applied sciences

MDPI

Article

TFR: A Novel Approach for Clock Synchronization Fault Recovery in Precision Time Protocol (PTP) Networks

Alfarooq Omar Alshaikhli and Jong Myung Rhee *

Department of Information and Communications Engineering, Myongji University, 116 Myongji-ro, Yongin, Gyeonggi 17058, Korea; farokalshekly@hotmail.com

* Correspondence: jmr77@mju.ac.kr; Tel.: +82-10-5227-0419; Fax: +82-31-330-6824

Received: 17 November 2017; Accepted: 21 December 2017; Published: 24 December 2017

Abstract: Accurate and precise clock synchronization is one of the fundamental requirements for various applications, such as telecommunication systems, measurement and control systems, and smart grid systems. Precision time protocol (PTP) was designed and specified in IEEE 1588 to meet that requirement. PTP provides a mechanism for synchronizing the clocks in a PTP system to a high degree of accuracy and precision based on exchange synchronization messages through a master–slave hierarchy. The best master clock (BMC) algorithm is currently used to establish the master–slave hierarchy for PTP. However, the BMC algorithm does not provide a fast recovery mechanism in case of master failures. The accuracy and precision of the PTP clocks could be deteriorated by the occurrence of failure in the network (link or node failure). These fault occurrences will affect network performance and reliability, and cause clock time drifting of the PTP nodes. In this paper, we present a novel approach, called timing fault recovery (TFR), to significantly reduce clock time drifting in PTP systems. TFR detects the fault occurrence in the network and recovers it by using a handshake mechanism with a short duration. Therefore, the TFR approach provides clock stability and constancy and increases the reliability and the availability of PTP systems. The performance of TFR has been analyzed and compared to that of the standard PTP. Various simulations were conducted to validate the performance analysis. The results show that, for our sample network, the TFR approach reduces clock drifting by 90% in comparison to the standard PTP, thus providing better clock firmness and synchronization accuracy for PTP clocks.

Keywords: IEEE 1588; precision time protocol (PTP); timing fault recovery (TFR); clock synchronization; fast recovery

1. Introduction

Recently, high-precision clock synchronization has become one of the key requirements for distributed systems in many applications, such as telecommunication systems, measurement and control systems, and smart grid systems. In distributed systems, clocks are synchronized by means of dedicated messages and protocols through communication networks. Two prevalent protocols are used for clock synchronization in these networks: network time protocol (NTP) [1] and precision time protocol (PTP) [2]. NTP is widely used to synchronize computer clocks on the internet. The current version, NTPv4 [3], was formalized in IETF RFCs 5905–5908 in 2010. The primary drawback of NTP is the limitation level of the accuracy of milliseconds, although it is cost-effective and it does not require any specific hardware assistance. For distributed computer networks in information technology (IT) environments, this is sufficient. However, several Ethernet-based real-time networks in other domains, such as automatic control systems or substation automation systems, require higher synchronization accuracies. In these systems, accuracies in the microsecond range are often desired, and the number of

nodes can be in the range of hundreds. To meet these needs, PTP was designed and specified in IEEE 1588 [4]. The PTP is based on a straightforward master–slave synchronization principle. The PTP first establishes a master–slave hierarchy for clocks in a PTP system, and the grandmaster clock (GMC) at the top of the hierarchy determines the reference time for the entire system. Synchronization between clocks is achieved by exchanging PTP timing messages through the hierarchy, with the slaves using the timing information to adjust their clocks to the time of their master in the hierarchy. The best master clock (BMC) algorithm is used to establish the master–slave hierarchy for PTP systems. The BMC algorithm divides the clocks in the PTP system into master clocks (MCs) and slave clocks (SCs). SCs synchronize their local clocks with the time of their MCs. However, the BMC algorithm does not provide a fast recovery mechanism in case of master failures. The failure of a master requires the BMC algorithm to re-elect a new master and re-establish the hierarchy. This rate of recovery is dependent on the interval of the Announce message and the topology of the network [5]. This drawback causes the loss of reference clock synchronization from the GMC as well as clock drift for the clocks during the re-election of the new master, thus decreasing the synchronization performance of the network.

Several approaches have been proposed to recover the failure occurrence in PTP systems. These approaches can be based on fault tolerance or fault recovery. Methods based on fault tolerance mainly focus on providing a property that enables the standard PTP to continue operating properly if failure occurs. Thus, several approaches have been proposed to provide a fault-tolerance mechanism with the standard PTP [6–10]. The fault tolerance-based category is out of the scope of the comparison and analysis in this paper, due to the fact that several features of the PTP protocol have been excluded with these methods (i.e., BMC algorithm, master–slave synchronization hierarchy and cyclic path pruning) and excluding the PTP nodes from the network also; therefore, the synchronization accuracy of the system will be deteriorated [11,12]. Therefore, we primarily focus on the fault recovery-based category that includes all the PTP features and nodes. Fault recovery-based methods mainly focus on detecting and recovering the failure in the synchronized networks in the case of link failure or node failure. Furthermore, fault recovery is used to manage network devices and traffic, and it provides reliable communication without alternative communication paths. Considering this, the rapid spanning tree protocol (RSTP) [13] with the standard PTP provides a failover mechanism in time-synchronization systems [14,15]. As was previously stated, the standard PTP can work with various protocols, such as RSTP, which constructs the synchronization tree instead of PTP and blocks the redundant paths in order to construct a loop-free synchronization topology. Moreover, RSTP detects the failure occurrence in the network after a specific duration, ranging between hundreds of milliseconds and a few seconds [14], and subsequently, RSTP recovers the failure in the network and starts the synchronization mechanism through the standard PTP. No studies have yet reported on fault recovery in PTP systems.

In this paper, we propose a novel fault recovery-based approach, called timing fault recovery (TFR), to solve the fault occurrence issue in PTP systems. TFR detects and recovers the fault occurrence in the network with a short duration using a synchronization message detection mechanism. Subsequently, TFR has a recovery mechanism to rebalance the synchronization in the network. Accordingly, TFR significantly reduces the local clock drifting of the nodes in PTP networks. Additionally, it increases clock constancy and network reliability and availability. The fault recovery feature of the TFR approach enables the standard PTP to be suitable for all types of network topologies with mission-critical conditions.

The rest of the paper is organized as follows. Section 2 briefly introduces the standard PTP. Section 3 describes the proposed TFR approach. In Section 4, the performance of TFR is analyzed and compared to that of the standard PTP with RSTP. Section 5 describes the simulation results to validate the analysis that is described in Section 4. Finally, we provide our conclusions in Section 6.

2. The Standard PTP

2.1. PTP Operation

The standard PTP provides a synchronization mechanism for the clocks of participating nodes in a system to a high degree of accuracy and precision. The PTP defines several kinds of clocks, as follows:

- Ordinary Clock (OC): a clock that has one PTP port in a single domain and that maintains the timescale used in the domain. It may serve as a source of time (i.e., the GMC) or it may synchronize to another clock (i.e., an SC).
- Boundary Clock (BC): a clock that has multiple PTP ports in a single domain or multiple domains and that maintains the timescale used in the domain. It may serve as the source of time (i.e., an MC) and it may synchronize to another clock (i.e., an SC).
- Transparent Clock (TC): a clock that measures and calculates the elapsing time (called the residence time) for PTP event messages, including measurement of the processing and queuing delays. Furthermore, TC measures and calculates the link-propagation delay between similarly equipped ports at the end of the communication path.

The standard PTP is based on a master/slave principle. There are two phases in the normal execution of PTP: (1) establishing a master–slave hierarchy and (2) synchronizing the clocks.

2.1.1. Establishing the Master–Slave Synchronization Hierarchy

The first step for clock synchronization in the PTP system is establishing the master–slave synchronization hierarchy, which defines the clock synchronization flow from the highest accurate clock in the network (GMC) toward the end devices. The synchronization hierarchy is represented as a synchronization tree structure that only includes the ports of the OCs and BCs, as shown in Figure 1.

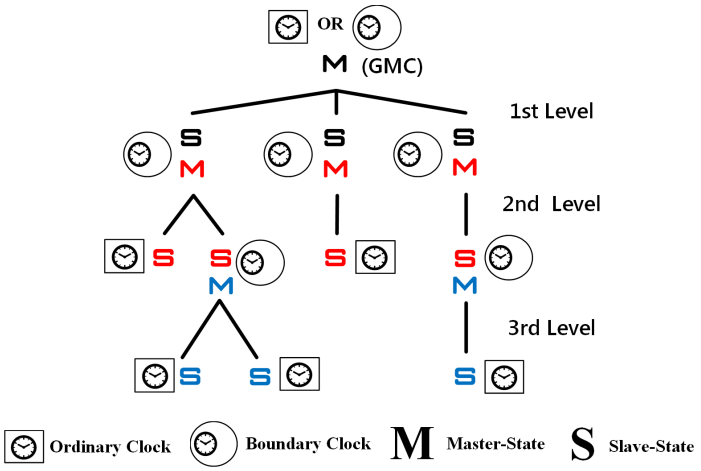

Figure 1. Example of the PTP master–slave synchronization hierarchy. PTP: Precision Time Protocol; GMC: grandmaster clock.

Specifically, GMC represents the root of the synchronization tree. In fact, the intermediate nodes of the tree are composed of BCs; thus, each port of these BCs is considered to be an independent clock (MC or SC) in the synchronization hierarchy. More importantly, the leaves of each level of the synchronization tree are represented as OCs, and they are considered to be SCs (Figure 1). However, TCs and non-PTP devices are not considered in the master–slave synchronization hierarchy, and they are represented as a connection point in the synchronization tree.

The establishment of the master–slave synchronization hierarchy can be accomplished by using the BMC algorithm; this algorithm determines the BMC in the network segment and elects the GMC depending on the clock accuracy of the nodes to represent the root of the synchronization

hierarchy. Subsequently, the algorithm determines the MCs for the remaining levels of the hierarchy, which is considered to be the source of the clock synchronization at each level. Consequently, the BMC algorithm determines each port state for all nodes in the PTP system as MCs and SCs, as shown in Figure 2. The structure of the master–slave synchronization hierarchy differs from the physical topology structure, which is converted by the BCM algorithm. Because the remaining ports of the intermediate nodes (BCs) are considered to be loop-free ports (passive-state ports) that will be determined by BMC algorithm, these ports prevent the loop inside the network.

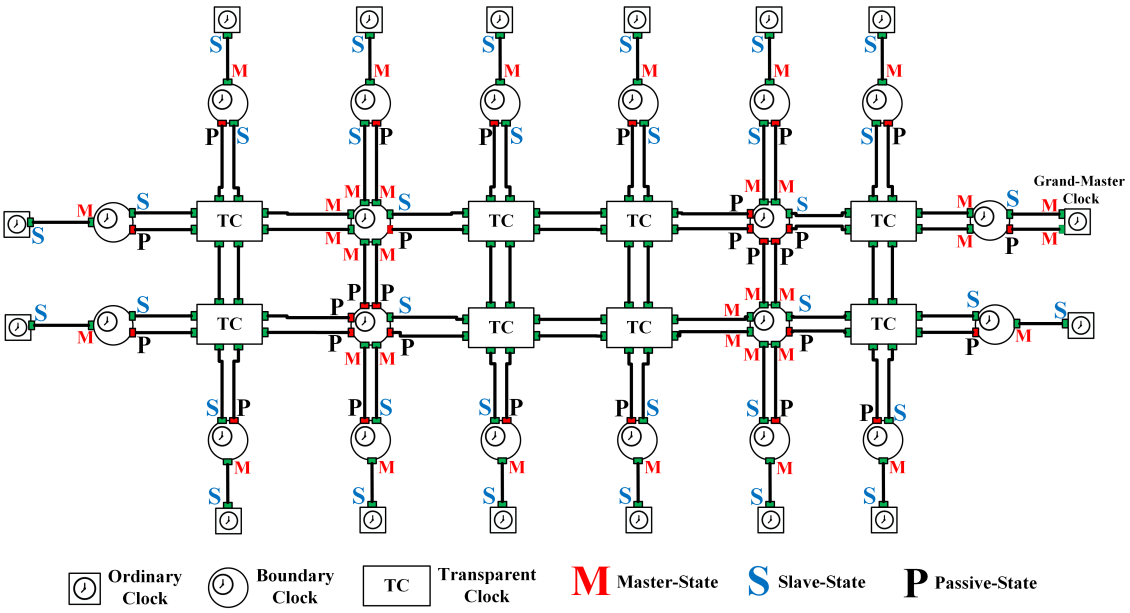

Figure 2. A sample PTP network.

Afterward, the timing flow is forwarded from the GMC toward the slave ports in the first level of the synchronization hierarchy. Subsequently, the MC ports in each level of the hierarchy distribute the time toward the slave ports until it reaches the leaf ports in the synchronization tree.

2.1.2. Timing-Exchange Mechanism

The second step of the clock synchronization in the PTP system is a timing exchange between the MC and the slave ports using event and general messages. In the standard PTP, the timing-exchange mechanism is called a delay request–response mechanism. This mechanism identifies the path of the time information exchange by using Sync, Follow_Up, Delay_Req and Delay-Resp messages to measure and calculate the clock drifting between the master and slave ports, as shown in Figure 3.

The synchronization mechanism begins by transmitting a Sync message from the MC toward the SC; consequently, this message will be time-stamped at the instant of the message departure and arrival at the MC physical layer (t_1) and the SC physical layer (t_2), respectively. After that, the MC transmits a Follow_Up message towards SC that contains the time-stamp value of the Sync message (t_1). Therefore, the SC responds with a Delay_Req message; thus, this message will be time-stamped at the instant of the message departure and arrival at the SC physical layer (t_3) and the MC physical layer (t_4), respectively. Finally, the MC forwards a Delay_Resp message towards the SC that contains the time-stamp information of the Delay_Req. Hence, the SC becomes aware of all the required time stamps (t_1, t_2, t_3, t_4) that are used to correct its local clock.

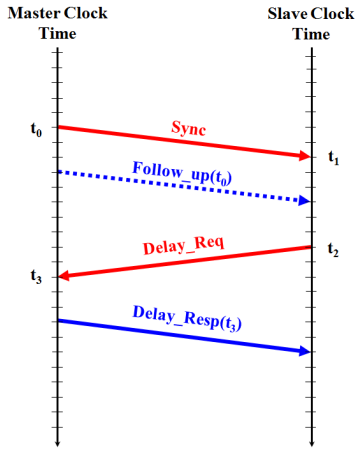

Figure 3. Delay request–response mechanism of the PTP timing exchange protocol.

2.2. PTP Issues

The synchronization rule of the standard PTP is that the BC node prevents the loop of the messages inside the network by converting several ports of the node into passive-state ports based on the BMC algorithm, as shown in Figure 2. In this case, the node receives the timing information through a single slave port and it distributes the local clock timing information through multiple master ports toward the remaining networks. The main drawback of the standard PTP operation is that it does not provide a protection mechanism for the occurrence of link failure and/or MC failure. Consequently, the slave ports that have been separated from the synchronized network start listening for an Announce message from the MC for a specified duration called an Announce-Receipt-TimeOut; afterwards, the slave ports are converted into a master state port that represents the root of the synchronization hierarchy of the isolated network. Hence, the node distributes the local clock timing information toward the remaining connected networks.

The timing accuracy of the isolated node's local clock is deteriorated by the link failure and the master failure occurrences in the network. Specifically, the fault occurrence causes the clock time drifting of the isolated nodes, which affects the synchronization of the network and reduces the reliability and the availability of PTP systems. This problem degrades the network performance and it may cause network delay. For this reason, we propose a TFR approach to solve the fault occurrence issue in PTP systems.

3. Proposed TFR Approach

This section describes the concepts, operations and algorithm of the proposed TFR approach.

3.1. Concepts

There are several new concepts defined in the proposed TFR approach, as follows:

- Link-Alive timer: a timer for counting the number of Sync-Intervals that have to pass without receiving Sync messages from the MC. Specifically, the timeout duration is equal to 3 × Sync-Interval.
- Announce timer: a timer for counting the number of Announce-Intervals that have to pass without receiving Announce messages from the MC. Specifically, the timeout duration is equal to 3 × Announce-Interval.
- Link-Alive message: a message that has been used to recover a failure in the network by using specific information inside the message. This message is used to identify a failure in the network of an SC.

- Sync-Request message: a message that has been used to acknowledge the recovery mechanism by the slave nodes after detecting a failure in the network using specific information inside this message.

3.2. Operations

The TFR functions of detecting and recovering a fault occurrence in the network are implemented in the MC and SC in PTP systems. The fundamental mechanism is shown in Figure 4.

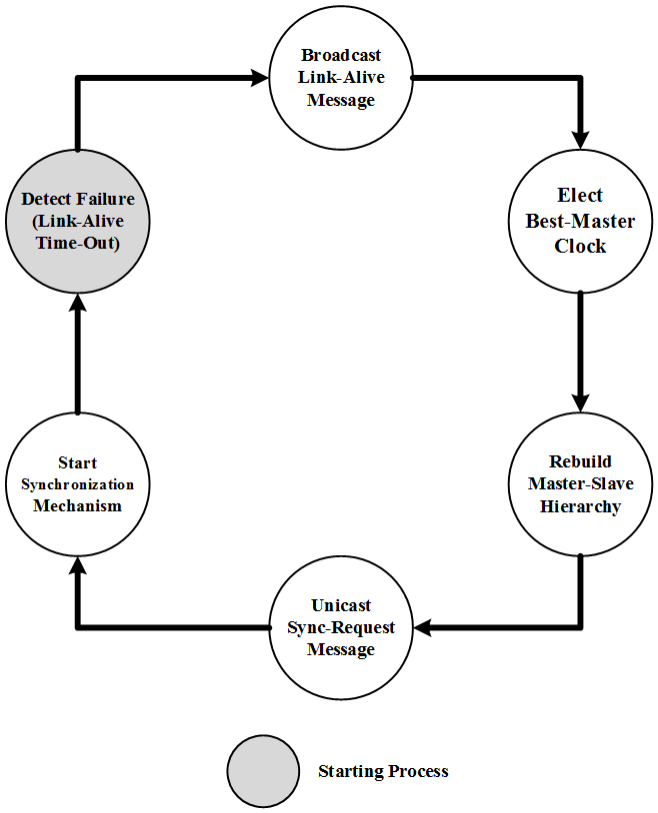

Figure 4. The TFR process. TFR: timing fault recovery.

3.2.1. Detecting the Fault Occurrence

In the case of a fault occurrence in the network, such as a link failure between the MC and the SCs, the SCs detect the failure by using the Link-Alive timer. Each OC and BC in the network starts this specific timer to identify the failure in the network segment. After this duration, the slave ports identify the fault occurrence and start activating all the ports of the node, including the passive state ports, and they broadcast a Link-Alive message through all the ports toward the MCs in the same network segment. The Link-Alive message targets the MCs in the network exclusively as the sources of the synchronization in the PTP networks. Consequently, the Link-Alive message includes information to identify the slave nodes that receive this specific message and request a new synchronization mechanism from the MCs. The fields of the Link-Alive message are addressed as follows:

- Link-Alive flag: identifies the port state of the transmitting node as either a slave port (flag = 1) or a master port (flag = 0).
- Clock-Request flag: identifies the port of the transmitting node requesting clock synchronization (flag = 1).
- PortID: identifies the port identifier of the transmitting node (slave port).

In this case, the slave ports identify the Link-Alive flag (flag = 1) and the Clock-Request flag (flag = 1), and they use their specific port identifiers. This specific message, as shown in Figure 5, uses the PTP message header [5] and the Signaling message type.

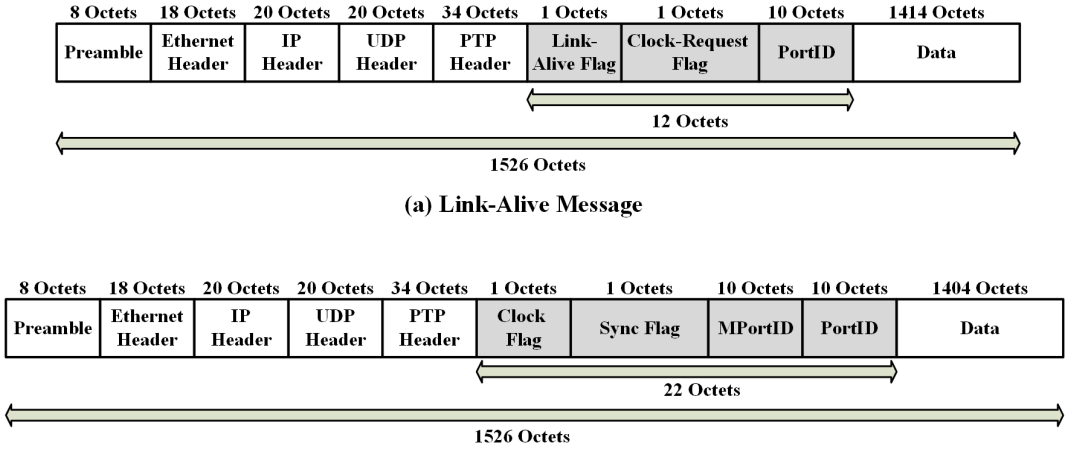

Figure 5. The structures of the Link-Alive and Sync-Request messages. IP: Internet Protocol; UDP: User Datagram Protocol.

3.2.2. Re-establishing the Synchronization Hierarchy

The second step in the TFR approach is to identify the live MCs to the slave nodes in order to elect the best master between them and to reestablish the master–slave synchronization hierarchy, depending on the MC and the active ports of the slave nodes. As a result of the previous step, each MC in the network segment will broadcast an Announce message toward the SCs after receiving the Link-Alive message. This Announce message contains the local clock timing of the MC; depending on this particular message, the SC identifies the accuracy of each MC in the network. Subsequently, the slave ports receive the Announce message and start by performing a Best-Master election operation that selects the best MC between all the clocks that transmitted the Announce message toward the slave ports.

Specifically, this operation performs the standard BMC algorithm to identify the best MC in the network segment based on the clock accuracy of the MC. After electing the best MC, the nodes perform the synchronization hierarchy rebalancing operation by establishing the synchronization tree based on the active ports of the slave nodes that exclusively received the Announce message. Afterwards, the SCs perform the ports pruning operation to prevent a loop inside the PTP network by changing the state of these ports to a passive state. Furthermore, this process excludes the main ports of the synchronization from the ports pruning operation by maintaining the state of these ports as active ports.

Eventually, the result of this step is that the separated slave ports in the network perfectly recover the fault occurrence in the network by using alternative synchronization paths without affecting network traffic performance, and by using multiple paths without a loop-free structure. Furthermore, the SCs eliminate the faulty paths from the synchronization flow between the MC and the slave ports, and they elect the BMC in the network to fulfill accurate synchronization for the SCs in PTP networks.

3.2.3. Fault Recovery Acknowledgement

The last step of the TFR approach is to identify the communication paths between the slave ports and the MC port, and to start the standard synchronization mechanism between the clocks because the MC still does not precisely identify the communication paths with the slave ports. Therefore,

the slave ports unicast the synchronization Acknowledgement message (Sync-Request message), which identifies the Active slave ports that have been used in the synchronization hierarchy. Thus, the slave ports exclusively target the best MC ports that have been used in the synchronization hierarchy. The content of the Sync-Request message includes specific information to notify the MC port that receives this particular message and the slave nodes that are requesting to start the synchronization mechanism immediately.

The fields of the Sync-Request message are addressed as follows:

- Clock flag: identifies the port state of the transmitting node as either a slave port (flag = 1) or a master port (flag = 0).
- Sync flag: identifies the port of the transmitting node that is requesting to start the clock synchronization mechanism immediately (flag = 1).
- PortID: identifies the port identifier of the transmitting node (slave port).
- MPortID: identifies the port identifier of the receipt node (master port).

In this case, the slave ports will identify the Clock flag as (flag = 1) and the Sync flag as (flag = 1), and they use the PortID as their specific port identifier and the MPortID as the specific port identifier of the MC in the PTP system. This specific message uses the PTP message header and signaling message type, as shown in Figure 5b.

3.2.4. Repairing the Fault Recovery Mechanism Failure

When the fault recovery is complete, the nodes in the PTP network start the standard PTP synchronization mechanism between the MC and the SCs. In the case of a fault that occurs in the SC network after the fault recovery, the SC enables the fault recovery mechanism repetitively until the failure in the network is repaired.

The extra feature of the TFR fault recovery mechanism is that it can repair the recovery mechanism if the failure occurs during the fault recovery operation; several possible scenarios are covered in the TFR, as follows:

- Fault occurrence after broadcasting a Link-Alive message: If a failure occurs after broadcasting a Link-Alive message, the SC starts an Announce timer. If the Announce timer times out, the slave port broadcasts the Link-Alive message once again toward the master and starts waiting for the Announce message from the MC.
- Fault occurrence after broadcasting a Link-Alive message for the second time: If a failure occurs after broadcasting a Link-Alive message for the second time, the SC starts an Announce timer. Consequently, the slave port starts waiting for the Announce message from the MC; if the SC does not receive an Announce message after this duration, it converts the slave port state to an MC state and distributes its local clock in the same network segment.
- Fault occurrence after unicasting a Sync-Request message: If a failure occurs after unicasting the Sync-Request message, the SC starts a Link-Alive timer. If the Link-Alive timer times out, the slave port starts the fault recovery mechanism once again to detect and recover the failure occurrence in the network.

Here, two extreme detection scenarios can be considered. The first is the worst-case detection scenario for a fault recovery occurrence after unicasting the Sync-Request message. In this case, the timing cost is ($6 \times$ Announce-Interval $+ 6 \times$ Sync-Interval); specifically, the timing costs are ($3 \times$ Announce-Interval) for an Announce timer for the first broadcast duration timer and ($3 \times$ Announce-Interval) for an Announce timer of the second broadcast duration timer that was noted in the first two points. Additionally, the timing cost is ($3 \times$ Sync-Interval) for a Link-Alive timer multiplied by 2 for the main duration of detecting the failure and the detection duration after unicasting the Sync-Request message that was noted in the last point. In the second case, the best-case detection scenario is that the node detects the failure after a Link-Alive timer duration and receives the standard synchronization messages before the Announce timer times out. In this case, the timing cost is ($3 \times$ Sync-Interval), which is only caused by a Link-Alive timer.

3.3. State Machine and Algorithm

The state machine of the TFR algorithm is shown in Figure 6. The corresponding pseudo-code of the TFR algorithm is shown in Algorithm 1.

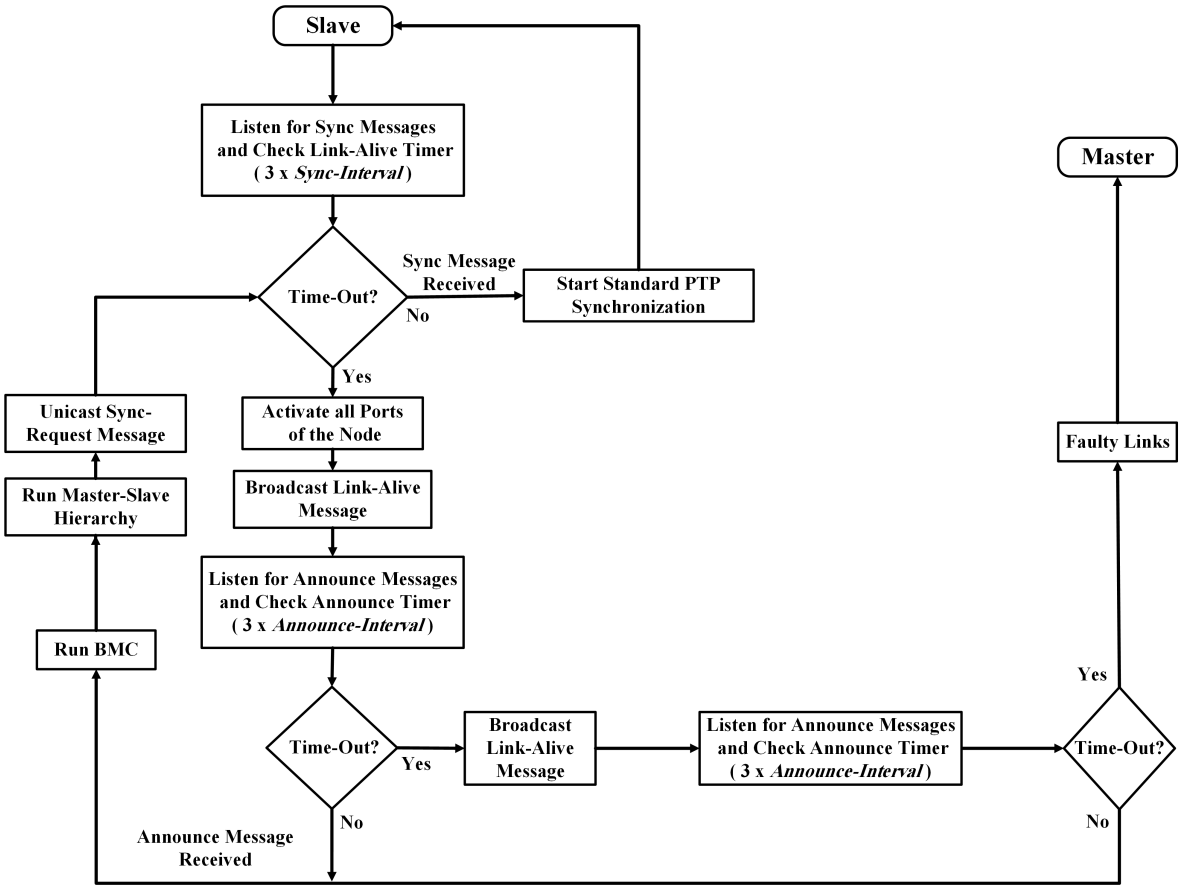

Figure 6. The TFR state machine. BMC: best master clock.

4. Performance Analysis

This section presents an analysis and evaluation of the fault recovery performance of the TFR approach in comparison to the standard PTP. We analyzed and evaluated the delay of detecting and recovering the failure with BC nodes in the sample network. TC nodes are considered to be the connection point; as previously mentionned, they are not considered in the synchronization mechanism.

The sample PTP network shown in Figure 2 is considered in order to analyze the performance of the proposed TFR approach. It is assumed that the link distance between each node is 50 m and the node processing rate is 100 Mbps. Furthermore, the Sync message rate is assumed to be 128 messages-per-second and the Announce message rate is 8 messages-per-second, which was defined in the ITU-T G.8275.2 standard profile [16]. Therefore, the Sync-Interval will be determined as 1/128 that is equal to 0.0078125 s and the Announce-Interval will be determined as 1/8 that is equal to 0.125 s. Finally, we consider that the Hello packet interval duration of RSTP is 2 s, which was defined in the RSTP standard as a default value [13]. Additionally, the port identity size is 10 octets and the MC dataset size is 14 octets, which was defined in the standard [5].

We considered multiple scenarios in which link failures occur between the MCs and SCs in the network. In the first scenario, a link failure only occurs in one node in the network; afterwards, we increased the number of link failures in the network until all the nodes in the network experience

Algorithm 1: TFR Algorithm

```
Input  : Synchronization Messages (Announce, Sync)
Output : TFR Messages (Link-Alive, Sync-Request)
Link-Alive-timer = 3;
Announce-timer = 3;
Link-Alive-TimeOut = False;
Announce-TimeOut1 = False;
Announce-TimeOut2 = False;
for (i = 1; i <= Link-Alive-timer; i++) do
    if (SyncorAnnounce( then
        Start Standard PTP Synchronization;
    end
    else if (i = 3) then
        Link-Alive-TimeOut = True;
    end
end
if (Link-Alive-TimeOut == True) then
    Activate All Ports;
    Broadcast Link-Alive;
    for (j = 1; j <= Announce-timer; j++) do
        if (Announce) then
            Run Best-Master-Clock;
            Run Master-Slave-Hierarchy;
            Unicast Sync-Request;
        end
        else if (j = 3) then
            Announce1-TimeOut = True;
        end
    end
end
if (Announce-TimeOut1 == True) then
    Broadcast Link-Alive;
    for (k = 1; k <= Announce-timer; k++) do
        if (Announce) then
            Run Best-Master-Clock;
            Run Master-Slave-Hierarchy;
            Unicast Sync-Request;
        end
        else if (k = 3) then
            Announce-TimeOut2 = True;
        end
    end
end
if (Announce-TimeOut2 == True) then
    Port State = Master;
end
```

a link failure, as shown in Figure 7. In each scenario, we determined the total delay duration for detecting and recovering the failure between the BC slave ports and the MC ports in the same communication path.

We did not consider the case of an all-links failure for a certain node in the sample network in the performance analysis in order for this specific node to not be isolated from the network. Afterwards, this specific node will be considered as the root of the synchronization hierarchy for the isolated network segment that will cause local clock drifting in comparison to the GMC. Eventually, the TFR approach will not solve this issue unless there is a live link between the MC ports and the SC ports in the same network segment.

4.1. PTP with RSTP

The standard PTP with RSTP recovery mechanism starts by detecting the Bridge Max Age time-out duration that detects the reception of the periodical Hello messages from the other nodes [13]. Considering that, in the case of link failure occurrence in the network, the failure node broadcasts a Bridge Protocol Data Units (BPDU) frame through all the ports after the above-mentioned duration, this specific frame notifies the remaining nodes that the topology has been changed. Subsequently, the root nodes that received this specific frame will broadcast a BPDU frame, which contains a Proposal Agreement for re-establishing the RTSP tree. Eventually, the nodes that receive the Proposal Agreement notification will replay with a BPDU frame, which contains an Agreement Acknowledgment towards the root of that level of the tree.

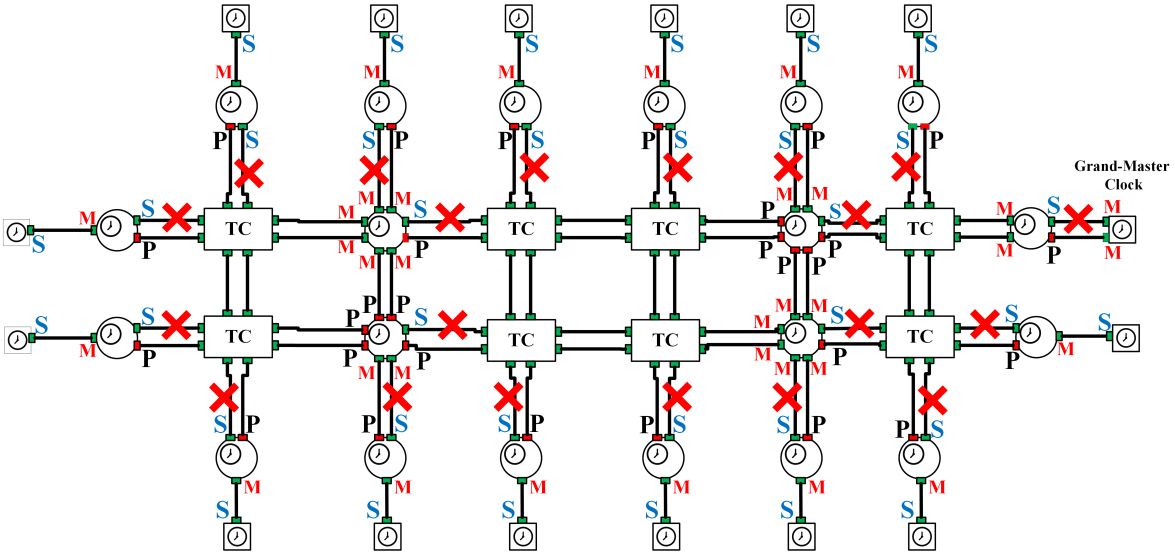

Figure 7. A sample PTP network with possible link failures.

Hence, the total delay of the standard PTP with RSTP ($D_{PTP-RSTP}$) for detecting and recovering the failure occurrence in the network can be determined as follows:

$$D_{PTP-RSTP} = 2 \times (D_{TR} + D_{PG}) + BD_{MAX}, \quad (1)$$

where

$$BD_{MAX} = 2 \times H_{PCK}. \quad (2)$$

BD_{MAX} refers to the Bridge max age of detecting the failure and H_{PCK} refers to the duration of non-receiving a Hello packet from the connected nodes, which indicates the link failure occurrence in the same network segment. Usually, the H_{PCK} detection duration ranges between hundreds of milliseconds to a few seconds [14]. In addition, D_{TR} and D_{PG} refer to transmission delay and propagation delay, respectively. Since these delays consume a negligible amount of time, the total delay of PTP with RSTP can be approximated as follows:

$$D_{PTP-RSTP} \cong 2 \times H_{PCK}. \quad (3)$$

For the sample PTP network shown in Figure 2, the delay of PTP with RSTP using typical $H_{PCK} = 2$ s is:

$$\mathrm{D}_{PTP-RSTP} \cong 2 \times 2 = 4\,(\mathrm{sec}). \quad (4)$$

4.2. TFR

The TFR failure recovery time, D_{TFR} can be approximated as follows:

$$D_{TFR} = D_{DT} + D_{BMC}, \quad (5)$$

where D_{DT} refers to the detection and recovery delay and D_{BMC} refers to the BMC algorithm reestablishment delay.

The BMC reestablishment delay, D_{BMC}, refers to the amount of time required to reestablish the BMC algorithm in the PTP nodes when a failure occurs in the network; therefore, D_{BMC} can be determined as follows:

$$D_{BMC} = N_{Recovery} \times \sum_{j=1}^{C_{BMC}} (O_{CK} + O_{MCE}), \quad (6)$$

where j represents the number of clocks that will be involved in the recovery operation, $N_{Recovery}$ refers to the total number of clocks that will be involved in the recovery operations and C_{BMC} refers to the number of clocks that are involved in BMC reestablishment. O_{CK} refers to the port identification operation and O_{MCE} refers to the MCs compassion and election operation. Specifically, O_{CK} refers to the amount of time required to identify all the ports in the network segment by each node that will be involved in the recovery operation; then, O_{CK} is determine as follows:

$$O_{CK} = N_{Ports} \times \frac{I_{Port}}{PC_R}, \tag{7}$$

where I_{Port} refers to the ports state identifier and PC_R refers to node processing rate. Furthermore, O_{MCE} refers to the amount of time required to compare all the MCs in the network segment and to elect the best master from among them. Therefore, it can be determined as follows:

$$O_{MCE} = N_{MC} \times \frac{2 \times MC_{DS}}{PC_R}, \tag{8}$$

where N_{MC} refers to the total number of MCs in the network segment and MC_{DS} refers to the MC dataset that identifies the timing accuracy of each node.

The TFR fault detection delay refers to the time duration that the node consumes when failure is detected, which can be determined as follows:

For the best-case detection scenario:

$$D_{DT(TFR-Best)} = 3 \times S_{INT}. \tag{9}$$

For the worst-case detection scenario:

$$D_{DT(TFR-Worst)} = 2 \times (3 \times S_{INT}) + 2 \times (3 \times A_{INT}), \tag{10}$$

where S_{INT} refers to the Sync-Interval and A_{INT} refers to the Announce-Interval.

Considering these delays, the total delay of TFR approach (D_{TFR}) with the best-case detection scenario is determined as follows:

$$D_{TFR(Best-Case)} = 3 \times S_{INT} + N_{Recovery} \times \sum_{j=1}^{C_{BMC}} \left(N_{Ports} \times \frac{I_{Port}}{PC_R} + N_{MC} \times \frac{2 \times MC_{DS}}{PC_R} \right). \tag{11}$$

The TFR for the worst-case detection scenario, D_{TFR} is determined as follows:

$$\begin{aligned} D_{TFR(Worst-Case)} = {} & 2 \times (3 \times S_{INT}) + 2 \times (3 \times A_{INT}) \\ & + N_{Recovery} \times \sum_{j=1}^{C_{BMC}} \left(N_{Ports} \times \frac{I_{Port}}{PC_R} + N_{MC} \times \frac{2 \times MC_{DS}}{PC_R} \right). \end{aligned} \tag{12}$$

For the sample PTP network shown in Figure 2, the TFR delays with 20 link failures are:

$$\begin{aligned} D_{TFR(Best-Case,\,20\,\text{links failure})} &= 3 \times 0.0078 + 20 \times \sum_{i=1}^{40} \left(22 \times \frac{10 \times 8}{10^8} + 2 \times \frac{2 \times 14 \times 8}{10^8} \right) \\ &= 0.0411\,(\text{s}) \end{aligned} \tag{13}$$

$$\begin{aligned} D_{TFR(Worst-Case,\,20\,\text{links failure})} &= 6 \times 0.0078 + 6 \times 0.125 + 20 \times \sum_{i=1}^{40} \left(22 \times \frac{10 \times 8}{10^8} + 2 \times \frac{2 \times 14 \times 8}{10^8} \right) \\ &= 0.8145\,(\text{s}) \end{aligned} \tag{14}$$

Table 1 shows the total delay duration for detecting and recovering the failure occurrence for the sample network. The results show that the delay duration in the TFR best-case detection scenario ranges between 0.0234 and 0.0411 s, the delay duration for the TFR worst-case detection scenario ranges between 0.7969 and 0.8145 s, and the delay duration for the PTP under RSTP is 4 s.

Table 1. The delay values (seconds) of PTP with RSTP, and the TFR approach with the best-case and worst-case detection scenarios. PTP: precision time protocol; RSTP: Rapid Spanning Tree Protocol; TFR: Timing Fault Recovery.

Number of Failed Links	PTP with RSTP	TFR Best-Case Scenario	TFR Worst-Case Scenario
1	4	0.0234	0.7969
2	4	0.0236	0.7970
3	4	0.0238	0.7972
4	4	0.0241	0.7975
5	4	0.0245	0.7979
6	4	0.0250	0.7984
7	4	0.0256	0.7990
8	4	0.0262	0.7997
9	4	0.0270	0.8004
10	4	0.0278	0.8012
11	4	0.0287	0.8022
12	4	0.0297	0.8032
13	4	0.0309	0.8043
14	4	0.0320	0.8055
15	4	0.0322	0.8068
16	4	0.0347	0.8081
17	4	0.0361	0.8096
18	4	0.0377	0.8111
19	4	0.0393	0.8128
20	4	0.0411	0.8145

5. Simulations

To validate the analytical performance, various simulations were conducted using the OMNeT++ v4.6 simulation tool (Version 4.6, OpenSim Ltd., Stanford, CA, USA, 2014) [17].

5.1. Simulation Description

For the sample PTP network with 20 BCs, 16 OCs, eight TCs and one GMC, the OC was regarded as terminal nodes, the BC and TC were considered to be intermediates nodes, and the GMC was considered to be the source of the clock synchronization, as shown in Figure 2. We considered multiple cases in which link failure occurred between the MC ports and SC ports in the network. First, we examined a scenario in which a link failure occurred in only one node in the network; afterwards, we increased the number of link failures up to 20, as shown in Figure 7.

5.2. Simulation Results

Figure 8 shows the total delay duration for the detection and recovery of the failure occurrence for the proposed TFR approach and the standard PTP under RSTP. The simulation results show that the total delay duration was reduced by about 100% in comparison to the standard PTP because the standard PTP does not include a fault detection and recovery mechanism. However, the total delay duration was reduced by about 79–99%, in the case of fault occurrence between the MC and the SC.

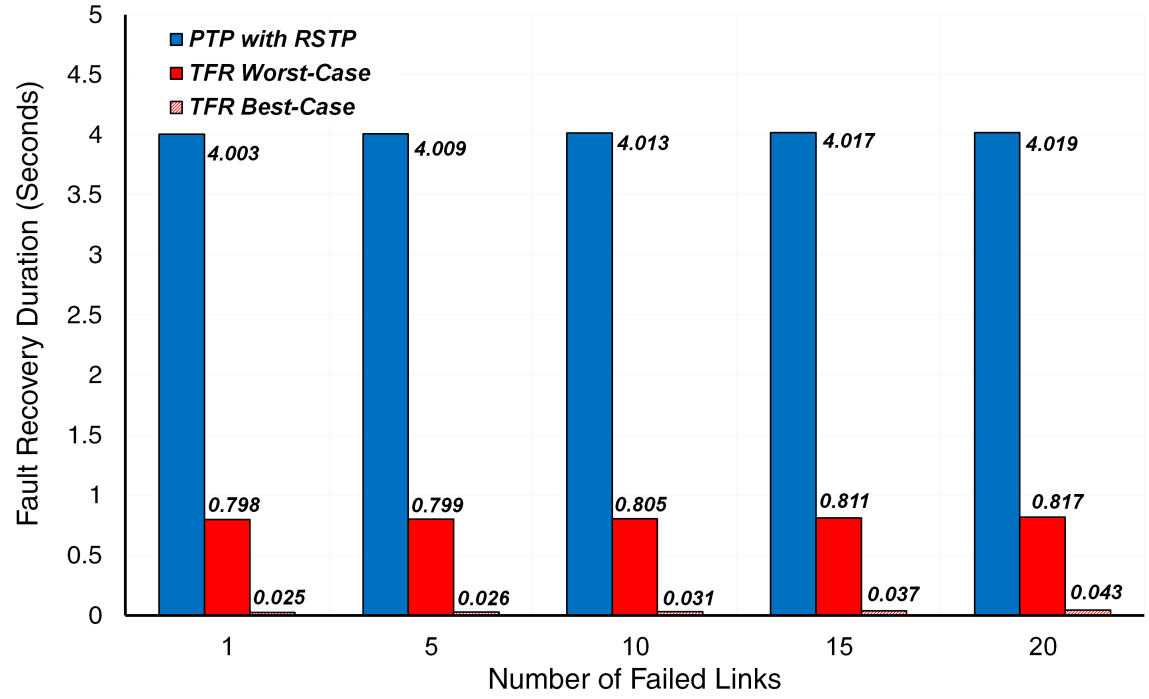

Figure 8. A simulation comparison of fault recovery durations in the TFR and PTP under RSTP approaches. TFR: Timing Fault Recovery; RSTP: Rapid Spanning Tree Protocol.

Figure 9 shows the local clock timing comparison of the SC with respect to MC in the same communication path for the TFR approach, the standard PTP, and the standard PTP protocol with RSTP.

The simulation results show that the TFR approach significantly reduced the local clock drifting of the SC in comparison to the standard PTP by about 90% and the standard PTP with RSTP by about 75%, in the case of fault occurrence between the MC and the SC.

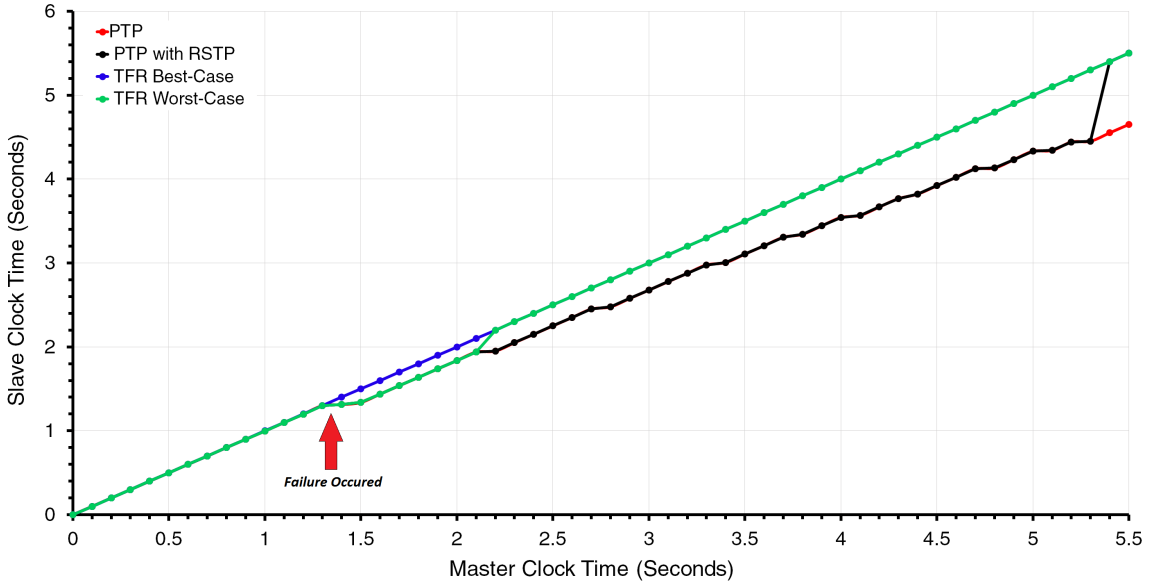

Figure 9. Slave clock timing values with TFR and PTP with RSTP with respect to MC timing. MC: Master Clock.

5.3. Discussion

The simulation results demonstrate that the fault recovery performance of the TFR approach significantly reduced the clock drifting of the local clock in comparison to the standard PTP and the standard PTP under RSTP. This is due to the fact that the TFR approach detects the failure with a short duration, and it starts to recover it by using a handshake mechanism between the master and slave nodes. In contrast, the standard PTP does not include any fault recovery mechanism to identify the failure in the network, so it does not perform a recovery process to retrieve it.

As a final point, the reason for the variation in the detection and recovery delay of the simulation results in comparison to the analytical results is due to the variation in the total network delay, specifically the queueing and transmission delays, and the latency of reestablishing the BMC and the synchronization hierarchy after the failure occurrence in the simulated network. More importantly, the small variation that has been stated in the results is owing to the fact that the sample failure scenario that has been used in the analysis and simulation validation includes the available links failure without isolating the GMC or synchronization reference from the network. Additionally, we did not consider a node failure scenario in the sample network, due to the fact that the node failure in the network leads to isolating the GMC and synchronization reference.

6. Conclusions

In this paper, we proposed a novel approach, known as TFR, for detecting and recovering a fault occurrence in PTP systems. Unlike the standard PTP, which does not contain a fault recovery mechanism, our TFR approach detects and recovers a failure occurrence in the network with a short duration of failure detection and recovery delay. Moreover, the TFR approach detects and recovers failure occurrences more efficiently than the standard PTP under RSTP, which detects and retrieves a failure occurrence within a few seconds. Therefore, the proposed TFR approach significantly reduces the local clock drifting of PTP nodes. The analytical and simulation results showed that, for our sample network, the TFR approach reduced the total delay detection and recovery duration by 100% in comparison to the standard PTP and by 79–99% in comparison to the standard PTP with RSTP. Furthermore, the clock drifting simulation results illustrated that the TFR approach reduced the local clock drifting of the slave nodes by 90% in comparison to the standard PTP and by 75% in comparison to the PTP with RSTP. Hence, TFR improves the reliability and the availability of the PTP network; moreover, TFR enhances the network performance and the synchronization accuracy of PTP clocks.

Our future work will involve developing a new BMC technique that has less time complexity for establishing the master–slave hierarchy, and we will apply it using a TFR approach in PTP systems.

Acknowledgments: This work (Grants No. C0453855) was supported by Business for Cooperative R&D between Industry, Academy, and Research Institute funded Korea Small and Medium Business Administration in 2016 and Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (grant number: 2017R1A2B4003964).

Author Contributions: Author Alfarooq Omar Alshaikhli conceived and developed the ideas behind the research. Alfarooq Omar Alshaikhli carried out the performance analysis and simulations, and wrote the paper under supervision of Jong Myung Rhee. Jong Myung Rhee supervised the research and finalized the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mills, D.L. Internet time synchronization: The network time protocol. *IEEE Trans. Commun.* **1991**, *39*, 1482–1493.
2. Subrahmanyan, R. Implementation considerations for IEEE 1588v2 applications in telecommunications. In Proceedings of the 2007 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication, Vienna, Austria, 1–3 October 2007; pp. 148–154.
3. Mills, D.; Martin, J.; Burbank, J.; Kasch, W. *Network Time Protocol Version 4: Protocol and Algorithms Specification*; Technical Report; Internet Engineering Task Force (IETF): Fremont, CA, USA, 2010.

4. *IEEE Standard for A Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*; IEEE Std 1588-2002; IEEE: Piscataway, NJ, USA, 2002; pp. i–144.
5. *IEEE Standard for A Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*; IEEE Std 1588-2008 (Revision of IEEE Std 1588-2002); IEEE: Piscataway, NJ, USA, 2008; pp. 1–300.
6. Gaderer, G.; Loschmidt, P.; Sauter, T. Improving fault tolerance in high-precision clock synchronization. *IEEE Trans. Ind. Inf.* **2010**, *6*, 206–215.
7. Kozakai, Y.; Kanda, M. Keeping clock accuracy on a master clock failure in substation network. In Proceedings of the 2010 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication (ISPCS), Portsmouth, NH, USA, 27 September–1 October 2010; pp. 25–29.
8. Murakami, T.; Horiuchi, Y. A master redundancy technique in IEEE 1588 synchronization with a link congestion estimation. In Proceedings of the 2010 IEEE International Symposium on Precision Clock Synchronization for Measurement Control and Communication (ISPCS), Portsmouth, NH, USA, 27 September–1 October 2010; pp. 30–35.
9. Puhm, A.; Mahmood, A.; Bigler, T.; Kerö, N. Synchronizing an IEEE 1588 slave clock over both paths of a redundant Ethernet system. In Proceedings of the 2016 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control, and Communication (ISPCS), Stockholm, Sweden, 4–9 September 2016; pp. 1–6.
10. Shpiner, A.; Revah, Y.; Mizrahi, T. Multi-path time protocols. In Proceedings of the 2013 IEEE International Symposium on Precision Clock Synchronization for Measurement Control and Communication (ISPCS), Lemgo, Germany, 22–27 September 2013; pp. 1–6.
11. Ferrari, P.; Flammini, A.; Rinaldi, S.; Prytz, G. High availability IEEE 1588 nodes over IEEE 802.1aq Shortest Path Bridging networks. In Proceedings of the 2013 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication (ISPCS), Lemgo, Germany, 22–27 September 2013; pp. 35–40.
12. TC57. *Communication Networks and Systems in Substations Part 90-4: Network Engineering Guidelines*; Technical Report; TR 61850-90-4; IEC: Geneva, Switzerland, 2013.
13. *IEEE Standard for Local and Metropolitan Area Networks: Media Access Control (MAC) Bridges*; IEEE Std 802.1D-2004 (Revision of IEEE Std 802.1D-1998); IEEE: Piscataway, NJ, USA, 2004; pp. 1–277.
14. Prytz, G. Redundancy in industrial Ethernet networks. In Proceedings of the 2006 IEEE International Workshop on Factory Communication Systems, Torino, Italy, 28–30 June 2006; pp. 380–385.
15. Meier, S.; Weibel, H. IEEE 1588 applied in the environment of high availability LANs. In Proceedings of the IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication (ISPCS), Vienna, Austria, 1–3 October 2007; pp. 100–104.
16. *Precision Time Protocol Telecom Profile for Time/Phase Synchronization with Partial Timing Support From the Network*; ITU-T G.8275.2 Recommendation; International Telecommunication Union/ITU Telcommunication Sector: Geneva, Switzerland, 2016; pp. 1–46.
17. OMNeT++ Discrete Event Simulator. Version 4.6. Available online: http://www.omnetpp.org/ (accessed on 10 October 2017).