey, M. Schapira, and S. Shenker, "Ensuring connectivity via data plane mechanisms," in *Proc. 10th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, Lombard, IL, USA, 2013, pp. 113–126.

[9] J. Wei and D. Kundur, "GOAliE: Goal-seeking obstacle and collision evasion for resilient multicast routing in smart grid," *IEEE Trans. Smart Grid*, vol. 7, no. 2, pp. 567–579, Mar. 2016.

[10] R. Kateb, P. Akaber, M. H. K. Tushar, A. Albarakati, M. Debbabi, and C. Assi, "Enhancing WAMS communication network against delay attacks," *IEEE Trans. Smart Grid*, vol. 10, no. 3, pp. 2738–2751, May 2019.

[11] B. Zhou, Q. Yang, Y. Wang, and C. Wu, "Reliable communication in transmission grids based on nondisjoint path aggregation using software-defined networking," *IEEE Trans. Ind. Informat.*, vol. 15, no. 2, pp. 842–855, Feb. 2019.

[12] N. McKeown *et al.*, "OpenFlow: Enabling innovation in campus networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, 2008.

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

[14] F. A. Kuipers, "An overview of algorithms for network survivability," *ISRN Commun. Netw.*, vol. 2012, Dec. 2012, Art. no. 932456.

[15] S. Petale and J. Thangaraj, "Link failure recovery mechanism in software defined networks," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 7, pp. 1285–1292, Jul. 2020.

[16] S. A. Astaneh and S. S. Heydari, "Optimization of SDN flow operations in multi-failure restoration scenarios," *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 3, pp. 421–432, Sep. 2016.

[17] K. Qiu, J. Zhao, X. Wang, X. Fu, and S. Secci, "Efficient recovery path computation for fast reroute in large-scale software-defined networks," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 8, pp. 1755–1768, Aug. 2019.

[18] M. Rezaee and M. H. Y. Moghaddam, "SDN-based quality of service networking for wide area measurement system," *IEEE Trans. Ind. Informat.*, vol. 16, no. 5, pp. 3018–3028, May 2020.

[19] Y. Qu, G. Chen, X. Liu, J. Yan, B. Chen, and D. Jin, "Cyber-resilience enhancement of PMU networks using software-defined networking," in *Proc. IEEE Int. Conf. Commun. Control Comput. Technol. Smart Grids (SmartGridComm)*, Tempe, AZ, USA, Nov. 2020, pp. 1–7.

[20] S. C. Müller *et al.*, "Interfacing power system and ICT simulators: Challenges, state-of-the-art, and case studies," *IEEE Trans. Smart Grid*, vol. 9, no. 1, pp. 14–24, Jan. 2018.

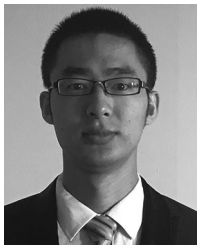
[21] S. Chakrabarti and E. Kyriakides, "Optimal placement of phasor measurement units for power system observability," *IEEE Trans. Power Syst.*, vol. 23, no. 3, pp. 1433–1440, Aug. 2008.

[22] X. Zhu, M. H. F. Wen, V. O. K. Li, and K.-C. Leung, "Optimal PMU-communication link placement for smart grid wide-area measurement systems," *IEEE Trans. Smart Grid*, vol. 10, no. 4, pp. 4446–4456, Jul. 2019.

[23] H. C. Thomas, E. L. Charles, L. R. Ronald, and S. Clifford, *Introduction to Algorithms*, 3rd ed. Cambridge, MA, USA: MIT Press, 2009, p. 658.

[24] S. Peyghami, P. Davari, M. Fotuhi-Firuzabad, and F. Blaabjerg, "Standard test systems for modern power system analysis: An overview," *IEEE Ind. Electron. Mag.*, vol. 13, no. 4, pp. 86–105, Dec. 2019.

[25] "RYU SDN Framework." RYU Project Team. [Online]. Available: https://book.ryu-sdn.org/en/Ryubook.pdf (Accessed: May 2021).

**Tong Duan** (Member, IEEE) received the B.Eng. degree in electronic engineering from Tsinghua University, Beijing, China, in 2013, and the Ph.D. degree in energy systems from the University of Alberta, Edmonton, AB, Canada, in 2021. He is currently a Research Assistant with the National Digital Switching System Engineering and Technological Research Center, Zhenzhou, China. His research interests include smart grid, industrial network, power system simulation, and communication networking.

**Venkata Dinavahi**, photograph and biography is not available at the time of publication.