Hierarchical Time-frequency Synchronization Mechanism for Time Sensitive Networking

Yunzhu Yu , Kaijie Wu*, Qimin Xu , Xiang Chen , and Cailian Chen

Department of Automation, Shanghai Jiao Tong University, Shanghai 200240, P. R. China
Key Laboratory of System Control and Information Processing, Ministry of Education of China
*Corresponding Author. Email:Kaijiewu@sjtu.edu.cn

***Abstract*—Modern Industrial Internet-of-Things (IIoT) requires reliable interaction and integration of data in the network. Thus, time sensitive networking (TSN) is a promising technology due to deterministic latency guarantee mechanisms for data transmission. However, the multiple mechanisms are based on the networkwide precise time synchronization. In this paper, to achieve precise and reliable clock synchronization of TSN, we propose a hierarchical timing-frequency synchronization mechanism. Specifically, the network is divided into two layers according to the network clock synchronization function. The top layer adopts tree-based synchronization to provide a global clock and an interface with external standard clock synchronization, and the underlying one adopts a distributed synchronization protocol to improve the reliability, which obtains the reference clock by multiple nodes to avoid the single point of failure. To improve synchronization efficiency, a time-frequency fusion synchronization mechanism is proposed, which comprehensively considers the synchronization time slot difference and the time-frequency coupling relationship to improve the synchronization speed under the same synchronization period. Simulation results show that the proposed synchronization mechanism has higher synchronization efficiency than the traditional methods, and improves the clock synchronization speed and accuracy significantly.**

***Index Terms*—Clock synchronization, Time sensitive networking, Precision time protocol (PTP), AS6802**

I. Introduction

With the introduction of Industry 4.0, smart factories have become a new stage of modern factory informatization development [1]. Smart factories realize the cluster interaction of different types of industrial data through the Industrial Internet-of-Things (IIoT) and promote the intelligent and networked transformation and upgrading of manufacturing through data fusion [2]. The precise interaction and fusion of data need to ensure that data is transmitted within a specific limited time. It requires the network to be deterministic and real-time, which are also the most important characteristics of industrial networks [3]. Although there are currently deterministic Ethernet protocols, such as Profinet [4], EtherCAT [5], Ethernet Powerlink [6], etc., they cannot achieve deterministic transmission under mixed data streams. So the IIoT based on these protocols cannot guarantee the accuracy and reliability of the data. To improve the real-time performance and reliability of Ethernet with mixed data, the IEEE 802.1 task group began the research on Time Sensitive Networking (TSN) [7].

TSN is considered as a universal communication tool under Industry 4.0 [8], extending the traditional Ethernet data link layer standard to ensure that data transmission has limited ultra-low latency and low jitter [9]. Clock synchronization provides a general and accurate time for all TSN devices, which is the basis of TSN. Without precise clock synchronization, TSN is not able to control the gate switch in the Gate Control Lists (GCL) at an accurate time during the flow schedule, affecting the deterministic delay of TSN. Therefore, TSN puts forward significant requirements for clock synchronization accuracy and reliability between devices.

Aiming at the problem of clock synchronization, effective methods have been proposed in previous studies. In terms of synchronization structure, the current mainstream methods mainly include tree-based synchronization and distributed synchronization [10].

The tree-based protocol needs to establish a hierarchical structure in the network, which includes a master node and other slave nodes. Typical representatives are the network time protocol (NTP) [11], the precise time protocol (PTP) [12] and IEEE 802.1AS generalized precision time protocol (gPTP) [13]. NTP implements Internet synchronization at the application layer, with low synchronization accuracy due to errors generated by the network protocol stack. PTP obtains time stamps at the location between the physical layer and the MAC layer, which improves synchronization accuracy. gPTP is developed based on PTP, which uses a two-step delay measurement. Although tree-based synchronization is simple to implement, errors will occur in each level of synchronization. As the network scale increases, errors will continue to accumulate, which seriously affects the synchronization performance.

Distributed synchronization is different from tree-based synchronization. The clock reference is not provided by a master node but determined by multiple nodes in the network. AS6802 [14] is a typical distributed synchronization protocol, which is mainly applied to aviation networks. Due to the distributed structure, the network avoids the hierarchical structure and reduces the cumulative errors [10]. Since the reference clock is not entirely dependent on one node, the reliability of synchronization is significantly improved compared to tree-based. However, due to the lack of nodes synchronized with the external clock, synchronization can only be performed

within the network, and local clock synchronization causes the internal clock to shift overall and cannot synchronize with Coordinated Universal Time (UTC).

The above synchronization protocols are essentially time synchronization. Besides time synchronization, clock synchronization also includes frequency synchronization [15]. In practical applications, the clock frequency is time-varying due to the effects of temperature [15], aging [16], etc., resulting in continuous errors in the clock after the long-term operation, so the frequency offset cannot be ignored for high precise [17]. Therefore, frequency synchronization is required to improve the accuracy of clock synchronization, especially in a harsh industrial environment.

Some existing works have investigated the synchronization of frequency offset. For example, for multi-hop wireless sensor networks, Wang et al. [18] studied a time synchronization scheme and a corresponding clock frequency estimation method, assuming that the noise is a Gaussian model. Fan et al. [19] realized simultaneous synchronization of time and frequency by using a parametric differential clock synchronization algorithm. Zhang et al. [20] proposed a new clock synchronization scheme based on maximum consistency for wireless sensor networks. However, the above methods are complicated and require frequent timing information exchange between nodes, resulting in longer calculation time and ultimately affecting the low-latency characteristics of TSN. Hence, the study of frequency synchronization without increasing the computational complexity is an important research topic.

Motivated by these, the main work of this paper is to propose a hierarchical time-frequency synchronization mechanism for TSN. The main contributions of this work are as follows.

- A hierarchical synchronization mechanism was established to improve synchronization reliability, in which the reference clock is calculated in a distributed multi-node collaborative decision mode to avoid the single point failure of the reference clock.
- To improve synchronization efficiency, a time-frequency fusion synchronization mechanism is proposed. Considering the synchronization time slot difference and the time-frequency coupling relationship, the offset of time and frequency are cooperatively regulated to improve the synchronization speed under the same synchronization period.

The main structure of this work is as follows. The second part is the clock model. The third part introduces the hierarchical synchronization mechanism proposed in this paper. The fourth part validates the effectiveness of the proposed mechanism through experiments. The fifth part is the conclusions.

II. Clock Model

The clock of network equipment is usually driven by a quartz crystal oscillator. The oscillator drives a counter which is filled with a constant value beforehand. When the oscillator sends a pulse, the counter is decremented by one until it reaches zero, and the clock is driven to generate a tick. The constant value is refilled into the counter and reloaded after that, repeating the above operation process [21].

We assume the time of the reference clock is $C(t) = t$, where t is the reference time with a frequency of 1. After a synchronization period t_0, a typical quartz-stabilized oscillator clock model of node i is,

$$C_i(t_0) = \Delta_i + \int_0^{t_0} f_i(t)dt, \tag{1}$$

where $C_i(t_0)$ and $f_i(t)$ is the clock time and real clock frequency of node i, respectively. Δ_i represents the initial clock offset between node i and the reference clock.

In practical applications, $f_i(t)$ is not a fixed value, and it will be affected by many factors, including initial errors and random errors caused by temperature and oscillator aging, etc. [15], [16]. So we can get the expression of the clock frequency of node i,

$$f_i(t) = F_i + \delta_i + \phi_i, \tag{2}$$

where F_i is the nominal frequency. δ_i is the initial errors after delivery due to manufacturing process errors, etc. of the oscillator. We assume that the maximum value is ρ, that is, $-\rho \leq \delta_i \leq +\rho$. $\phi_i \sim \mathcal{N}(0, \sigma^2)$.

III. Proposed Hierarchical Timing-frequency Synchronization Mechanism

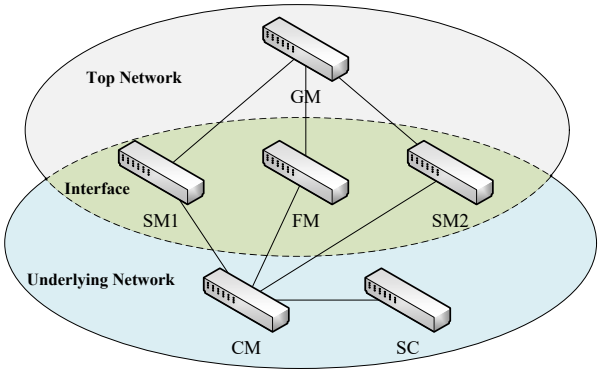


Fig. 1. Network hierarchy structure.

We divide the nodes in the network into the following categories: Grand Master (GM), Frequency Master (FM), Synchronization Master (SM), Compression Master (CM), and Synchronization Client (SC). The network is divided into two layers according to the network clock synchronization function. In terms of synchronous structure, the top network and underlying network adopt tree-based and distributed architectures, respectively. The hierarchical network structure is shown in Fig. 1. The two layers of the network use the SM and FM as the interface for information exchange, and time-frequency synchronize. We define a time synchronization period P_t as a period for performing a time offset correction and a frequency synchronization period P_f as a period for

performing a frequency offset correction. A frequency synchronization period consists of multiple time synchronization periods.

A. Top Network Synchronization Mechanism Design

The top network refers to PTP and uses the tree-based synchronization mechanism. The specific process is as follows.

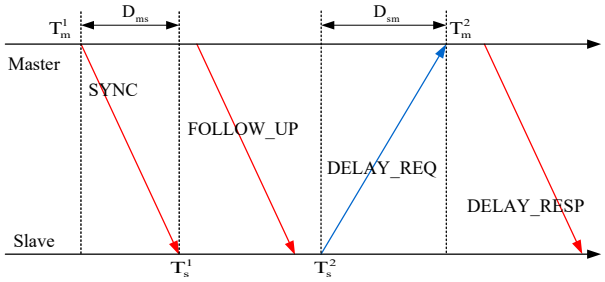


Fig. 2. Top network time synchronization process, D_{ms} and D_{sm} are represent downstream delay and upstream delay, respectively.

We assume that the master and slave nodes have the same frequency and are fixed, then,

$$\Delta = \frac{(T_s^1 - T_m^1) - (T_m^2 - T_s^2)}{2}, \tag{3}$$

where Δ is the clock offset.

In this way, the time offset between the master and slave nodes is calculated. In our top network, we set GM as the master node and SM as the slave node. At the same time, we take into account the frequency variation between nodes and adopt frequency synchronization to correct the frequency offset. We use the method shown in Fig. 3 to synchronize the frequency.

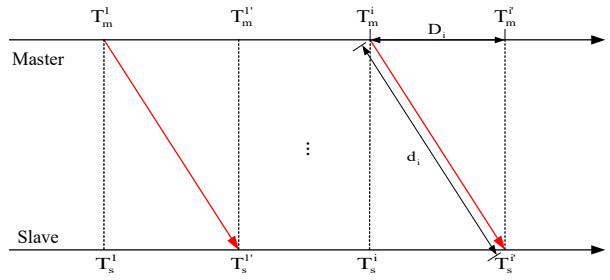


Fig. 3. Top network frequency synchronization process, D_i is the actual one-way delay (master to slave) of the detection packet and d_i is the measured one-way delay of the detection packet.

The sending time of the probe packet i at the master is T_m^i, and at the same time, the time of the slave is T_s^i; The time when the probe arrives at the slave is $T_s^{i'}$, and at the same time, the time of the master is $T_m^{i'}$.

Assume that the clock offset of the master and the slave at the beginning of the measurement is Δ_0, and $\Delta_0 = T_s^1 - T_m^1$. Assume that the clock frequency of the master is f_m, and the clock frequency of the slave is f_s. The ratio of the master clock to the slave clock is α. Then,

$$\alpha = \frac{f_m}{f_s} = \frac{F_m + \delta_m + \phi_m}{F_s + \delta_s + \phi_s}. \tag{4}$$

The measured one-way delay d_i is,

$$\begin{aligned} d_i &= T_s^{i'} - T_m^i = T_s^i + \alpha D_i - T_m^i \\ &= T_s^1 + \alpha(T_m^i - T_m^1) + \alpha D_i - T_m^i + (T_m^1 - T_m^1), \\ &= (\alpha - 1)(T_m^i - T_m^1) + \Delta_0 + \alpha D_i \end{aligned} \tag{5}$$

where $D_i = T_m^{i'} - T_m^i$. Because the clock frequency offset is generally not greater than 10^{-4}, and the one-way delay is usually in the millisecond range, d_i can be simplified as,

$$d_i = (\alpha - 1)(T_m^i - T_m^1) + \Delta_0 + D_i. \tag{6}$$

When the network structure does not change, it can be considered that the value of D_i is stable. At this time, it can be regarded as that the changing trend of d_i is determined by α, that is, the clock frequency offset can be calculated by measuring the delay and used to compensate.

In practical applications, we use GM as the master node and FM as the slave node. When GM sends data to SM, GM starts to synchronize the data to FM at a fixed time interval. The frequency offset is calculated according to formula (6), where Δ_0 can be approximately the time offset calculated by the first time synchronization period.

B. Underlying Network Synchronization Mechanism Design

After completing the clock synchronization with GM, SM enters the distributed time synchronization stage, generates frames that contain their clock information, and sends them to CM. This process refers to the AS6802 protocol.

After receiving the data sent by SM, CM obtains several permanence points through the permanence operation. During this stage, the required permanence point is obtained after collection. The permanence point is obtained by,

$$pp_i = d_{max} + rp_i - d_i, \tag{7}$$

where pp_i corresponds to the i-th collected permanence point. d_{max} is the maximum transmission delay in the network which can be defined in advance or measured experimentally. rp_i is the time point when CM received the data, and d_i represents the data transmission delay in the link.

Then we can have $p_i = pp_i - pp_1$, where p_i corresponds to the time difference between the i-th collected permanence point and the first permanence point. The following formula is taken to calculate the compression correction value $corr$:

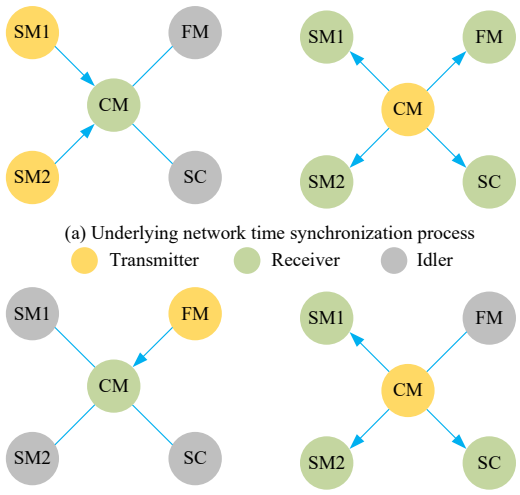
$$\begin{cases} 0 < k \leq 5, corr = \frac{p_f + p_{k-f+1}}{2} \\ k > 5, corr = \frac{p_f + p_{k-f}}{2} \end{cases}, \tag{8}$$

where k corresponds to the number of collected permanence points; f is the maximum number of SMs allowed to error in the network. If $0 < k \leq 2$, $f = 1$, and if $2 < k \leq 5$, $f = 2$.

After calculating the $corr$ based on the received frames, CM generates a new frame and send it to GM, SM, FM, and

SC. They adjust the local clock after receiving the frame sent by CM. This process completes the time synchronization of a time synchronization period.

We also use FM as the starting node and use synchronous Ethernet technology to transform some nodes of the underlying network and add clock phase-locked loop modules (PLL). In this way, FM injects the clock synchronized with GM into its physical layer and sends it to other nodes through the network. The other nodes act as relay nodes and continue to send clocks to other nodes. All nodes can restore the clock frequency based on the received data. This process completes the frequency synchronization of a frequency synchronization period. The specific synchronization process of the underlying network is shown in Fig. 4.



(a) Underlying network time synchronization process

(b) Underlying network frequency synchronization process

Fig. 4. Underlying network clock synchronization process.

C. Hierarchical Network Connection Mechanism Design

The top and underlying network use SM and FM as interfaces. SM and FM are responsible for time and frequency synchronization, respectively. At the end of each P_f, GM needs to send its local clock information to CM, which is compared with the underlying network clock reference calculated by the CM. If CM does not receive data from GM or the time offset between them is bigger than a preset threshold ζ, GM is considered to be in an abnormal state.

If GM is in the abnormal state, SM directly adopts distributed synchronization, and FM uses its frequency as the reference for frequency synchronization. Otherwise, SM continues to perform tree-based time synchronization with GM, and FM also adjusts the local frequency according to the frequency of GM. The processing of SM, CM, and FM can be summarized as the following three algorithms.

Algorithm 1 SM clock synchronization process in a P_f period

Input: Time stamp sent by GM, T_m^1 and T_m^2; Local time stamp of SM, T_s^1 and T_s^2; Time correction value sent by CM, $corr$; Frequency sent by FM, f_{FM}; GM status GS;
Output: Local time information of SM, T'_{SM};

while $time \in P_f$ **do**
  while $time = nP_t, n = 1, 2 \ldots$ **do**
    if GS = normal **then**
      GM exchanges information with SM, and SM gets T_s^1 and T_s^2;
      if receives T_m^1 and T_m^2 **then**
        SM adjusts local time with $\Delta = [(T_s^1 - T_m^1) - (T_m^2 - T_s^2)]/2$;
      end if
    end if
    SM sends T'_{SM} to CM;
    if receives $corr$ **then**
      SM adjusts local time;
    end if
    if $time + P_t > P_f$ **and** receives f_{FM} **then**
      SM adjusts local frequency;
    end if
  end while
end while

Algorithm 2 CM clock synchronization process in a P_f period

Input: Time information sent by GM, T_{GM}; Time information sent by SM, T'_{SM}; Local time information of CM T_{CM}; Frequency sent by FM, f_{FM}; Preset threshold, ζ;
Output: Time correction value $corr$; GM status, GS;

while $time \in P_f$ **do**
  while receives T'_{SM} **do**
    CM calculates $corr$ and adjusts local time;
    CM sends $corr$ to FM, SM and SC;
    if $time + P_t > P_f$ **then**
      if receives f_{FM} **then**
        CM adjusts local frequency;
      end if
      $GS = abnormal$;
      if receives T_{GM} **and** $|T_{GM} - T_{CM}| \leq \zeta$ **then**
        $GS = normal$;
      end if
    end if
  end while
end while

IV. Performance Evaluation

In this section, we investigate the effectiveness of our proposed hierarchical clock synchronization mechanism. A network structure was built in OMNET++ to verify the performance, as shown in Fig. 5. We first remodeled the clock in OMNET++ according to the mathematical model of the clock (1) and (2) established in section II. The frequency of GM is set to 1MHz. Set the parameters of all other nodes to $\Delta_0 = 20$ppm, $F_i = 1$MHz, $\delta = 20$ppm, and $\phi_i \sim \mathcal{N}(0, 0.1^2)$. We choose the SC node as the comparison node and set the bandwidth of all links to 100M. The propagation delay of all

Algorithm 3 FM clock synchronization process in a P_f period

Input: Time information sent by GM, T'_{GM}; Time correction value sent by CM, $corr$; GM status, GS;
Output: Local frequency information of FM, f_{FM};

while $time \in P_f$ **do**
  repeat
    if GS = normal **then**
      if receives T'_{GM} **then**
        FM calculates frequency offset and adjusts local frequency;
      end if
      if receives $corr$ **then**
        FM adjusts local time;
      end if
    end if
  until $time + P_t > P_f$
  FM generates f_{FM} and injects it into the network;
end while

links is set to 25ns. Then the proposed mechanism is compared with the gPTP in terms of synchronization accuracy and speed.

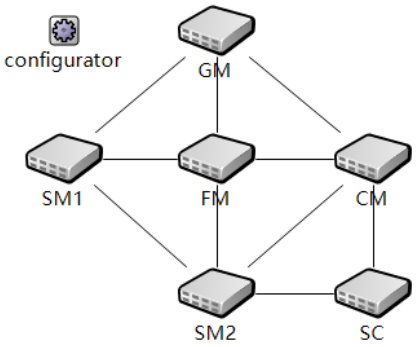


Fig. 5. Network structure in OMNET++.

A. Synchronization Accuracy

Fig. 6 shows the tendency of the synchronization errors over time when using the synchronization mechanism we proposed and gPTP with $P_t = 2$s. From Fig. 6, one can see that the synchronization errors of gPTP produce large fluctuations due to the influence of frequency offset, and errors of our proposed mechanism change steadily. Under our proposed mechanism, the errors are lower than those under gPTP. The results imply the effectiveness of our mechanism. The frequency offset can cause cumulative time errors, resulting in poor performance. Our proposed mechanism synchronizes the frequency and eliminates errors caused by frequency offset.

Fig. 7 shows the synchronization errors that our proposed mechanism and gPTP can achieve with different P_t. The box plot in the figure is the synchronization errors obtained with the increase of simulation time in each period. The line in the figure and the small graph are the medians of the synchronization errors obtained with different P_t. We can see that with the same P_t, the errors of the proposed mechanism

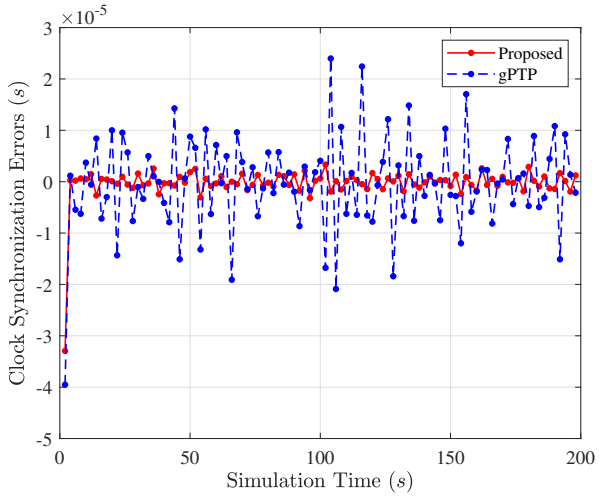


Fig. 6. Synchronization process comparison of two methods when $P_t = 2$s.

is smaller than that of gPTP, which means our proposed mechanism can maintain high synchronization accuracy. As the P_t decreases, this difference also decreases. This is because as P_t decreases, the errors caused by the frequency offset gradually decrease, and the gPTP errors also become smaller. Furthermore, from the box plot in the figure, we can see that the errors of the proposed mechanism are limited to a specific range, and the fluctuation is smaller, which is better than gPTP.

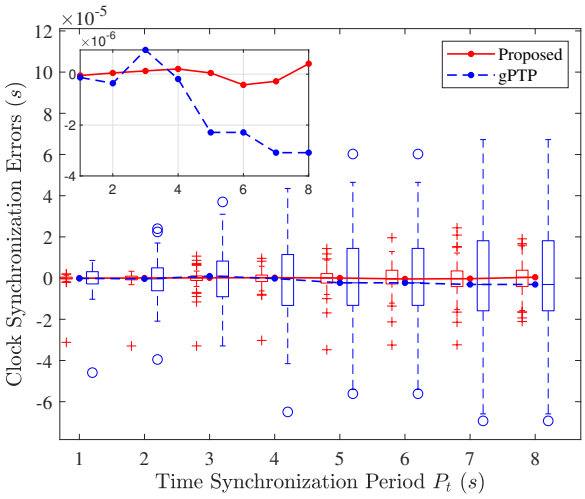


Fig. 7. Synchronization errors comparison of two methods with different P_t.

B. Synchronization Speed

Fig. 8 shows the synchronization speed of the two mechanisms with different P_t. We define the synchronization speed as the time when the relative error between the current synchronization accuracy and the median synchronization accuracy is 20%. It can be observed from the figure that the time required for our proposed mechanism is shorter than gPTP, which means that the synchronization speed of our proposed

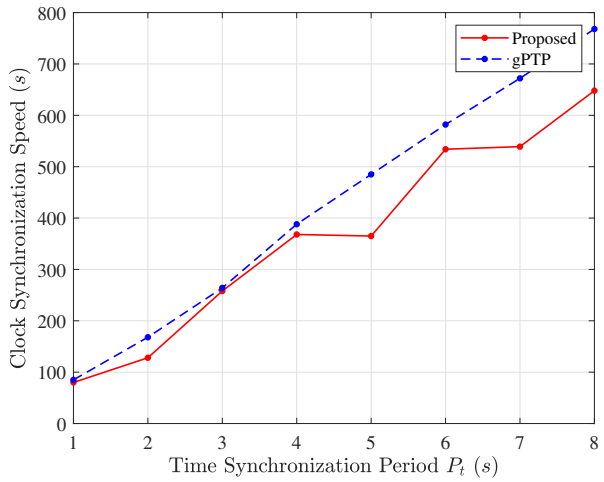


Fig. 8. Synchronization speed comparison of two methods with different P_t.

mechanism is faster and synchronization efficiency is higher than gPTP. The synchronization speed of gPTP is slow because it requires multiple cycles of synchronization to eliminate the errors gradually. Moreover, the frequency offset will continue to bring errors, resulting in a long time to achieve the required synchronization accuracy. Our method relies on the SM to provide a reference clock to eliminate errors at the beginning of synchronization quickly. After frequency synchronization, the errors become smaller, so that the synchronization accuracy is always kept in a low range.

C. Performance Analysis

The synchronization mechanism proposed in this work adopts a hierarchical structure. Since all SMs are synchronized with GM, the calculation of the underlying network can eliminate random errors. Moreover, due to the combination of distributed synchronization, which avoids the situation that tree-based synchronization errors will accumulate as the network scale increases, the synchronization process can be increased compared to gPTP with the same synchronization period. It can achieve the required synchronization accuracy faster, resulting in higher synchronization efficiency, which can also be seen from Fig. 8.

V. Conclusions

This paper has studied the clock synchronization for TSN. A hierarchical clock synchronization mechanism was proposed. Considering the synchronization time slot difference and the time-frequency coupling relationship, the offset of time and frequency are cooperatively regulated. Through the performance analysis and verification, the effectiveness and performance improvement of the proposed hierarchical synchronization mechanism was demonstrated. In the future, we will do further research on synchronization reliability.

Acknowledgment

This work was supported by National Key Research and Development Program of China (2018YFB1702100).

References

[1] W. Zhang, Z. Wu, G. Han, Y. Feng, and L. Shu, "LDC: A lightweight dada consensus algorithm based on the blockchain for the industrial internet of things for smart city applications," *Future Generation Computer Systems*, vol. 108, pp. 574–582, 2020.

[2] J. Wan, J. Li, M. Imran, and D. Li, "A blockchain-based solution for enhancing security and privacy in smart factory," *IEEE Transactions on Industrial Informatics*, vol. 15, pp. 3652–3660, 2019.

[3] S. Vitturi, C. Zunino, and T. Sauter, "Industrial communication systems and their future challenges: Next-generation ethernet, IIoT, and 5G," *Proceedings of the IEEE*, vol. 107, pp. 944–961, 2019.

[4] G. Sestito, A. Turcato, A. Dias, M. Rocha, M. Da Silva, P. Ferrari, and D. Brandao, "A method for anomalies detection in real-time ethernet data traffic applied to profinet," *IEEE Transactions on Industrial Informatics*, vol. 14, pp. 2171–2180, 2018.

[5] K. Langlois, T. Van Der Hoeven, D. Rodriguez Cianca, T. Verstraten, T. Bacek, B. Convens, C. Rodriguez-Guerrero, V. Grosu, D. Lefeber, and B. Vanderborght, "Ethercat tutorial: An introduction for real-time hardware communication on windows [tutorial]," *IEEE Robotics and Automation Magazine*, vol. 25, pp. 22–25 and 122, 2018.

[6] M. Knezic, B. Dokic, and Z. Ivanovic, "Theoretical and experimental evaluation of ethernet powerlink pollresponse chaining mechanism," *IEEE Transactions on Industrial Informatics*, vol. 13, pp. 923–933, 2017.

[7] M. Wollschlaeger, T. Sauter, and J. Jasperneite, "The future of industrial communication: Automation networks in the era of the internet of things and industry 4.0," *IEEE Industrial Electronics Magazine*, vol. 11, pp. 17–27, 2017.

[8] F. Zezulka, P. Marcon, Z. Bradac, J. Arm, T. Benesl, and I. Vesely, "Communication systems for industry 4.0 and the IIoT," *IFAC-PapersOnLine*, vol. 51, pp. 150–155, 2018.

[9] A. Nasrallah, A. Thyagaturu, Z. Alharbi, C. Wang, X. Shao, M. Reisslein, and H. ElBakoury, "Ultra-low latency (ULL) networks: The IEEE TSN and IETF detnet standards and related 5G ULL research," *IEEE Communications Surveys and Tutorials*, vol. 21, pp. 88–145, 2019.

[10] W. Yang and M. Fu, "A filter-based clock synchronization protocol for wireless sensor networks," *Asian Journal of Control*, vol. 21, pp. 1389–1403, 2019.

[11] D. Mazur, R. Entzminger, J. Kay, and P. Morell, "Time synchronization mechanisms for the industrial marketplace," *IEEE Transactions on Industry Applications*, vol. 53, pp. 39–46, 2017.

[12] "IEEE draft standard for a precision clock synchronization protocol for networked measurement and control systems," *IEEE P1588/D1.5, May 2019*, pp. 1–568, June 2019.

[13] L. S. C. of the IEEE Computer Society, "Timing and synchronization for time-sensitive applications in bridged local area networks," *IEEE Transactions on Industrial Electronics*, vol. 57, pp. 4106–4114, 2010.

[14] W. Steiner, G. Bauer, B. Hall, and M. Paulitsch, *Time-triggered ethernet*, 2011.

[15] M. Lévesque and D. Tipper, "A survey of clock synchronization over packet-switched networks," *IEEE Communications Surveys and Tutorials*, vol. 18, pp. 2926–2947, 2016.

[16] Z. Liu, Y. Cheng, P. Wang, Y. Yu, and Y. Long, "A method for remaining useful life prediction of crystal oscillators using the bayesian approach and extreme learning machine under uncertainty," *Neurocomputing*, vol. 305, pp. 27–38, 2018.

[17] P. Jia, X. Wang, and K. Zheng, "Distributed clock synchronization based on intelligent clustering in local area industrial iot systems," *IEEE Transactions on Industrial Informatics*, vol. 16, pp. 3697–3707, 2020.

[18] H. Wang, L. Shao, M. Li, and P. Wang, "Estimation of frequency offset for time synchronization with immediate clock adjustment in multihop wireless sensor networks," *IEEE Internet of Things Journal*, vol. 4, pp. 2239–2246, 2017.

[19] L. Fan, Y. Ling, T. Wang, X. Zhu, and X. Tang, "Novel clock synchronization algorithm of parametric difference for parallel and distributed simulations," *Computer Networks*, vol. 57, pp. 1474–1487, 2013.

[20] X. Zhang, H. Chen, K. Lin, Z. Wang, J. Yu, and L. Shi, "RMTS: A robust clock synchronization scheme for wireless sensor networks," *Journal of Network and Computer Applications*, vol. 135, pp. 1–10, 2019.

[21] S. Hwang, "A network clock model for time awareness in the internet of things and artificial intelligence applications," *Journal of Supercomputing*, vol. 75, pp. 4309–4328, 2019.