A Novel Architecture Combining Oracle with Decentralized Learning for IIoT

Yijing Lin, Zhipeng Gao, *Member, IEEE,* Weisong Shi, *Fellow, IEEE,* Qian Wang, Huangqi Li, Miaomiao Wang, Yang Yang, and Lanlan Rui, *Member, IEEE*

***Abstract*—The rapid development of digital technology is reshaping the architecture of the Industrial Internet of Things (IIoT). The traditional architecture can not process vast amounts of data exchanges and provide entities with trust. The future IIoT is expected to be a decentralized architecture in which blockchain and digital twin-driven IIoT can enable trusted data exchanges. However, this architecture can not obtain huge amounts of external real-time data and isolated data. Moreover, it can not handle complex industrial computing tasks. Therefore, we combine oracle with decentralized learning to propose a novel IIoT-oriented digital twin architecture. We also propose an effective decentralized collaboration mechanism to support external data and resources exchanges. Moreover, we propose a novel computing collaboration mechanism to expand the learning capabilities of the industrial ecology. Experiments show that our proposed paradigm has less processing time, a more stable process, and better learning ability compared to other paradigms.**

***Index Terms*—IIoT, Digital Twin, Blockchain, Oracle**

I. Introduction

THE fast-developing technologies such as IoT, 5G, and digital twins are reshaping the industrial world. Networked equipment, ultra-low latency, and rapid feedback to physical and digital systems change the way that industrial machines work. The development of industrial machines goes through three stages. First, they require human intervention to complete tasks. Second, they customize production for human needs. Finally, they give feedback to human beings. Human wisdom is illuminating the foggy future of Industry 4.0, devoting all efforts to manufacturing new intelligent IIoT machines.

However, every coin has two sides. Firstly, advanced industrial machines and bidirectional data flow of digital twin bring rapid data exchanges. When data is transmitted to cloud servers, it consumes lots of bandwidth, bringing severe challenges to the traditional architecture. Secondly, different types of data sources, like database diagrams, interfaces, and files, of multiple industrial participants make it hard for digital entities to give feedback to physical entities. Finally, industrial entities are reluctant to share data with others, which challenges the precarious centralized architecture. Blockchain is a brand new decentralized trust machine [1]. It can connect entities without trust through consensus mechanisms. Jiewu L. *et al.* [2] proposed a two-tier industrial architecture based on blockchain and digital twin for smart factories, which realizes synchronization of physical and digital systems. Yunlong L. *et al.* [3] proposed a digital twin edge network based on federated learning, edge computing, and blockchain to enhance the safety of learning.

However, blockchain is a collection of programs based on rules and method calls with redundant backup, while an industrial digital twin network possesses massive data. It is difficult for a single blockchain to handle tons of IIoT data. Furthermore, since the logic of applications runs in the virtual machine of blockchain. It is required that each virtual machine has the same result. The reason is that unknown data sources may produce different results on different virtual machines. For example, if a request includes an interface about real-time industrial data which changes every second, it is impossible for every node in the blockchain to obtain the same result. Therefore, blockchain is hard to actively interact with the industrial environment, which hinders data exchanges between digital and physical entities. Moreover, it is almost impossible for an isolated blockchain to execute cumbersome computing tasks because tasks run in the virtual machine. And it is inefficient to perform those tasks repeatedly in the virtual machines. There is an urgent need to give birth to a new data and collaborative computing paradigm for IIoT. Therefore, we propose a novel blockchain-based digital twin architecture for IIoT. This architecture exploits oracle as a communication tool to link data in the IIoT ecology. The oracle is not a well-known database but a tool that connects on-chain and off-chain states. Moreover, we also proposed a decentralized collaboration mechanism based on oracle, verifiable random functions, and threshold signatures for trusted off-chain data forwarding and callback. We also propose a data collaboration mechanism and a computing collaboration mechanism to implement data interactions and resource sharing between digital and physical entities based on the decentralized collaboration mechanism.

Main contributions of this paper are summarized as follows.

- We propose a novel decentralized IIoT architecture based on blockchain, oracle, and digital twin. This architecture

This work is supported by National Natural Science Foundation of China (62072049), and BUPT Excellent Ph.D. Students Foundation (CX2021133). (*Corresponding author: Zhipeng Gao)*

Yijing Lin, Zhipeng Gao, Huangqi Li, Miaomiao Wang, Yang Yang and Lanlan Rui are with the State Key Laboratory of Networking & Switching Technology, Beijing University of Posts and Telecommunications, Beijing, 100876, China. (e-mail: yjlin@bupt.edu.cn; gaozhipeng@bupt.edu.cn; lhq320@bupt.edu.cn; wangmiaomiao@bupt.edu.cn; yyang@bupt.edu.cn; llrui@bupt.edu.cn)

Weisong Shi is a professor at Wayne State University, Detroit, 48202, US. (e-mail: weisong@wayne.edu)

Qian Wang is with Beijing University of Technology, Beijing, 100124, China. (e-mail: wangqian2020@bjut.edu.cn)

is hierarchical to meet the requirements of a complex industrial environment. Moreover, it helps rapid data flows and computing feedbacks between digital and physical entities.
- We propose an effective decentralized collaboration mechanism based on oracle, verifiable random functions, and threshold signatures, which supports external data and resources exchanges between industrial entities, and protects data flows consistency. Moreover, we propose a data collaboration mechanism based on the decentralized collaboration mechanism to provide data exchanges between digital entities and physical entities.
- We propose a novel computing collaboration mechanism based on blockchain and decentralized learning to expand learning capabilities of the industrial ecology, and keep the authenticity of callback computing results.

The rest of the paper is organized as follows. Related work is discussed in Section II. Bi-level architecture for digital twin is introduced in Section III. Key enabling techniques are presented in Section IV. The proposed method is evaluated in Section V and the paper is concluded in Section VI.

II. Related Work

This architecture is based on various emerging and enabling technologies such as blockchain, oracle, decentralized learning, and digital twin. Therefore, we divide the literature review into three parts: (a) data collaboration with blockchain, (b) computing collaboration with learning, and (c) digital twin.

A. Data Collaboration with Blockcahin

Blockchain is an emerging peer-to-peer data sharing paradigm with the birth of Bitcoin [1]. It has received extensive attention from institutes. At present, the development of blockchain is in the 3.0 era, which is an application outside the financial scenario [4]. The current stage of blockchain is working on satisfying more complex business logic and promoting industrial reform. Furthermore, IIoT is becoming an important application scenario. At the same time, smart contract [5] [6], a product of the 2.0 era, has become an essential mean to meet the complex IIoT logic in the 3.0 era.

However, it is hard to deal with a complex IIoT environment with a large amount of data because of the redundant backup feature of blockchain, which challenges the storage capability of blockchain. For example, it is impossible for most existing blockchains to store tons of pictures. And data from multiple sources challenges the collaboration ability of blockchain. For example, blockchain cannot reach a consensus since every node may get different results from the same interface. To solve the challenge of storage capacity, Yijing L. *et al.* [7] proposed an on-chain and off-chain collaboration mechanism to expand the storage of blockchain by swapping off-chain decentralized storage spaces for on-chain storage abilities. Zhaofeng M. *et al.* [8] proposed to manage data of edge devices on cloud services based on blockchain. Moreover, there are some methods to solve the challenge of collaboration ability, including sidechain, rollup, multiple blockchains, etc. Considering a large number of participants and the relationship of industrial ecology, we focus on the solution of multiple blockchains to cope with problems of the above issue. Gokhan S. *et al.* [9] proposed a hybrid blockchain architecture. The sub-chain adopts Proof of Work while the inter-chain adopts relays like Polkadot[10] and Cosmos[11] to realize data collaboration in the hybrid architecture. Wenyu L. *et al.* [12] proposed a scalable PBFT consensus based on multiple blockchains to improve collaboration efficiency. Clients' transactions are first distributed on blockchain in the first layer to complete initial consensus and then uploaded to blockchain in the second layer for final consensus. Shaoyong G. *et al.* [13] proposed a collaboration method based on the master-slave chain. Transactions are firstly voted on the slave chain, and then the node with the highest reputation uploads transactions to the master chain for data sharing. However, those methods fail to actively obtain external data and execute complex computing tasks. For example, blockchain cannot actively obtain real-time price data of raw materials and train item sorting tasks. It means that blockchain cannot react to the unknown state from the outside world according to specific interfaces because the same interface may lead to different results in the virtual machines.

Oracle is a tool that connects blockchain with real external data [14], which is considered as an important application scenario for interoperability [15]. It bridges smart contracts with the real world to achieve data collaboration. Chainlink [16] is the first decentralized data oracle on Ethereum. It proposed a reporter-based aggregation method to reduce transaction fees in the current version. DOS Network [17], Astraea [18], Nest [19] and other solutions also propose blockchain-agnostic schemes of data oracle for public blockchains. However, IIoT has a large amount of data, numerous equipment, and limited participants, which is not suitable for adopting solutions of public blockchains. Moreover, there is no solution for executing complex computing tasks.

B. Computing Collaboration with Learning

Edge computing [20] is an emerging paradigm, which decentralizes computing and storage capabilities to the edge reducing the load on the core network. The distributed nature of edge computing matches with blockchain. Kaile X. *et al.* [21] proposed a blockchain-based resource pre-allocation algorithm for edge computing. Yueyue D. *et al.* [22] implemented content caching based on deep reinforcement learning and blockchain. Moreover, Swarm Learning [23] proposed to train medical models based on blockchain, edge computing, and federated learning [24]. Clients train models locally and write model parameters into smart contracts. The client which completes training first is the node that aggregates parameters and distributes optimal models. Yunlong L. *et al.* [3] integrated blockchain with federated learning and edge computing and proposed a new edge digital twin network. Digital entities upload local parameters to blockchain for aggregation. The node selected by consensus is required to aggregate global models. However, they do not consider malicious entities in the network. Chao Q. *et al.* [25] proposed an edge intelligence

blockchain, which exploits computing models as mining puzzles to avoid waste of energy. It uses computing power to solve model parameters. The node which claims the smallest loss is the coinbase node in the communication round. The coinbase node has the privilege to write global parameters into blockchain. Other nodes need to use the same dataset to verify the authenticity of the claimed loss. If the loss is valid, they update local models. However, it is a waste of energy in the large-scale network to push all nodes to verify the authenticity of the claimed loss. In summary, blockchain does not execute data and computing collaboration in the above methods, which hinders interaction between participants.

It should be noted that we only focus on that clients may be malicious entities. For the situation of malicious servers, we can refer to Xianglong Z. *et al.* [26] who proposed a bilinear aggregate signature and homomorphic encryption-based method to verify whether the specific server aggregates the parameters of each client.

C. Digital Twin

With the development of new generation digital technologies, the future IIoT is expected to enable a new and wide range of decentralized systems [27]. Mohammad A. *et al.* [28] proposed a middleware based on fog computing to adapt to different industrial scenarios. Sambit Kumar M. *et al.* [29] proposed a sustainable service distribution method based on fog computing to solve energy consumption. At the same time, we can control industrial equipment bidirectionally by commands and data, making the concept of digital twins attract more attention. Digital twin is one of the basic techniques of metaverse. Its concept is simple that it can connect physical and digital entities in a precise and real-time manner [30]. However, heterogeneous data and trust between entities are stumbling the development of digital twins [31].

Trusted sharing of blockchain naturally adapts to digital twin. Jiewu L. *et al.* [2] proposed a two-layer architecture combined with blockchain and digital twin. This architecture exploits blockchain to control the self-organization of low-level nodes for consensus, while it uses digital twin to control the high-level ones for data flows. Jiafu W. *et al.* [32] proposed a blockchain-based IIoT architecture and data interaction algorithms to solve data heterogeneity. However, how to realize the life cycle of data collaboration and computing collaboration in digital twin remains unexplored.

III. Bi-level Architechture for Digital Twin

This section mainly describes the network architecture of digital twin, as shown in Figure 1. It is divided into physical entities and digital entities. The physical entities include various industrial equipment and network devices in the workshop. Digital entities have a control panel that receives requirements and preferences, blockchains that maintain credibility, and an oracle connecting blockchains and entities. And it can also be classified into on-chain parts and off-chain parts. On-chain parts mean that operations and data involve in the blockchain. Off-chain parts mean that data, state and operations are performed outside the blockchain. And the collaboration means that data or operations should be performed in the cooperation of on-chain and off-chain parts. Since blockchain is hard to obtain real-time data from the real world, it is necessary to collaborate between on-chain and off-chain parts.

A. Architecture

The architecture comprises Device Layer, Network Layer, Storage Layer, Pedal Layer, and Control Layer in the order of data flow.

a) Device Layer: This layer is composed of physical entities like intelligent machines and workpieces. Physical entities collect data through sensors and upload data to digital entities through Network Layer. Moreover, they are also actuators, which convert electrical signals into some physical actions.

b) Network Layer: This layer is composed of communication equipment like edge gateways and base stations. They are managed over the internet through remote procedure calls. They receive sensing data from physical entities, upload data to Storage Layer after preprocessing, and distribute commands of digital entities to Storage Layer. They requires quick response and security measures to help achieve bidirectional data flows.

c) Storage Layer: This layer includes off-chain cloud services and edge blockchain networks, both of which complement each other. The redundant backups of blockchain make the architecture unable to store large-scale data. Moreover, the simple data type and consensus of blockchain mean it is hard to obtain data from multiple sources of digital twin. Most importantly, each participant in the IIoT ecosystem would like to store data in a local data center. They are not willing to upload all data to decentralized storage. Therefore, we choose cloud services and blockchain to form the Storage Layer. However, all cloud-based solutions need to solve the problem that data may be tampered with. We use cloud services to keep source data produced by physical entities and upload metadata of source data to edge blockchain networks for a unique mapping index. This solution builds unified data and model standards while expanding the storage capacity of blockchain and preventing data from being tampered with. Blockchain and cloud services of Storage Layer interacts with digital and physical entities through Pedal Layer, see details in Section IV.

d) Pedal Layer: Pedal means an essential part of the architecture. This layer comprises data oracles for data collaboration and computing oracles for computing collaboration. Blockchain can not actively interact with external industrial entities because each node needs to have the same result for consensus in the same transaction request. Moreover, contracts run in the virtual machines while it is inefficient to execute cumbersome tasks. Therefore, we introduce data and computing oracles to connect blockchain and the external industrial world.

e) Control Layer: This layer is composed of a digital twin platform. It provides an interface for users to query, add, delete, and check based on above layers. The platform can also provide predictive maintenance, diagnostics, and failure avoidance. Moreover, it can help decision-makers build an

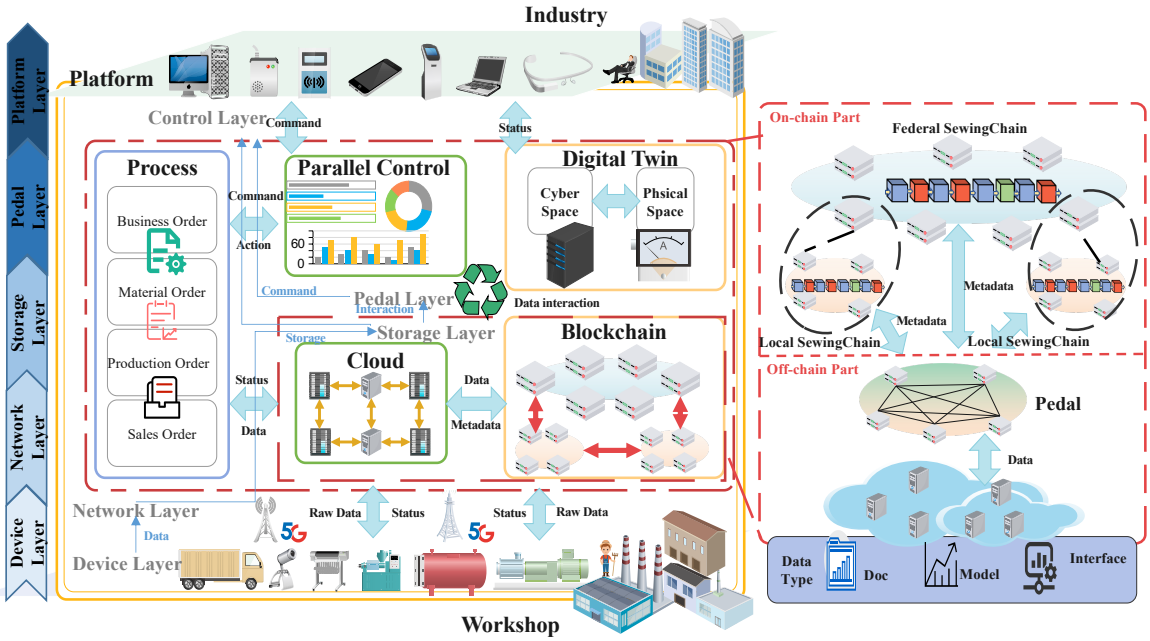


Fig. 1: System overview

industrial portrait at all levels to control and adjust equipment efficiently.

The corresponding relationship between the five layers is described in the following details. Industrial data is firstly produced in the device layer and transmitted to the storage layer by the network layer. Digitial entities and physical entities interact with each other through the pedal layer. And users can control the above processes by the control layer.

B. SewingChain: On-chain Part

There are many entities with numerous data from many fields in the IIoT ecosystem. It is difficult for participants in the ecosystem to build a single blockchain for data sharing. Therefore, we build a multi-blockchain ecosystem named SewingChain. SewingChain is composed of a federal blockchain for a regulatory role and local blockchains for participating roles.

a) Federal Blockchain (FBC): FBC is composed of entities to achieve global industrial data governance. Nodes participating in consensus and verification processes of FBC are called federal nodes, denoted as F_g.

b) Local Blockchain (LBC): Digital entities can form multiple LBCs to realize local industrial data governance. Nodes participating in consensus and verification processes of LBC are called local nodes, denoted as L_g. Nodes with query authorities are called observation nodes, denoted as L_o. Nodes that can interact with FBC are denoted as L_f. L_g accepts metadata from entities and stores it in LBC after consensus. If cloud services tamper with source data, then the metadata of source data is not equal to that of blockchain. L_o provides a query interface for the digital twin platform to facilitate the analysis and mining of industrial data. Keeping the integrity of source data is a challenge of most existing blockchains. SewingChain can verify the integrity of off-chain source data by mapping unique metadata.

C. Pedal: Off-chain Part

Pedal is a off-chain network composed of data oracle and computing oracle. Data oracle is a bridged network for data collaborations, see details in Section IV-B2. Computing oracle is a bridged network for computing collaborations, see details in Section IV-C.

a) Data Oracle: This network consists of an application contract for receiving industrial data requests, a proxy contract for providing a unified data interface, and data oracle for obtaining and callbacking off-chain industrial data. The application contract is a contract that implements data interaction according to user needs, while the proxy contract is a contract that provides interactive interfaces for data oracle. Entities call the application contract, then the application contract forwards requests to the proxy contract and triggers events listened by data oracle. Data oracle executes operations according to specific requests, aggregates results, and callbacks results to the proxy contract. Finally, the proxy contract calls application contract to broadcast data to the client. For detail, see Section IV-B2.

b) Computing Oracle: This network consists of an application contract for receive computing requests, a proxy contract for providing a unified computing interface, and a computing oracle for computing and callback training results. Entities write metadata of computing tasks into the application contract, then the application contract forward tasks to the proxy contract, and trigger the computing shared event listened by computing oracle. Computing oracle performs calculations and callback training results to blockchain. See Section IV-C for details.

IV. Key Enabling Techniques

This section introduces the decentralized collaboration mechanism in Pedal, data collaboration mechanism with data oracle, and computing collaboration mechanism with computing oracle. The decentralized collaboration mechanism is the core of the proposed architecture, which helps data and computing collaboration to implement trusted off-chain data forwarding and callback. Data collaboration mechanism constructs trusted data interactions between blockchains and the real world. the computing collaboration mechanism constructs trusted resource sharing between providers.

A. The Decentralized Collaboration Mechanism in Pedal

This section introduces the decentralized collaboration in Pedal, as shown in Algorithm 1. The method is based on verifiable random functions [33] and threshold signatures [34] to support collaboration in data and computing oracles. The collaboration process is divided into registration, election, aggregation, and callback. We assume that nodes of Pedal are anonymous to each other and can not know their true identities.

a) Registration: Nodes participating in the pedal need to register in advance for legal identities. The generation of identities require k positive integers $s_1, s_2, ..., s_k$ while $gcd(s_i, s_j) = 1(i \neq j)$ as the seed. Generators g_1 of multiplicative cyclic group $\mathbb{G}_1$ generates $key(pk_i, sk_i, x_i)$, where x_i is the bilinear aggregate signature calculating as $g_1^{x_i}$. Threshold signature includes a bilinear map e where $\mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$. For two requests req_1 and req_2, they satisfy $h(req_1) \in \mathbb{G}_2$ and $h(req_1 + req_2) = h(req_1)\Delta h(req_2)$, where h is a homomorphic hash function.

b) Election: After nodes register in the contract $FeElect$, it is necessary to elect a master node by verifiable random function. It relies on an arbitrary-length string to generate $(sk_v, \alpha) \overset{compute}{\longrightarrow} \tau$, where α is a string composed of message. Moreover, it is necessary to determine a specific term to prevent fraud. The seed of the master node is empty in the first term.

Nodes generates public keys pk_{v_i}, private keys sk_{v_i}, signatures $(sk_{v_i}, \alpha) \overset{hash}{\longrightarrow} \beta_i$, random and proof $(sk_{v_i}, \alpha) \overset{prove}{\longrightarrow} (d_i, \pi_i)$ for β_i. Nodes whose d_i is less than $threshold$ are selected as the master node. The $threshold$ is determined by α. The sk and π are generated locally by each node while other nodes can not know which is selected as the master node in advance. There may be two situations. First, if only one node meets the condition, this node is the master node. Second, if there are multiple nodes or no node meeting conditions, we can poll nodes from $FeElect$. Subsequently, the master node randoming to select work nodes from the candidate pool and generate public key of this working group.

c) Aggregation: Nodes obtain results $req \leftarrow (req, result)$ according to contents of message after catches req. Nodes generates signatures $\sigma_i = (h(req))^x$, and forward (σ_i, req) to the master node. The master node aggregate (σ_i, req) as (1).

$$e(g_1, \sigma) = e(g_1, \prod_{i=1}^{k} \sigma_i) \tag{1}$$

Algorithm 1 The Decentralized Collaboration in Pedal

function ELECTION(α) ▷SewingChain
  for each node $i \in$ Pedal **in parallel do**
    $\beta_i \leftarrow$ vrf.hash(sk_{v_i}, α)
    $d_i, \pi_i \leftarrow$ vrf.prove(sk_{v_i}, α)
  end for
  $\tau \leftarrow$ vrf.genThreshold(sk_v, α)
  if vrf.compare(d_i, τ) <0 **then**
    master[$term$] = $node_i$
    group[$term$] = group[$node_i$ mod g]
    group[$term$].pks = $\prod_{i=1}^{k} pk_i$
  end if
end function

function AGGREGATE(req) ▷Pedal
  for each node in group[$term$] **in parallel do**
    $\sigma_i \leftarrow$ bls.sign(sk_i, req)
    Send (req, σ_i, pk_i) to master[$term$]
  end for
  $(sum, \sigma_i, pk_i, req) \leftarrow$ master[$term$] receive from group[$term$]
  if $sum >= \frac{2}{3} len(group[term])$ **then**
    $\sigma \leftarrow$ bls.aggregate($\sum_(\sigma_i, req, pk_i)$)
    master[$term$] write $(\sigma, pk_i, req, pk_{v_i}, \alpha, \beta_i, \pi_i)$ into SewingChain
  end if
end function

function CALLBACK($node, req, msg, asig$)▷SewingChain
  if vrf.verify($pk_{vi}, \alpha, \beta_i, \pi_i$) **then**
    if bls.AVerify($\sigma, req, group[term].pks$) **then**
      Callback $LoProd$
      Remove req from the pending queue
    end if
  end if
end function

The master node writes aggregated message and signature into $LoProxy$. $LoProxy$ checks data as described in (2) and (3).

$$(pk_{v_i}, \alpha, \beta_i, \pi_i) \overset{verify}{\longrightarrow} valid \tag{2}$$

$$(d_i, \tau) \overset{compare}{\longrightarrow} valid \tag{3}$$

$LoProxy$ accepts if the following equation holds as (4), otherwise, reject.

$$\begin{aligned} e(g_1, \sigma) = e(g_1, \prod_{i=1}^{k} (h(req_i))^x) = e(g_1, (\prod_{i=1}^{k} h(req_i)))^x \\ = e(g_1^x, h(\sum_{i=1}^{k} req_i)) = e(g_1^x, h(req)) \end{aligned} \tag{4}$$

$LoProxy$ callbacks aggregated results to $LoProd$ and removes req from the executing queue.

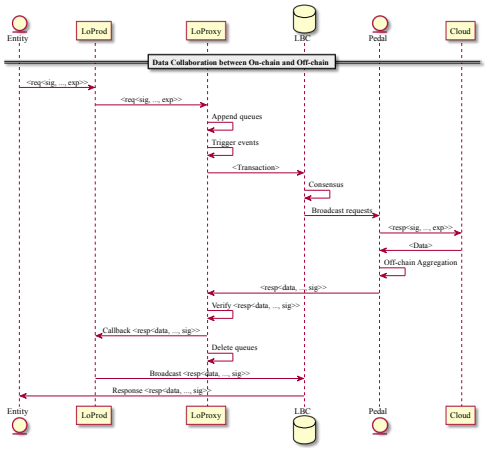


Fig. 2: On-chain and Off-chain Collaboration

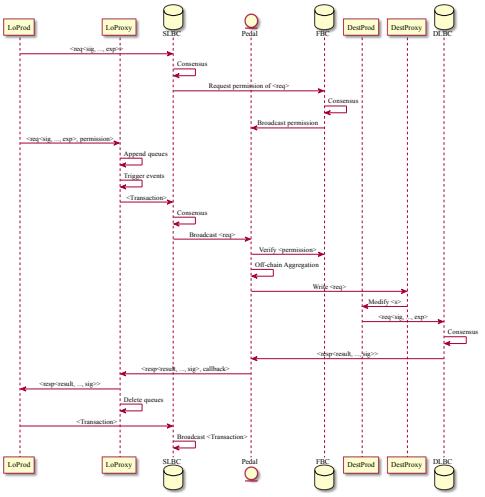


Fig. 3: Cross-chain Collaboration

B. Data Collaboration Mechanism with Data Oracle

Data collaboration includes on-chain and off-chain collaboration, and cross-chain collaboration. Data collaboration provides SewingChain with real-time data from the outside industrial world and isolated data from other blockchains.

1) Initialize: We exploit permissioned blockchain as the basic technology to restrict participants' identities. Industrial equipment, gateways, and edge servers use elliptic curve digital signature algorithms and asymmetric cryptography algorithms to create unique identities in the SewingChain. Entities need to pass the authentication of permissioned blockchain. The legal identity on the blockchain includes account address, public key, private key, and certificate, denoted as $Addr_{e_i}, PK_{e_i}, SK_{e_i}, Cert_{e_i}$. $Addr_{e_i}$ is a string calculated by PK_{e_i} through cryptographic algorithms with collision-resistance. SK_{e_i} is used for signature to uniquely map entities' ownership of $Addr_{e_i}$. $Cert_{e_i}$ is used for authentication.

2) Data Collaboration:

a) On-chain and Off-chain Collaboration: It is necessary to request real-time off-chain data in IIoT. However, LBC and FBC are limited by blockchain, while they can not actively obtain real-time data from the industrial world. Therefore, we use data oracle to achieve on-chain and off-chain data collaboration. The workflow is implemented in the Pedal Layer, as shown in Fig. 2.

The workflow includes entities that require off-chain industrial data, cloud services that store source data, application contracts $LoProd$ that receive logic of requirements, proxy contracts $LoProxy$ that provide data interfaces, LBC, and data oracle.

The entity e_i sends a request of off-chain industrial data to $LoProd$. $LoProd$ calls the interface provided by $LoProxy$ and forwards the request to $LoProxy$, which contains data source d_s and public key PK_{e_i}, signature Sig_{e_i}, certificate $Cert_{e_i}$, callback address $addr_c$, timestamp t_{s_1}, expiration exp_t. The $addr_c$ is the contract address of $LoProd$ in LBC. The request can be denoted as

$$req = E_{PK_{e_i}}(Sig_{e_i}||Cert_{e_i}||d_s||callback_{addr}||t_{s_1}||exp_t) \tag{5}$$

$LoProxy$ first verifies identities of e_i and pushes req to the pending request queue. A data collaboration event will be triggered after the transaction is written to the LBC. When data oracle catches the event, nodes of data oracle obtain, verify and aggregate data from specific d_s. The aggregation process see details in Section IV-A. Finally, data oracle callbacks data to $LoProxy$. The callback message contains public key group $PK_{\sum_{i=1}^{n} e_i}$, callback address $addr_c$, timestamp t_{s_2} and aggregated signature Sig_a, can be denoted as

$$resp = E_{PK_{\sum_{i=1}^{n} e_i}}(data||addr_c||t_{s_2}||Sig_a) \tag{6}$$

$LoProxy$ verifies Sig_a of $resq$ and check whether t_{s_2} is less than exp_t. If all conditions are complete, $LoProxy$ callbacks $resp$ to $LoProd$ and delete req from the pending request queue. When LBC broadcasts transactions, e_i can get the data of the specified d_s.

b) Cross-chain Collaboration: The heterogeneous interconnection ecosystem makes LBCs hard to collaborate with other LBCs. Therefore, we use data oracle to achieve cross-chain collaboration. The workflow is implemented in the Pedal Layer, as shown in Fig. 3.

The workflow includes Source LBC (SLBC) for proposing requests, FBC for filing requests, data oracle, and Destination LBC (DLBC).

The entity e_i sends a cross-chain request to $LoProd$ of SLBC, while $LoProd$ calls cross-chain interfaces provided by $LoProxy$ on SLBC and forwards the request. The request contains the specified DLBC, $LoProd$ address $Addr_d$ in DLBC, data status s, public key PK_{e_i}, signature Sig_{e_i}, certificate $Cert_{e_i}$, callback address $addr_c$, timestamp t_{s_1}, expiration exp_t, can be denoted as

$$req_{on} = E_{PK_{e_i}}(Sig_{e_i}||Cert_{e_i}||s||DLBC||Addr_d||t_{s_1}||exp_t) \tag{7}$$

After $LoProxy$ obtains req_{on}, it verifies $Cert_{e_i}$ of e_i and adds req_{on} to the pending request queue. A data collaboration event will be triggered after the transaction is written to the LBC. After L_f selected by random methods puts on record to

FBC, it writes the merkel root $root_m$ of the filed transaction into SLBC's $LoProxy$. Data oracle catches the events and verifies whether $root_m$ exists in the FBC. Subsequently, data oracle aggregates, consensus, and verifies events. The aggregation process see details in Section IV-A. Finally, it writes transactions into $LoProxy$ of DLBC, which can be denoted as

$$req_{off} = E_{PK_{\sum_{i=1}^{n} e_i}}(Sig_a||s||DLBC||Addr_d||t_{s_2}||exp_t) \tag{8}$$

DLBC's $LoProxy$ first verifies Sig_a and check whether t_{s_2} is less than exp_t. $LoProxy$ calls $LoProd$ specified by $Addr_d$ to modify s and trigger callback event. Data oracle catches event and call SLBC's $LoProxy$ to execute the callback operation. The content of the callback can be denoted as

$$resp = E_{PK_{\sum_{i=1}^{n} e_i}}(result||addr_c||t_{s_3}||Sig_a) \tag{9}$$

After SLBC's $LoProxy$ verifies Sig_a and $resp$, it calls $LoProd$ to return the result, and removes req_{on} from the pending request queue.

C. Computing Collaboration Mechanism with Computing Oracle

This section introduces the computing collaboration mechanism used by entities to outsource computing tasks. Computing oracle combines federated learning to enable nodes of Pedal to complete tasks. Nodes may do evil in the process of outsourcing. First, the server may deliver random parameters. Xianglong Z. [26] proposed a privacy-preserving and verifiable federated learning scheme to prevent malicious aggregation nodes. Second, clients may upload random parameters. Our paper focuses on the second situation and proposes an aggregation method for learning inspired by [25], as described in Algorithm 2. We assume that the number of malicious nodes m does not exceed $\frac{1}{3}$ [35] of the total number of nodes n in the network.

a) Release: Entities write computing requirements into req of $LoProd$. The requirement contains dataset D, storage location url, communication rounds r, iteration rounds e, learning rate η, batch size B, seed, and other necessary training parameters. Computing oracle calculates optimal model parameters according to (10). Then req is forwarded to $LoProxy$ to trigger computing events monitored by Pedal's master node.

$$\min_{w \in \mathbb{R}^d} f(w) \text{ where } f(w) \stackrel{def}{=} \frac{1}{n} \sum_{i=1}^{n} f_i(w) \tag{10}$$

b) Local Update: The master node of pedal broadcasts req to other nodes. Nodes obtain D_i according to url after receiving tasks. And they train models to get gradient w_i layer by layer according to $(r, e, \eta, ..., B)$, as shown in (11). After that, they send w_i to the master node to perform aggregation.

$$w_i(r) = w_i(r-1) - \eta \nabla f(w_i(r-1)) \tag{11}$$

Algorithm 2 The Decentralized Aggregation for Learning

function RELEASE($tasks$) ▷Run on SewingChain
  $master \leftarrow$ ELECTION(α)
  for each $task \in tasks$ **in parallel do**
    $(D, r, ..., \eta) \rightarrow TaskQueue[task]$
    Trigger events $\rightarrow$ LOCALUPDATE(args)
  end for
end function

function LOCALUPATE($args$) ▷Run on Pedal
  for each $node \in nodes$ **in parallel do**
    for $e_i \in e$ **do**
      $w_i^r \leftarrow w_i^{r-1} - \eta \bigtriangledown f(w; b)$
    end for
    AGGREGATION(r, w_i^r)
  end for
end function

function AGGREGATION(r, w_i^r) ▷Run on Pedal
  Receive w_i from n nodes
  Test w_i with $\frac{1}{3} D_{test}$
  $w_s \leftarrow$ Sort w_i according to acc_i
  Aggegregate w_g as equation 13
  Write $Digest(w_g)$ into LBC
end function

function GLOBALUPDATE($r, w_{g'}$) ▷Run on Pedal
  if $Digest(w_{g'}) == Digest(w_g)$ **then**
    for each $node \in nodes$ **in parallel do**
      $w \leftarrow w - \eta \bigtriangledown f(w; b)$
    end for
  end if
end function

c) Aggregation: The master node does not directly aggregate w_i like FedAvg [24] after it receives w_i from all nodes. And it is difficult to check the effectiveness of w_i since D_i of each node is different. Nodes may upload w_i from training or randoming. Therefore, the master node needs to use $\alpha = \frac{1}{3}$ test set to sort w_i, as described in (12). The α is determined by empirical values. And w_g is aggregated from the top $\beta = \frac{1}{3}$ w_i, as described in 13. Subsequently, the master node delivers w_g to other nodes. At the same time, the master node needs to write the digest of w_g into LBC to prevent it from assigning different parameters.

$$w_s = [w_i^{(1)}, w_i^{(2)}, w_i^{(3)}, ..., w_i^{(n)}] \tag{12}$$

$$w_g = \frac{\sum_{j=1}^{\frac{1}{3}n} |D_i| * w_i^{(j)}}{|D|} \tag{13}$$

d) Global Update: Each node verifies whether the digest of $w_{g'}$ is equal to that of w_g in LBC. After that, they update local models according to w_g.

V. Performance Analysis

This section evaluates the performance of the SewingChain. First, we conduct a security analysis of Pedal. Second, we evaluate the efficiency of decentralized collaboration. Third, we evaluate the efficiency of the decentralized aggregation for learning.

A. Security Analysis

1) Security of Decentralized Collaboration: The attacker may pretend to be the master node to initiate interactive requests to $Pedal$ so that it can gain the trust of nodes and decentralized collaboration. However, it does not possess the private key sk_{v_i} of the master node, which can not generate the valid signature β_i and proof (d_i, π_i) and fail to obtain trust from other nodes, as shown in (14) and (15).

$$(pk_{v_i}, \alpha, \beta_i, \pi_i) \neq (pk_{v_i}, \alpha, \beta_a, \pi_a) \rightarrow invalid \tag{14}$$

$$(d_i, \tau) \neq (d_a, \tau) \rightarrow invalid \tag{15}$$

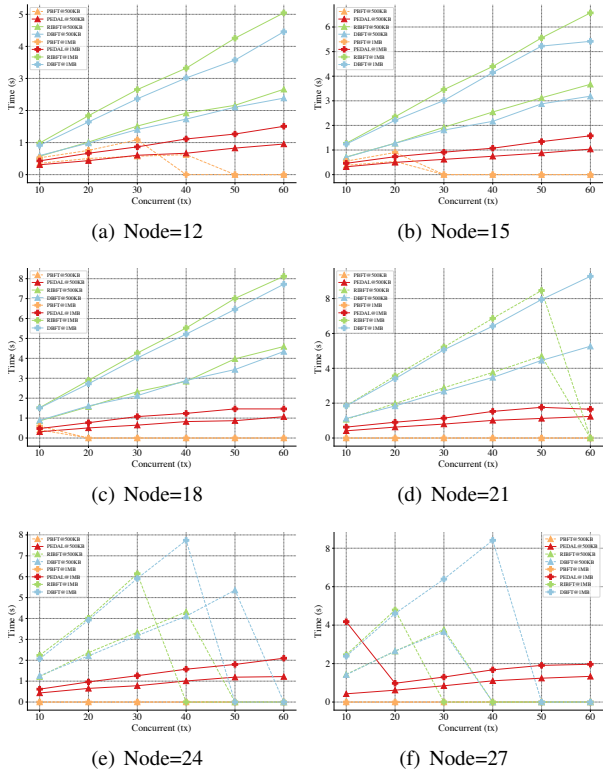
The malicious master node may callback wrong results. It means that it does not aggregate results from other nodes and returns random results to blockchain. However, results from the malicious node do not attach aggregated signatures σ from other nodes, which can not pass the verification of $LoProxy$, as shown in (16).

$$e(g_1, \sigma_b) \neq e(g_1, , \prod_{i=1}^{k} (h(req_i))^x) = e(g_1^x, h(req)) \tag{16}$$

The malicious attacker may repeatedly forward previous computing tasks to waste precious computing resources. However, each task is published in $LoProxy$ while other nodes can reject replay attacks by monitoring the states of $LoProxy$.

The attacker may intentionally initiate a large number of meaningless computing tasks through $LoProxy$ to occupy computing resources. It needs to combine with incentive mechanism. Participants of the proposed mechanism needs compulsively deposit some stakes. Then they have the access to occupy computing resources to carry out subsequent tasks.

2) Security of Decentralized Learning: Malicious nodes may upload random results to the master node while the number of malicious nodes does not exceed $\frac{1}{3}$ of the total number of nodes in the network. There are two situations. First, when random results of malicious nodes perform worse than that of honest nodes. Therefore, results from malicious nodes can not be aggregated according to (13). Second, random results perform better than hard workers. The proposed mechanism can directly use them for training models in the next rounds while not considering whether it would affect the convergence. The reason is that more local epochs can alleviate the performance degradation caused by malicious clients in the learning system [36]. With the help of at least $\frac{2}{3}$ honest nodes, the training task will finally converge.



(a) Node=12 (b) Node=15

(c) Node=18 (d) Node=21

(e) Node=24 (f) Node=27

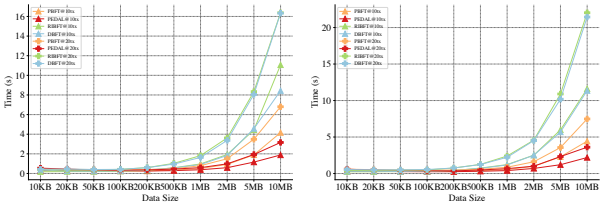
Fig. 4: Collaboration time of different concurrent transactions

B. Decentralized Collaboration

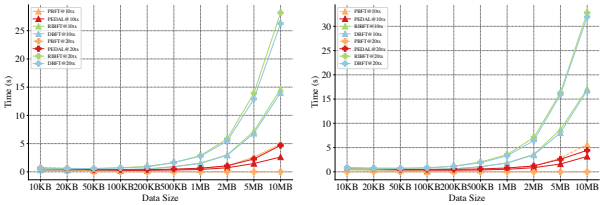
We simulated a digital twin network system based on blockchain and oracle to provide tamper-proof services, data collaboration, and computing collaboration in go1.17 linux/amd64. It is deployed on Ubuntu 16.04.7 LTS, Intel(R) Xeon(R) CPU E5-2620v4@2.10GHz with 8 core, 64G memory and 1000Mb/s. The collaborative part of SewingChain (denoted as PEDAL) is compared with DBFT[12], RIBFT[13], and PBFT[35] respectively. The number of collaborative groups for PEDAL, DBFT, and RIBFT is 3, and the number of nodes in each group can be specified ($>= 4$). The number of collaborative groups for PBFT is 1, and the total number of nodes for the four algorithms is the same.

Fig. 4 and Fig. 5 are the total number of nodes in the network with [12, 27, 3], and the number of nodes in each cooperative group is [4, 9, 1]. In two figures, red represents PEDAL, orange represents PBFT, green represents RIBFT, and blue represents DBFT. The dotted line means that the algorithm cannot crash under this condition. We set a different number of group nodes in the subgraph to show the performance of different algorithms as the network scale and concurrent transaction increase.

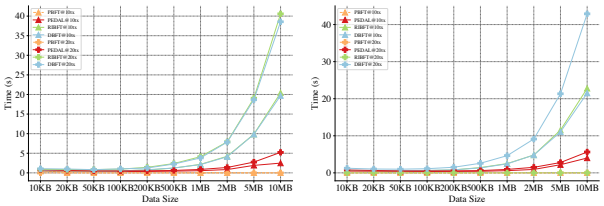
Fig. 4 is the processing time while concurrent transactions is [10, 60, 10] and data size are [500KB, 1MB]. We can see that PEDAL, represented by the solid red line, has the lowest transaction processing time when the network scale gradually increases. Moreover, PEDAL can maintain normal status while the network scale is 27 and the concurrent transactions are 60. It can be seen from Fig. 4 that some algorithms cannot



(a) Node=12 (b) Node=15

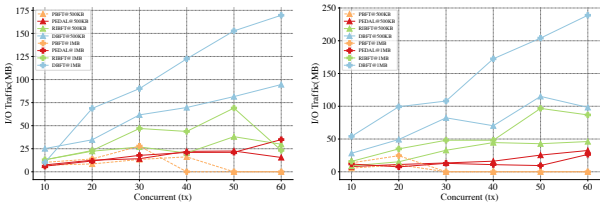


(c) Node=18 (d) Node=21

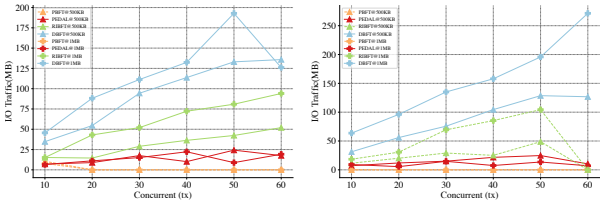


(e) Node=24 (f) Node=27

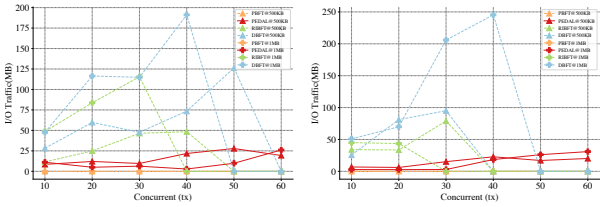
Fig. 5: Collaboration time of different data size



(a) Node=12 (b) Node=15



(c) Node=18 (d) Node=21



(e) Node=24 (f) Node=27

Fig. 6: I/O traffic of different concurrent transactions

withstand high concurrency as the network scale increases.

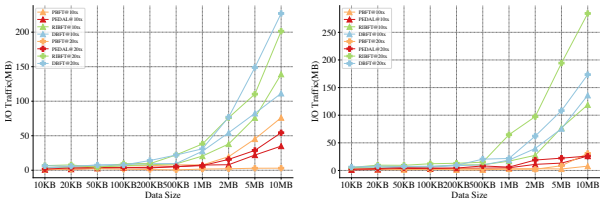
Fig. 5 is the processing time while data volume is different and the concurrent transaction is [10tx, 20tx]. It can be seen from the figure that PEDAL has the shortest processing time under different data volumes.

We also tested the I/O traffic of the collaborative network, as shown in Fig. 6 and Fig. 7. Those two figures are measured while the total number of network nodes is [12, 27, 3]. The reason why sets a different number of group nodes in the subgraph is the same as above.

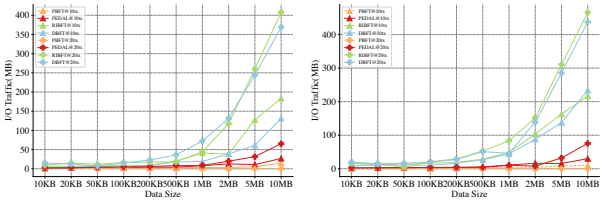
Fig. 6 is the I/O traffic while the concurrent transactions is [10, 60, 10] and the data size are [500KB, 1MB] respectively. It can be seen from the figure, as the concurrent transaction volume increases in the PEDAL network, the overall network traffic remains stable, and the overall traffic cost is less than the other algorithms.

Fig. 7 is the I/O traffic while data volume is different and the concurrent transaction is [10tx, 20tx]. It can be seen from the figure that with the increase of data volume in the PEDAL network, the trend of network traffic is smaller than that of the other algorithms, and the overall traffic cost is less than that of the other algorithms.

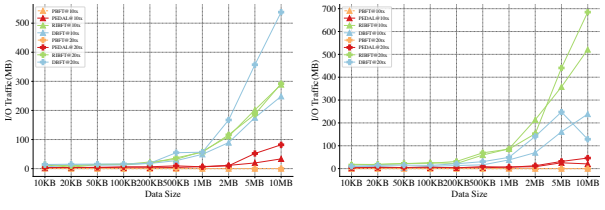
It can be seen from Fig. 4-7 that the compared algorithms crash down when node=27. Therefore, we only tested the performance of the proposed algorithm in a large-scale network. Fig. 8 (a) and (b) are the processing time when the network volume is [30, 99, 3] and the concurrent transaction is [10, 90, 10] with different data sizes. Fig. 8 (c) and (d) are the processing time when the network volume is [30, 99, 3] and the data volume is different with different concurrent



(a) Node=12 (b) Node=15



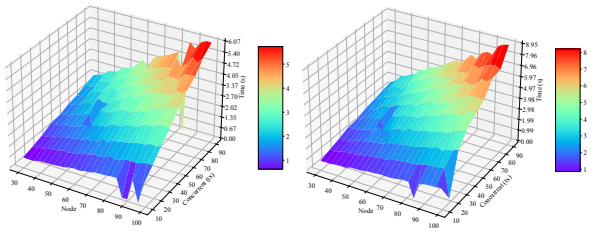
(c) Node=18 (d) Node=21



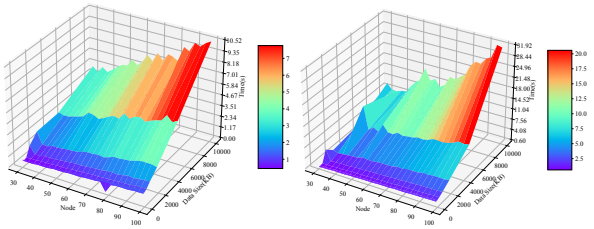
(e) Node=24 (f) Node=27

Fig. 7: I/O traffic of different data size

transactions. It can be seen from the figure that PEDAL has good performance when the number of nodes and concurrent transactions increase.



(a) DataSize=500KB (b) DataSize=1MB



(c) Concurrent=10tx (d) Concurrent=20tx

Fig. 8: Collaboration time of different concurrent transactions and data size in the large-scale network

C. Decentralized Learning

In this section, we implemented the proposed approach in XuperChain[1], PyTorch 1.7.0 and go1.17 linux/amd64. All experiments are performed on a server with Ubuntu 16.04.7 LTS, Intel(R) Xeon(R) CPU E5-2620v4@2.10GHz with 8 core, 64G memory, and four NVIDIA RTX 3090 GPUs. The decentralized aggregation for learning of SewingChain (denoted as Pedal) is compared with proof of learning (denoted as Proof)[25] and FedAvg[24]. In the experiment, the number of nodes participating in the calculation is 100, and the active nodes are 10 [24]. The iteration epoch of client nodes is 5, and the communication round is 30. We use MNIST[2], Fashion-MNIST[3], CIFAR-10[4] and KMNIST[5] as the primary dataset of the calculation task. Moreover, we set up IID and non-IID cases based on the above datasets. IID is constructed by randomly extracting the same amount of data from the dataset of clients. Non-IID is constructed by sorting all training sets according to labels, dividing them into 200 slices. And clients randomly select the same amount of data from slices.

We use a convolutional neural network as a local training model. Since the complexity of MNIST and KMINIST are different from that of Fashion-MNIST and CIFAR-10, we use different calculation graphs of CNN to train three datasets separately, as described in TABLE I, TABLE II and TABLE III. The experiment examines the collaboration efficiency of different computing paradigms by changing the proportion of malicious nodes. Moreover, we repeated 3 times experiments and took the average value as the final results.

It should be noted that Proof selects the nodes with the claimed smallest loss as the consensus node and delegates the parameters to other nodes. Other nodes will verify whether the claimed loss is true by re-executing tasks of that node. When malicious nodes generate a loss randomly, other nodes will not update local models. Therefore, we choose to aggregate parameters of honest nodes as the accuracy of Proof.

[1]https://github.com/xuperchain/xuperchain
[2]http://yann.lecun.com/exdb/mnist/
[3]https://github.com/zalandoresearch/fashion-mnist
[4]https://www.cs.toronto.edu/ kriz/cifar.html
[5]https://github.com/rois-codh/kmnist

TABLE I: Architecture of CNN in MINIST and KMNIST

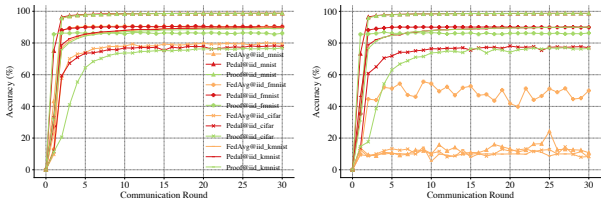
Layer	Shapes	Layer	Shapes
Conv2d_1	1 x 10 x 5	Maxpool	2
Maxpool	2	ReLu	
ReLu		Linear	320 x 10
Conv2d_2	10 x 20 x 5		

TABLE II: Architecture of CNN in Fashion-MINIST

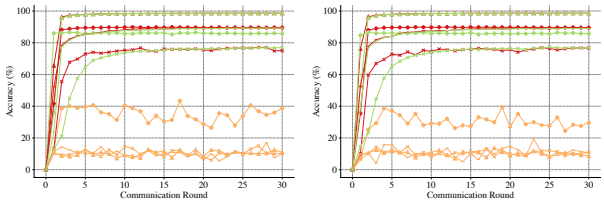
Layer	Shapes	Layer	Shapes
Conv2d_1	1 x 16 x 5 x 2	BatchNorm2d	32
BatchNorm2d	2	ReLu	
ReLu		Maxpool	2
Maxpool	2	Linear	7 x 7 x 32 x 10
Conv2d_2	16 x 32 x 5 x 2		

TABLE III: Architecture of CNN in CIFAR-10

Layer	Shapes	Layer	Shapes
Conv2d_1	3 x 64 x 3 x 1	Conv2d_5	3 x 512 x 3 x 1
MaxPool	2	Conv2d_6	3 x 512 x 3 x 1
Conv2d_2	3 x 128 x 3 x 1	MaxPool	2
MaxPool	2	Conv2d_7	3 x 512 x 3 x 1
Conv2d_3	3 x 256 x 3 x 1	Conv2d_8	3 x 512 x 3 x 1
Conv2d_4	3 x 256 x 3 x 1	MaxPool	2
MaxPool	2	Linear	512 x 10



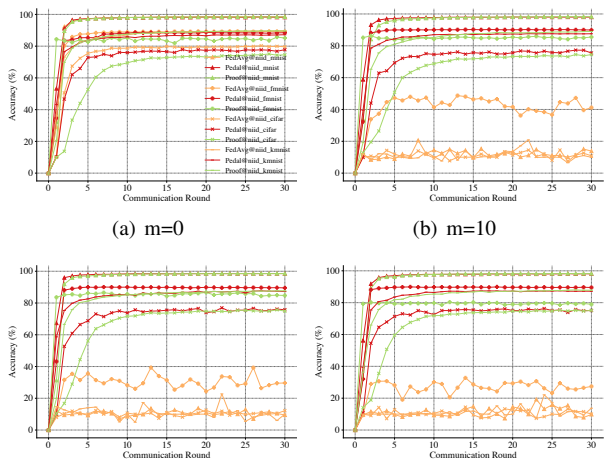
(a) m=0 (b) m=10



(c) m=20 (d) m=30

Fig. 9: Accuracy of IID dataset

Fig. 9 and Fig. 10 indicate different number of malicious nodes. It can be seen from the figure that when the number of malicious nodes is 0, the accuracy of Pedal and Proof are inferior to that of FedAvg in the IID and Non-IID of Fashion-MNIST, CIFAR-10 and KMNIST. However, as the proportion of malicious nodes increases, the accuracy of Pedal and Proof far exceeds FedAvg. At the same time, Pedal and Proof have



(a) m=0 (b) m=10

(c) m=20 (d) m=30

Fig. 10: Accuracy of Non-IID dataset

similar accuracy in the IID and Non-IID datasets of MNIST and KMNIST. Furthermore, the accuracy of Pedal is higher than that of Proof in the IID and Non-IID datasets of Fashion-MNIST and CIFAR-10. It can be considered that Pedal has better generalization ability in complex datasets.

VI. Conclusion

The new intelligent IIoT machine requires frequent data exchange between physical and digital entities, bringing severe challenges to the traditional centralized architecture. The decentralized industrial architecture is imminent. Blockchain is a decentralized trust machine. However, the decentralized architecture based on blockchain and digital twins cannot handle vast and complex data collaboration and computing collaboration. Therefore, we propose a novel architecture combining oracle with decentralized learning for IIoT. We also propose an effective decentralized collaboration mechanism to support trusted data sharing and resource exchanges. Moreover, we propose a novel computing mechanism to expand the learning capabilities of the industrial ecology. In the future, we will further study the blockchain-based computing collaboration mechanism and find a way to integrate blockchain and AI deeply. Moreover, further adaptation will be made to blockchain and oracle for IIoT. An industrial computing mechanism will come up that can carry large amounts of data, high concurrency, and low consumption.

References

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21260, 2008.

[2] J. Leng, D. Yan, Q. Liu, K. Xu, J. L. Zhao, R. Shi, L. Wei, D. Zhang, and X. Chen, "Manuchain: Combining permissioned blockchain with a holistic optimization model as bi-level intelligence for smart manufacturing," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 1, pp. 182–192, 2019.

[3] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Communication-efficient federated learning and permissioned blockchain for digital twin edge networks," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2276–2288, 2020.

[4] M. Swan, *Blockchain: Blueprint for a new economy*. " O'Reilly Media, Inc.", 2015.

[5] N. Szabo, "Formalizing and securing relationships on public networks," *First monday*, 1997.

[6] V. Buterin *et al.*, "A next-generation smart contract and decentralized application platform," *white paper*, vol. 3, no. 37, 2014.

[7] Y. Lin, Z. Gao, K. Xiao, Q. Wang, Z. Mo, Y. Yang, L. Rui, H. Guo, and D. Wang, "A model training mechanism based on onchain and offchain collaboration for edge computing," in *ICC 2021-IEEE International Conference on Communications*. IEEE, 2021, pp. 1–6.

[8] M. Zhaofeng, W. Xiaochang, D. K. Jain, H. Khan, G. Hongmin, and W. Zhen, "A blockchain-based trusted data management scheme in edge computing," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 2013–2021, 2019.

[9] G. Sagirlar, B. Carminati, E. Ferrari, J. D. Sheehan, and E. Ragnoli, "Hybrid-iot: Hybrid blockchain architecture for internet of things-pow sub-blockchains," in *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*. IEEE, 2018, pp. 1007–1016.

[10] G. Wood, "Polkadot: Vision for a heterogeneous multi-chain framework," *White Paper*, vol. 21, 2016.

[11] J. Kwon and E. Buchman, "Cosmos whitepaper," 2019.

[12] W. Li, C. Feng, L. Zhang, H. Xu, B. Cao, and M. A. Imran, "A scalable multi-layer pbft consensus for blockchain," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 5, pp. 1146–1160, 2020.

[13] S. Guo, F. Wang, N. Zhang, F. Qi, and X. Qiu, "Master-slave chain based trusted cross-domain authentication mechanism in iot," *Journal of Network and Computer Applications*, vol. 172, p. 102812, 2020.

[14] K. Mammadzada, "Blockchain oracles," 2019.

[15] V. Buterin, "Chain interoperability," *R3 Research Paper*, 2016.

[16] S. Ellis, A. Juels, and S. Nazarov, "Chainlink: A decentralized oracle network," *Retrieved March*, vol. 11, p. 2018, 2017.

[17] D. N. Team, "A decentralized oracle service boosting blockchain usability with off-chain data and verifiable computing power," https://s3.amazonaws.com/whitepaper.dos/DOS+Network+Technical+Whitepaper.pdf, 2019.

[18] J. Adler, R. Berryhill, A. Veneris, Z. Poulos, N. Veira, and A. Kastania, "Astraea: A decentralized blockchain oracle," in *2018 IEEE international conference on internet of things (IThings) and IEEE green computing and communications (GreenCom) and IEEE cyber, physical and social computing (CPSCom) and IEEE smart data*

(SmartData). IEEE, 2018, pp. 1145–1152.
[19] N. Protocol, "Nest protocol: A distributed price oracle network," https://nestprotocol.org/doc/ennestwhitepaper.pdf, 2020.
[20] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE internet of things journal*, vol. 3, no. 5, pp. 637–646, 2016.
[21] K. Xiao, W. Shi, Z. Gao, C. Yao, and X. Qiu, "Daer: A resource preallocation algorithm of edge computing server by using blockchain in intelligent driving," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9291–9302, 2020.
[22] Y. Dai, D. Xu, K. Zhang, S. Maharjan, and Y. Zhang, "Deep reinforcement learning and permissioned blockchain for content caching in vehicular edge computing and networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 4, pp. 4312–4324, 2020.
[23] S. Warnat-Herresthal, H. Schultze, K. L. Shastry, S. Manamohan, S. Mukherjee, V. Garg, R. Sarveswara, K. Händler, P. Pickkers, N. A. Aziz *et al.*, "Swarm learning for decentralized and confidential clinical machine learning," *Nature*, vol. 594, no. 7862, pp. 265–270, 2021.
[24] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.
[25] C. Qiu, H. Yao, X. Wang, N. Zhang, F. R. Yu, and D. Niyato, "Ai-chain: Blockchain energized edge intelligence for beyond 5g networks," *IEEE Network*, vol. 34, no. 6, pp. 62–69, 2020.
[26] X. Zhang, A. Fu, H. Wang, C. Zhou, and Z. Chen, "A privacy-preserving and verifiable federated learning scheme," in *ICC 2020-2020 IEEE International Conference on Communications (ICC)*. IEEE, 2020, pp. 1–6.
[27] G. Fortino, W. Russo, C. Savaglio, W. Shen, and M. Zhou, "Agent-oriented cooperative smart objects: From iot system design to implementation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 48, no. 11, pp. 1939–1956, 2017.
[28] M. Aazam, S. Zeadally, and K. A. Harras, "Deploying fog computing in industrial internet of things and industry 4.0," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4674–4682, 2018.
[29] S. K. Mishra, D. Puthal, J. J. Rodrigues, B. Sahoo, and E. Dutkiewicz, "Sustainable service allocation using a metaheuristic technique in a fog server for industrial applications," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4497–4506, 2018.
[30] M. Liu, S. Fang, H. Dong, and C. Xu, "Review of digital twin about concepts, technologies, and industrial applications," *Journal of Manufacturing Systems*, vol. 58, pp. 346–361, 2021.
[31] F. Tao and Q. Qi, "Make more digital twins," 2019.
[32] J. Wan, J. Li, M. Imran, D. Li *et al.*, "A blockchain-based solution for enhancing security and privacy in smart factory," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3652–3660, 2019.
[33] S. Micali, M. Rabin, and S. Vadhan, "Verifiable random functions," in *40th annual symposium on foundations of computer science (cat. No. 99CB37039)*. IEEE, 1999, pp. 120–130.
[34] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the weil pairing," in *International conference on the theory and application of cryptology and information security*. Springer, 2001, pp. 514–532.
[35] M. Castro, B. Liskov *et al.*, "Practical byzantine fault tolerance," in *OSDI*, vol. 99, no. 1999, 1999, pp. 173–186.
[36] C. Ma, J. Li, M. Ding, H. H. Yang, F. Shu, T. Q. Quek, and H. V. Poor, "On safeguarding privacy and security in the framework of federated learning," *IEEE network*, vol. 34, no. 4, pp. 242–248, 2020.

Yijing Lin Yijing Lin received the bachelor's degree from the North China Electric Power University (NCEPU). She is a Ph.D. student at the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications (BUPT), Beijing, China. Her current research interests include blockchain and edge computing.

Zhipeng Gao Zhipeng Gao received the Ph.D. degree from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2007. He is currently a Professor with the State Key Laboratory of Networking and Switching Technology. His research interests are edge computing and blockchain. He presides over a series of key research projects on network and service management, including the projects supported by the National Natural Science Foundation and the National High-Tech Research and Development Program of China.

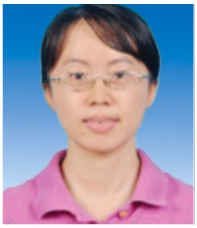
Weisong Shi received the B.S. degree from Xidian University, Xi'an, China, in 1995, and the Ph.D. degree from the Chinese Academy of Sciences, in 2000, both in computer engineering. Weisong Shi is a Charles H. Gershenson Distinguished Faculty Fellow and a Professor of Computer Science with Wayne State University, USA, where he directs the Mobile and Internet Systems Laboratory (MIST) and Connected and Autonomous dRiving Laboratory (CAR), investigating performance, reliability, power and energy-efficiency, trust and privacy issues of networked computer systems, and applications. He is one of the world leaders in the edge computing research community and published the first book on edge computing. His paper entitled "Edge Computing: Vision and Challenges" has been cited more than 1700 times. In 2018, Dr. Shi led the development of IEEE Course on Edge Computing. In 2019, Dr. Shi served as the lead guest editor for the edge computing special issue on the prestigious Proceedings of the IEEE journal. He is the Founding Steering Committee Chair of the ACM/IEEE Symposium on Edge Computing (SEC) and the IEEE/ACM Connected Health: Applications, Systems and Engineering (CHASE). He is an IEEE Fellow and an ACM Distinguished Scientist.

Qian Wang Qian Wang received the Ph.D. degree from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2020. She is currently a teacher with Beijing University of Technology. Her current research interests include wireless networks and opportunistic communications.

Huangqi Li Huangqi Li received B.Eng. degree at Beijing University of Posts and Telecommunications (BUPT), And he is currently pursing his MA.Sc. degree in the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications. His main research interests include blockchain and edge computing.

Miaomiao Wang Miaomiao Wang received M.S. degree from Beijing University of Posts and Telecommunications, Beijing, China, in 2017. She is currently pursuing the Ph.D. degree in Computer Science and Technology at Beijing University of Posts and Telecommunications. Her research interests include blockchain and network management.

Yang Yang Yang Yang received the Ph.D. degree from the Beijing University of Posts and Telecommunications (BUPT). She is currently an Associate Professor with the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications. Her current research interests are cooperation and management in the MANETs.



Lanlan Rui Lanlan Rui received the Ph.D. degree in computer science technology from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2010. She is currently an Associate Professor with the State Key Laboratory of Networking and Switching Technology, BUPT, China. Her research interests include IOT, MEC, content-based measurement and analysis, quality of service (QoS) and intelligent theory, and technology of network services. As a result of the standardization in network management, she received the 3GPP SA5 Outstanding Contribution Award.