Edge In-Network Computing Meets Blockchain: A Multi-Domain Heterogeneous Resource Trust Management Architecture

Linna Ruan, Shaoyong Guo, Xuesong Qiu, Luoming Meng, Shuang Wu, and Rajkumar Buyya

Abstract

In-network computing indicates the convergence of computing and network, which can be further considered as computing-based networking and network-enabled computing, from which comes the edge in-network computing paradigm. In this newly proposed paradigm, the computing, caching, and communication resources are deployed on in-transit edge nodes, such as routers or switches; hence, services are provided with less delay and energy consumption by the deep aggregation of edge-side resources. However, open challenges still exist in dealing with heterogeneous resource sharing, scheduling, and ensuring credibility of service processing. Given this background, this article aims to address the edge-side heterogeneous resource trust management in multi-domain scenarios. A multi-domain heterogeneous resource trust management architecture is first constructed to achieve trusted resource sharing and transaction. Then, under the proposed architecture, a smart-contract-based resource sharing mechanism is formulated, including an incentive mechanism based on end-side node contribution. Furthermore, a differential-evolution-enabled containerized microservice orchestration algorithm is finally designed to minimize operation delay and load imbalance during the container deployment. The application of this platform in a highly elastic power grid scenario has been tested as an example.

Introduction

In-network computing achieves higher attention in both the network and computing areas, where network devices are endowed with computing capability, compared to only being in charge of transferring data in the traditional form [1]. The primary force of this change is to integrate resources within the network to reduce the cost of task processing. Following this trend, the edge in-network computing paradigm has emerged and is increasingly adopted. In contrast to in-network computing, it specifically refers to resource integration at the edge, which means integration of the computing, caching, and communication resources between terminals and the remote cloud, providing services with less delay and energy consumption. This paradigm makes multi-domain heterogeneous resource management possible and provides a solution for edge nodes' cooperation. It is evolving with the development of 5G and 6G and has become a promising trend for decentralized latency-sensitive applications. However, as the edge nodes are geographically distributed and have limited capabilities, the collaborative management of edge-side resources mainly faces two challenges:

- The trust assurance of multi-domain resource integration
- The overall schedule of heterogeneous and geographically distributed edge resources

Luckily, cutting-edge technologies such as blockchain can help meet these challenges, although there are still limitations in the related research.

For the first challenge, blockchain is integrated in edge computing, to ensure trustworthiness and traceability. In [2], J. Feng *et al*. used blockchain to establish a distributed, tamper-proof digital ledger to ensure data security, and optimized the design of blockchain and mobile edge computing (MEC) collaboratively. A blockchain-based mobile edge computing (B-MEC) framework for adaptive resource allocation and computation offloading in wireless networks is proposed in [3], using blockchain to provide management and control functions, and smart contract to execute computation tasks automatically. Similarly, [4] proposed a simplified but effective framework named FogBus for facilitating end-to-end IoT-fog (edge)-cloud integration. However, most of the work only focus on information trustworthiness, ignoring the trust issues in the resource allocation process.

For the second challenge, there are some ongoing research works on the design of blockchain-based resource management framework. In [5], a blockchain-based trusted decentralized resource management framework was proposed to solve the problem of energy consumption in data centers. In [6], Y. He *et al*. proposed a general framework for IoT scenarios. In addition, they realized the machine learning algorithm A3C with a smart contract in a private blockchain network. Taking incentives and cross-server resource allocation into consideration, [7] employed blockchain technology to maintain a tamper-proof ledger database. The resource allocation procedure was executed as a smart contract in blockchain, ensuring secure, fair cross-server resource allocation. In [8], a blockchain-enabled decentralized self-organized trading platform for Industrial Internet

Digital Object Identifier: 10.1109/MNET.110.2100029

Linna Ruan, Shaoyong Guo (corresponding author), Xuesong Qiu, and Luoming Meng (corresponding author) are with the Beijing University of Posts and Telecommunications; Shuang Wu is with the Ningxia Xintong Network Technology Co., Ltd; Rajkumar Buyya is with the University of Melbourne.

0890-8044/21/$25.00 © 2021 IEEE

of Things (IIoT) devices was constructed. Moreover, it adopted Stackelberg game to describe the interaction between the cloud provider and miners. However, these works omitted the challenge brought by the heterogeneity of resources. Such methods and metrics are adopted to address this problem [9, 10]. In [9], the elastic regression tree is combined with sample information to allocate heterogeneous computing resources. Reference [10] utilized data fusion, combined with non-orthogonal multiple access (NOMA), successive interference cancellation (SIC), and beamforming techniques to allocate power and spectrum resources to heterogeneous devices. Besides these methods, virtual technology, widely referred to as network function virtualization (NFV), is also adopted to cope with resource heterogeneity [11–14]. In this way, the difference brought by varying dedicated hardware can be regarded, and the end-edge resource is integrated as a resource pool. However, these works have not taken the trust challenge into consideration.

In accordance with the discussions above, the existing work to a great extent might be able to figure out one of the two challenges at once. Compared to these, this article discusses both the aforementioned challenges in the context of edge in-network computing meeting blockchain, realizes deep aggregation of network resources, and provides an architecture to guide multi-domain heterogeneous resource sharing and scheduling. The main contributions of this article therefore can be considered as follows:

- A multi-domain heterogeneous resource trust management (MHRTM) architecture is constructed where consortium blockchain is adopted to ensure the information trustworthiness as well as transaction traceability, and virtual technology supports multi-domain resource management. To the best of our knowledge, this is the first architecture that addresses heterogeneous resource trust management under the background of blockchain meeting edge in-network computing.
- A smart-contract-based incentive mechanism for edge-side resource sharing is formulated. Terminals are encouraged to share resources under the condition of fair revenue distribution. Honest information submission is guaranteed by contribution-based reward. Furthermore, performance analysis on four aspects is included.
- A differential evolution (DE)-enabled containerized microservices orchestration algorithm is designed to provide decisions on microservices deployment while minimizing processing delay and load imbalance. As a result, resources are packed as containers with specific functions, and the heterogeneity can be regarded.

Architecture for the Trusted Edge Network

The MHRTM platform is shown in Fig. 1. It is divided into three layers: end-edge device layer, heterogeneous edge resource integration layer, and decentralized application layer. Blockchain is used to ensure the information credibility as well as transaction traceability, while the heterogeneous resource integration is implemented by virtual technology.

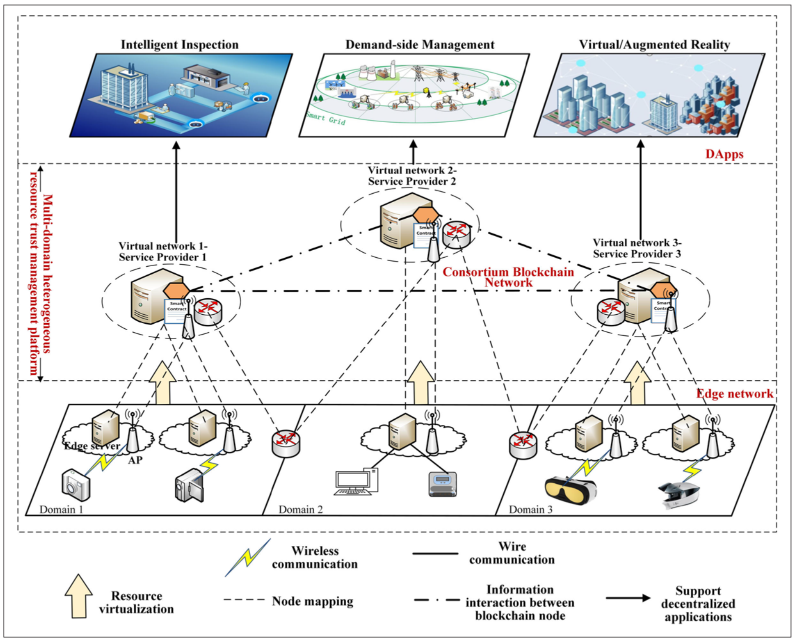
Intelligent Inspection
Demand-side Management
Smart Grid
Virtual/Augmented Reality
DApps
Multi-domain heterogeneous resource trust management platform
Virtual network 2-Service Provider 2
Virtual network 1-Service Provider 1
Virtual network 3-Service Provider 3
Smart Contract
Consortium Blockchain Network
Edge network
Edge server
AP
Domain 1
Domain 2
Domain 3
Wireless communication
Wire communication
Resource virtualization
Node mapping
Information interaction between blockchain node
Support decentralized applications

FIGURE 1. Multi-domain heterogeneous resource trust management (MHRTM) architecture.

In the designed edge in-network computing paradigm, each intra-domain device and inter-domain router are mapped as virtual nodes and form a virtual network with specific functions. These virtual networks are further abstracted as virtual nodes, where the function of blockchain is deployed. Considering scenarios with controllable access, the nodes are connected to form a consortium blockchain, which shows higher efficiency than the private chain and is more flexible than the public chain. Resource information of terminals, edge nodes, and network nodes, such as routers and service models, are uploaded to the blockchain to ensure the credibility and traceability. Furthermore, heterogeneous resource scheduling and benefit distribution are implemented on the basis of virtual technology and incentive mechanism.

Resource management is divided into resource sharing and resource scheduling processes, which are solved by a smart contract-based incentive mechanism and a DE-enabled containerized microservice orchestration algorithm in this article. Decentralized applications (DApps), such as intelligent inspection, demand-side management (DSM), and virtual/augmented reality are supported by the proposed resource management solution, which is carried out in the following sections.

The Workflow of the Resource Trust Management Process

The procedure of resource trust management is shown in Fig. 2. It can be divided into four parts: device registration, service request, service provision, and incentives for resource sharing. Consider a request submitted by a data service subscriber (DSS); the transaction procedure can be described as follows.

Device Registration

Edge network devices and end devices are abstracted as nodes and belong to different

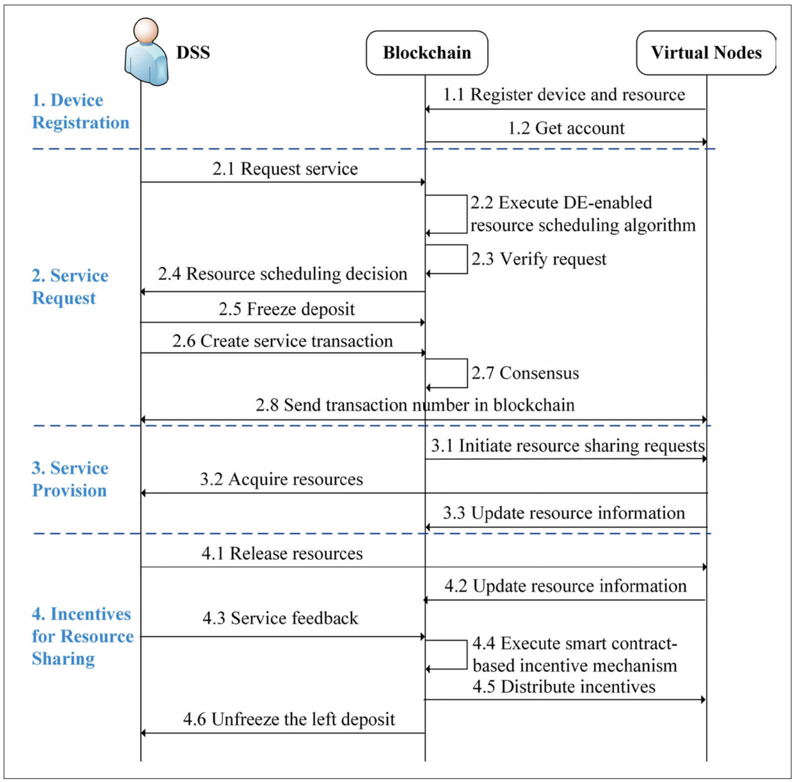
DSS
Blockchain
Virtual Nodes
1. Device Registration
1.1 Register device and resource
1.2 Get account
2.1 Request service
2.2 Execute DE-enabled resource scheduling algorithm
2.3 Verify request
2. Service Request
2.4 Resource scheduling decision
2.5 Freeze deposit
2.6 Create service transaction
2.7 Consensus
2.8 Send transaction number in blockchain
3.1 Initiate resource sharing requests
3. Service Provision
3.2 Acquire resources
3.3 Update resource information
4.1 Release resources
4.2 Update resource information
4. Incentives for Resource Sharing
4.3 Service feedback
4.4 Execute smart contract-based incentive mechanism
4.5 Distribute incentives
4.6 Unfreeze the left deposit

FIGURE 2. The procedure of resource trust management.

domains in the system. Through virtualization, the resources of nodes in a domain can be integrated, and these nodes are regarded as virtual nodes, where the functions of blockchain are deployed, such as participating in the consensus process, generating blocks, and maintaining the blockchain. End nodes participate in resource sharing through a smart-contract-based incentive mechanism, which is detailed later.

Back to the registration procedure. When a virtual node applies to join the system, it needs to upload its registration information to the blockchain, including the node ID, its location information and the amount of resources the nodes in the domain are willing to share. The shared resources in this article include computing, communication, and caching resources of devices. Furthermore, the computing resources refer to logical computing capability, parallel computing capability, and neural network acceleration capability.

After receiving the registration request from the virtual node, the smart contract will verify it. If the request is approved, the system will return a confirmation message (including the account information of the virtual node that initiated the request) and write the registration information to the blockchain in the form of transactions. Since the resources that nodes can share will be affected by their own needs, except for the fixed-time information submission required, the system allows nodes to actively submit their resource status.

In order to ensure normal operation of the system, after virtual nodes complete the registration of identities and resources, the system will give them a certain amount of digital currency as startup money. When nodes in the system need to share resources, they use the money to purchase resources. Since the currency issued can only be used in the system and cannot be exchanged for actual legal currency, a rational node will maintain the work of the system to protect its digital currency from losing value, enhancing the robustness of the system to some extent.

The registration of devices and resources is the basis for resource allocation. By writing the registration information on the blockchain, the traceability and tamper-proofing of the registration information are ensured. However, the authenticity of the registration information has not been guaranteed, which can be solved by the incentive mechanism based on the contribution described in the next section.

Service Request

When a DSS in the system initiates a resource sharing request, the system will verify and decide whether to approve it according to the price and account balance.

The specific steps are as follows.

Step 1: The DSS initiates a resource sharing request to the smart contract and submits the request information, including identity information of the DSS and the specific requirements.

Step 2: The system executes the DE-enabled containerized microservice orchestration algorithm to obtain the resource allocation decision. The resource scheduling algorithm will then guide tasks allocation according to the resources of each virtual node in the edge network, and the virtual nodes will accept them by default.

Step 3: The system calculates the price of this request according to the unit price of resources, and the amount and usage time of resources. Then the virtual node inquires about the balance of the DSS. If the balance is enough for this request, the resource sharing process continues.

Step 4: The system returns the resource allocation decision to the DSS.

Step 5: The system freezes part of the DSS's assets as the deposit of this process.

Step 6: The DSS records the information of this process as a transaction and submits it to blockchain.

Step 7: The system verifies trading information and conducts a consensus process.

Step 8: The system sends the transaction number to the DSS and virtual nodes for block propagation.

By executing the resource scheduling algorithm based on DE, the platform can provide personalized resource scheduling service for the DSS according to the resource request. The specific implementation of the algorithm is detailed below. According to the result of the resource scheduling algorithm, the platform calculates the price of this request and decides whether to provide resource scheduling service for the DSS according to its balance. According to the price, a portion of the DSS's funds is frozen, which means it still belongs to the DSS but is unavailable in the execution of the resource request, thereby guaranteeing the benefit of the resource provider.

Service Provision

Through the resource scheduling algorithm introduced below for resource allocation, virtual nodes are selected for resource sharing. The blockchain sends the resource sharing requests to the selected virtual nodes according to the scheduling decision. Through container creation and migration, the DSS acquires the resources it requests. Since the available resources of the selected virtual nodes change after each sharing, they upload and update the resource information to the blockchain.

As one of the core technologies of blockchain, the consensus mechanism ensures the consistency of the distributed ledger, which is the basis for the safe and reliable service provision in a blockchain network. In this article, scenarios with controllable access are considered; hence, we use the consortium blockchain to support the distributed heterogeneous resource trust management platform, and realize the multi-domain trusted scheduling and sharing of resources. As a simple and effective consensus algorithm, the Raft consensus algorithm is chosen to alleviate the computing and communication burden. Raft is a non-Byzantine fault-tolerant consensus algorithm, which is mainly implemented by leader election and log synchronization. The leader election is triggered by the heartbeat mechanism. Once a leader node is elected, the client's requests will be executed, and the follower nodes will copy logs safely until the next round begins. The maximum number of fault-tolerant failure nodes of the Raft algorithm is $(N - 1)/2$, where N is the total number of nodes in the cluster. Compared to consensus mechanisms such as proof of work (PoW) and proof of stake (PoS), the communication complexity of the Raft algorithm is $O(n)$, which greatly saves nodes' computing and communication resources.

Incentives for Resource Sharing

When a service is completed, the DSS releases the resources it acquired before, and the selected virtual nodes update their resource information again. The DSS sends service feedback to the blockchain, triggering the smart-contract-based incentive mechanism. Blockchain will distribute the incentives according to each virtual node's contribution and unfreeze the remaining deposit.

Smart-Contract-Based Incentive Mechanism for Edge-Side Resource Sharing

Edge-side resource virtualization is the basis for resource sharing and scheduling. Considering the limited capability of edge nodes, containers, which represent the operating system (OS)-level virtualization, are formulated instead of the hardware-level virtualization, acknowledged as virtual machines (VMs). The specific procedure is depicted in Fig. 3. As shown, the resources of terminals and edge devices in each domain are virtualized from computing and caching aspects. Then, following the resource scheduling decision made by edge nodes that received services, containers are created by the docker. Each container corresponds to a microservice. Afterward, the containers will migrate to the node that received the request, and after the service is completed, the smart-contract-based incentive mechanism for resource sharing is triggered and executed automatically.

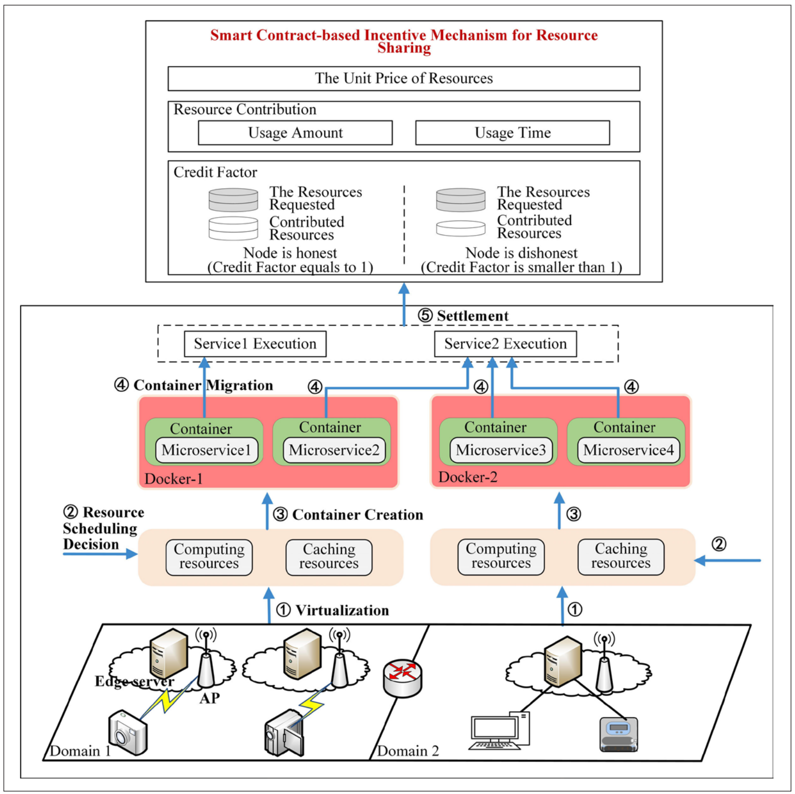

FIGURE 3. Container-based edge-side resource virtualization.

Edge-Side Resource Incentive Mechanism Based on Contribution

In this article, both the edge nodes and the end nodes are considered as selfish. Every node hopes to get higher profits by providing more resources, so their behaviors are similar and still regarded as one virtual node in the incentive mechanism to ease analysis. The smart contract will be triggered when the virtual node that uploads a resource sharing request submits service feedback to the blockchain, and the incentive mechanism will be automatically executed according to the evaluation results of nodes' contributions. In this article, we only consider the situation in which the total available resources in the system can meet the requirements of resource sharing requests.

When a resource sharing request is approved, the smart contract will call the resource scheduling algorithm to allocate resources and send a resource sharing request to the related nodes. Then the smart contract will reward every virtual node that participates in the sharing process. In order to reward nodes fairly and reasonably, an incentive mechanism based on nodes' contributions is proposed. The reward obtained by each node hinges on whether it is honest, the amount of resources it provides, and the usage time and unit price of resources, which can be calculated by multiplying the average selling price of resources by a credit factor defined as follows.

If a node is honest and can provide enough resources according to the resource sharing request made by the system, its credit factor is 1, and the system provides rewards to the node according to the average selling price of the resources. However, if a node is dishonest and cannot provide enough resources according to

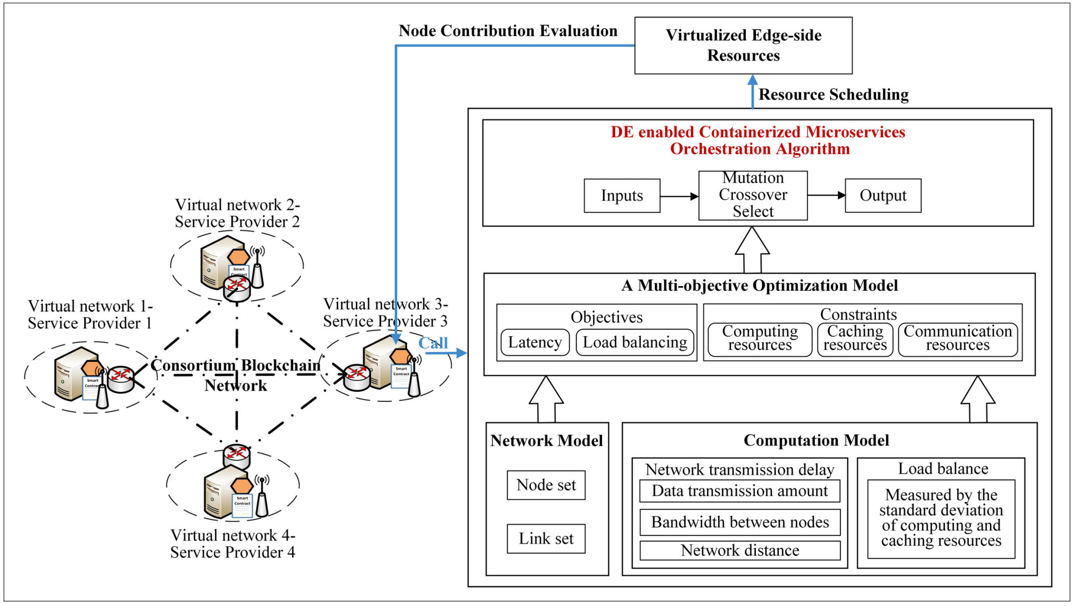
Node Contribution Evaluation
Virtualized Edge-side Resources
Resource Scheduling
DE enabled Containerized Microservices Orchestration Algorithm
Inputs
Mutation Crossover Select
Output
A Multi-objective Optimization Model
Objectives
Latency
Load balancing
Constraints
Computing resources
Caching resources
Communication resources
Network Model
Node set
Link set
Computation Model
Network transmission delay
Data transmission amount
Bandwidth between nodes
Network distance
Load balance
Measured by the standard deviation of computing and caching resources
Virtual network 1-Service Provider 1
Virtual network 2-Service Provider 2
Virtual network 3-Service Provider 3
Virtual network 4-Service Provider 4
Consortium Blockchain Network
Call

FIGURE 4. Edge-side containerized microservices orchestration architecture.

the requests issued by the system, its credit factor is smaller than 1, calculated by the ratio of the actual amount of resources provided by the node to the amount of resources it registered in the system. In this situation, the reward will be less. The unit price of resources is reached by all the parties before the resource allocation process. In order to ensure that a node can afford the request it initiated, the deposit that it needs to offer is calculated according to the average selling price of the required resources.

Performance Analysis

The Algorithm Satisfies the Truthfulness: Participants will not benefit from lying. In the resource scheduling strategy, we take the resource that a virtual node owns as a constraint rather than an optimization objective. Hence, the resource allocation strategy has no direct relationship with the resource amount owned by each node. Reporting more resource than actually owned will not increase the opportunity of a node to be selected in resource sharing.

We assume that the virtual nodes are rational but selfish, and they will not report less resources than they are willing to share. Under this assumption, the possible strategies adopted by virtual nodes can be divided into two types: honest report, where the amount of resources it registered is the same as the real value; and false report, where the amount of resources it registered is more than the real value.

In this article, resources are directly allocated to the requesting node through virtualization, and the system will issue incentives to each virtual node according to the resources it actually provides, so it is meaningless for virtual nodes to falsely report the amount of resources they are willing to provide. Moreover, the credit factor is added to the incentive mechanism. If a node reports falsely and fails to provide sufficient resources according to the requirements of the system, the reward will be less. It is obvious that deception shows no benefits; hence, as a rational node, there is no reason to make such false behavior.

The Algorithm Satisfies Individual Rationality: Each selected resource provider can obtain incentives. Although the amount of incentives that one node can get varies with the amount of resources it provides, the usage time of the resources, the unit price of resources, and whether it is honest, the incentives are always non-negative. Hence, it is always profitable for nodes to participate in resource sharing.

The Algorithm Satisfies Budget-Wise Feasibility: The price of a resource sharing request is related to the required amount, usage time, and unit price of resources. When a node submits a request, the smart contract will calculate the price of the request according to the specific requirements. If the account balance is lower than the budget, the resource request will end, and resource sharing operation will not be carried out. Since the budget is calculated under the assumption that each node is honest, which means the credit factor is set as 1, the actual payment will definitely not exceed the price.

The Algorithm Has High Computational Efficiency: The time complexity of the incentive mechanism is analyzed. Since the incentive obtained by each node is proportional to the amount of resources it shares and the usage time of the resources, the complexity of the incentive algorithm is $O(1)$ when only one virtual node participates in the resource sharing process. When n virtual nodes participate in the resource sharing process in the system, the operations above need to be performed n times, so the complexity of the incentive algorithm is $O(n)$.

DE-Enabled Containerized Microservice Orchestration Algorithm

Virtualization technology and a DE optimization algorithm are adopted to support container deployment determination, container creation, and container migration, as shown in Fig. 4. Consider the case when a request is submitted by DSS; the edge-side containerized microservices orchestration algorithm is then called to realize resource scheduling, which takes latency and load balance as objectives and computing, caching as well as communication resources as constraints. It is obviously an NP-hard problem, solved by DE as detailed below. The containers will be created and then migrate to the edge node that holds the request. The request will be finalized with the required containers, and the completion status of the task will finally be sent to the edge node for contribution evaluation.

Problem Formulation

For the sake of simplicity, we assume that each DSS only initiates one service request within a period of time. Services can be divided into microservices, which are carried by containers with specific functions. The service is described from two aspects: microservice set and dependency relationship between microservices. When one microservice needs the results or data of another microservice, a relationship is established between them. Microservices are directly initiated by users or requested by other users, and the resource information of microservices mainly includes computing and caching capabilities, in which the computing part can be further divided into logical computing capability, parallel computing capability, and neural network acceleration capability. The problem is established via three models described below.

The network model includes node set (computing and caching capabilities) and link set (bandwidth and network distance).

The computing model is used to evaluate load balance (measured by the standard deviation of computing and caching resources) and network transmission delay between microservices, which mainly depends on three factors:

- Data transmission amount requested between two microservice samples
- Bandwidth between edge nodes where microservice samples are deployed
- Network distance

A multi-objective optimization model is established, which aims to minimize transmission delay and load imbalance. Constraints are set on computing, communication, and caching resources. A linear weighting method is used to transform multiple objectives into a single objective. Hence, decisions on which node each microservice should be placed on can be made.

The implementation of the whole orchestration process includes container deployment location determination, container creation, and container migration. We mainly focus on the first part and design a containerized microservices orchestration algorithm as follows.

DE-Enabled Containerized Microservice Orchestration

This is a typical NP-hard problem. Heuristic algorithms are frequently used for this kind of problem to give an almost optimal result. DE is a widely adopted heuristic algorithm with fast convergence speed and accurate results. In this article, an adaptive DE-enabled containerized microservice orchestration algorithm is designed. The traditional differential evolution based on integers can easily be premature if the mutation rate is set too large, and the global search performance is poor. If the mutation rate is set small, the population diversity decreases, and the global search performance is poor. Adaptive operators can maintain diversity in the initial stage and prevent prematurity. With the progress, the mutation operator decreases to avoid the destruction of the optimal solution.

For the treatment of constraints: an idea similar to [11] is adopted to separate the fitness function from the constraints so that each individual has two fitness values. Evaluation criteria is established to cross-evaluate the feasible solution and the infeasible solution.

Inputs: These are relevant information of microservice application, edge node set, group capacity, maximum iteration times, shrinkage factor, and cross probability.

Process:

- Initialize the population.
- Judge whether the termination condition is met.
- If not, perform mutation, crossover, and boundary condition processing, calculate the objective function, select, and then judge whether the condition is met again; If the condition is met, the optimal result can be obtained.

Output: This comprises decisions on which node each microservice should be placed on.

Use Case

A distributed heterogeneous resource trust management platform is constructed in this article. The platform can support multi-domain heterogeneous resource trust sharing and microservices deployment for scenarios that hold latency-sensitive and computation-intensive applications, such as smart factories, smart grids, and smart cities. These scenarios all require multi-domain information interaction and task collaborative processing.

Smart grids, as an emerging trend for energy supply and management, hold a huge number of terminals and provide energy mainly for residential, industrial, as well as commercial customers. In this article, the highly elastic power grid, as an extension of smart grid, cooperates with industries to indicate different domains and is modeled as a use case. Due to the dynamicity of tasks, the resource usage of each domain varies a lot. However, because of the trust problem and integration difficulties caused by the heterogeneity of resources, it is hard to carry out multi-domain resource management. Given this, the proposed multi-domain heterogeneous resource trust management architecture plays the role of a distributed platform for resource sharing and scheduling, enhancing resource efficiency.

An architecture for the test environment is constructed in Fig. 5. Since guiding industrial users who have high power demand to actively participate in load sharing and shifting is the key to ensure high elasticity, demand-side management (DSM), which enables bidirectional energy information interaction, enhances energy usage performance further, and proposes real-time response requirements, is taken as a representative application in this part. Refer to the MHRTM architecture built previously. DSSs correspond to smart grid operators and industrial managers; requests correspond to industrial energy service

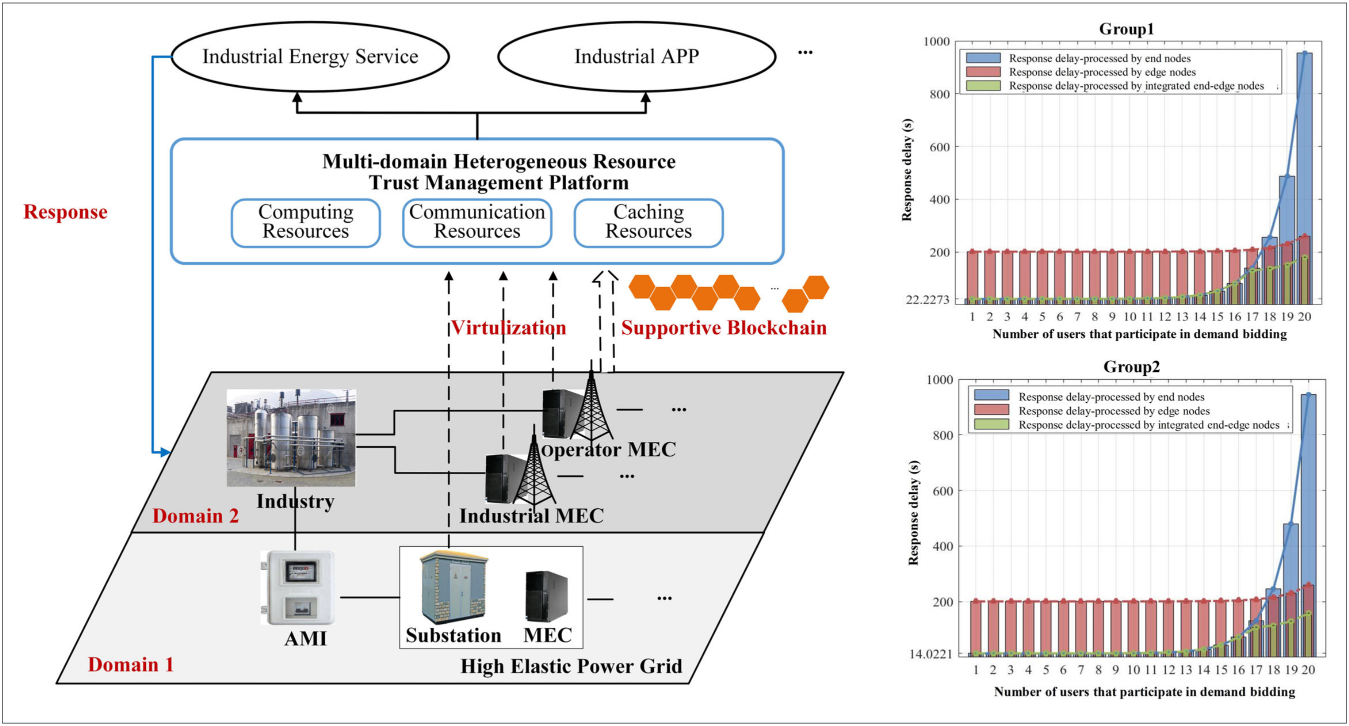
Industrial Energy Service
Industrial APP
...
Multi-domain Heterogeneous Resource Trust Management Platform
Computing Resources
Communication Resources
Caching Resources
Response
Virtulization
Supportive Blockchain
Operator MEC
Industry
Industrial MEC
Domain 2
AMI
Substation
MEC
Domain 1
High Elastic Power Grid
Group1
Group2
Response delay-processed by end nodes
Response delay-processed by edge nodes
Response delay-processed by integrated end-edge nodes
Response delay (s)
Number of users that participate in demand bidding
22.2273
14.0221

FIGURE 5. Use case and test results.

in the industrial domain; and energy price and demand negotiation, and energy usage schedule recommendation for the smart grid domain. MEC indicates a mobile edge server, while AMI represents advanced metering infrastructure. Once a service request is completed, a smart contract is triggered. The DE-enabled containerized microservices orchestration is used in the smart contract to implement microservices deployment, following which the containers with specific functions are created and finally migrate to the target edge node. The heterogeneous resources of the power grid and industry domains are virtualized and managed by the proposed platform, where blockchain, as a supportive technology, ensures trusted resource sharing.

A test environment is established in a province of China. Response delay is tested for demand bidding service while processing by end nodes, edge nodes, and integrated end-edge nodes. These two groups in Fig. 5 represent that different volumes of data are contained in the demand bidding process, which is an important part of DSM for determining who can participate in the demand shifting. The result shows that the third way (processing with integrated end-edge nodes), which was adopted in this article, shows the best performance, and its improvement becomes more and more obvious with the increase of number of participants. While the end nodes have limited processing capability, when it increases, the response delay increases exponentially. The test result verified the efficiency of the MHRTM architecture in reducing delay.

Conclusion

In this article, a multi-domain heterogeneous resource trust management architecture is proposed to address challenges when edge in-network computing meets blockchain. Blockchain is adopted for trusted resource sharing and transaction, while virtual technology is used to mitigate the heterogeneity of multi-domain resources, with a DE algorithm to support resource scheduling. A use case is provided to depict the application of the proposed solution for energy scenarios.

The proposed solution is still suffering from its own limitations. Each microservice is served by one container in this article, while the situation where a microservice corresponds to multiple containers should also be addressed. Furthermore, the architecture design may require some adjustments according to the scope of domains. There are more situations to discuss, and various use cases are also expected.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (62071070), in part by the Key Project Plan of Blockchain in the Ministry of Education of the People's Reputation of China (2020KJ010802), and in part by the Test Bed Construction of Industrial Internet Platform in Specific Scenes (new mode).

References

[1] Y. Tokusashi, H. Matsutani, and N. Zilberman, "LaKe: The Power of In-Network Computing," *Proc. Int'l. Conf. ReConFigurable Computing and FPGAs (ReConFig)*, Cancun, Mexico, 2018, pp. 1–8.
[2] J. Feng *et al.*, "Cooperative Computation Offloading and Resource Allocation for Blockchain-Enabled Mobile-Edge Computing: A Deep Reinforcement Learning Approach," *IEEE Internet of Things J.*, vol. 7, no. 7, July 2020, pp. 6214–28.
[3] F. Guo *et al.*, "Adaptive Resource Allocation in Future Wireless Networks With Blockchain and Mobile Edge Computing," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, Mar. 2020, pp. 1689–1703.
[4] S. Tuli *et al.*, "FogBus: A Blockchain-Based Lightweight Framework for Edge and Fog Computing," *J. Systems and Software*, vol. 154, Elsevier, Aug. 2019, pp. 22–36.
[5] C. Xu, K. Wang, and M. Guo, "Intelligent Resource Management in Blockchain-Based Cloud Datacenters," *IEEE Cloud Computing*, vol. 4, no. 6, Nov./Dec. 2017, pp. 50–59.

[6] Y. He *et al.*, "Blockchain-Based Edge Computing Resource Allocation in IoT: A Deep Reinforcement Learning Approach," *IEEE Internet of Things J.*, vol. 8, no. 4, Feb. 15, 2021, pp. 2226–37.

[7] W. Sun *et al.*, "Joint Resource Allocation and Incentive Design for Blockchain-Based Mobile Edge Computing," *IEEE Trans. Wireless Commun.*, vol. 19, no. 9, Sept. 2020, pp. 6050–64.

[8] H. Yao *et al.*, "Resource Trading in Blockchain-Based Industrial Internet of Things," *IEEE Trans. Industrial Informatics*, vol. 15, no. 6, June 2019, pp. 3602–09.

[9] Y. Tang *et al.*, "Elastic Regression-Tree Learning in a Heterogeneous Computing Environment," *IEEE Internet of Things J.*, vol. 6, no. 5, Oct. 2019, pp. 8826–34.

[10] K. Lin *et al.*, "Data-Driven Joint Resource Allocation in Large-Scale Heterogeneous Wireless Networks," *IEEE Network*, vol. 34, no. 3, May/June 2020, pp. 163–69.

[11] L. Yu *et al.*, "Stochastic Load Balancing for Virtual Resource Management in Datacenters," *IEEE Trans. Cloud Computing*, vol. 8, no. 2, 1 April-June 2020, pp. 459–72.

[12] H. Cao *et al.*, "Virtual Resource Allocation for Tactile and Flexible Services in UAVs-Integrated 5G Networks," *Proc. IEEE ICC*, Dublin, Ireland, 2020, pp. 1–6.

[13] Y. Zhou *et al.*, "Virtual Resource Allocation for Information-Centric Heterogeneous Networks With Mobile Edge Computing," *Proc. IEEE INFOCOM Wksps.*, Atlanta, GA, 2017, pp. 235–40.

[14] Z. Tan *et al.*, "Virtual Resource Allocation for Heterogeneous Services in Full Duplex-Enabled SCNs With Mobile Edge Computing and Caching," *IEEE Trans. Vehic. Tech.*, vol. 67, no. 2, Feb. 2018, pp. 1794–1808.

[15] G. Fan *et al.*, "Multi-Objective Optimization of Container-Based Microservice Scheduling in Edge Computing," *Computer Science and Info. Systems*, 2020, p. 41.

Biographies

Linna Ruan received her B.S. degree from Beijing Information Science and Technology University, China. She is currently studying for a Ph.D. degree at Beijing University of Posts and Telecommunications. Her research interests lie in edge and cloud computing for power and IoT scenarios, including resource allocation and service computation offloading.

Shaoyong Guo received his Ph.D. degree from Beijing University of Posts and Telecommunications. He is currently with the Department of State Key Laboratory of Networking and Switching Technology. His research interests include blockchain application technology, edge computing, and IIoT in energy Internet.

Xuesong Qiu is currently a professor and Ph.D. supervisor with the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications. He has authored about 200 SCI/EI index papers. He presides over a series of key research projects on network management and edge computing.

Luoming Meng graduated from Tsinghua University in 1987. He is a professor and Ph.D. supervisor. He is the Director of the Communications Software Technical Committee of China Institute of Communications and the Chairman of the National Network Management Standards Study Group.

Shuang Wu is a senior engineer and the deputy general manager of Ningxia Xintong Network Technology Co., Ltd. His current research interests lie in construction, and operation and management of power information and communication systems.

Rajkumar Buyya is a Redmond Barry Distinguished Professor and director of the Cloud Computing and Distributed Systems (CLOUDS) Laboratory at the University of Melbourne, Australia. He is a highly cited author in computer science and software engineering worldwide (hindex=148, g-index=320, 115100+ citations). For further information, please visit: www.buyya.com.