A Cooperative Game Theory and Blockchain based Elastic Bilateral Task Allocation Algorithm for In-network Computing

Wenyi Su*† , Zehang Qiu† , Kai Lei†
Mars Laboratory, Whittle School & Studios, Shenzhen, China
†ICNLAB, School of Electronic and Computer Engineering (SECE), Peking University,
Shenzhen 518055, P.R. China
Email:*wsu111@whittleschool.org,†zehangqiu@163.com
†Corresponding Author:leik@pkusz.edu.cn

2021 4th International Conference on Hot Information-Centric Networking (HotICN) | 978-1-6654-0549-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/HOTICN53262.2021.9680840

***Abstract*—The compute first networking is a new network architecture, which aims at combining the computing resources with the network, integrating the ubiquitous and dynamically distributed computing resources, network, storage and other computing resources in the network, and making them unified and intelligently scheduled. This study focuses on realizing the effective assignments of computing power related tasks and interactions between multi-belonging service providers in the service transaction process while taking into consideration of the complex network cooperation scenario, and meeting the needs of the service providers and users. In this paper, a formal abstract model is established for the compute first networking architecture, which highlights the characteristics of the multi-party complex cooperation situation and includes the solution of the task allocation problem in detail. Another main work of this paper is to propose a service trading system based on blockchain. Finally, the researchers conducted simulation experiments on Matlab platform and Ethereum Truffle framework, thereby proving the advantages of the proposed method in having comparatively high efficiency and performance.**

***Index Terms*—Compute First Networking, Blockchain, Game Theory, Resource Allocation**

I. Introduction

With the development of information and communication technology, not only did the number of electronic devices increased drastically, but also did the requirements for computing power. Specifically, many devices' users demand the usage of high bandwidth and low-delay computing services. However, current cloud computing models are far from sufficient in satisfying requirements of the users, and network with compute ability gains more attention. Thus comes compute first networking.

The key idea of compute first networking is to combine computing resources and networks, and intelligently regulate the dynamically distributed and ubiquitous computing power resources. There are two sides in compute first networking: the first is the thousands of users who send the tasks for computing power services, and the second is the service providers e.g. the cloud service providers, the carriers of basic communication infrastructures etc. who own the the service devices. The main goal of compute first networking is to accurately and dynamically match the tasks of the first side and the ability of the second by coming up with a reasonable service task allocation result. This result may allow multiple computing power tasks cooperatively completed on the network's multiple marginal nodes or cloud devices.

However, there are three challenges in applying compute first networking. First, users and service providers may have different requirements and priorities. For instance, users may focus more on the duration in completing their tasks, but the service providers may focus on the profits brought by the completion of the tasks. Second, since service devices may come from different service providers, trust constraint among devices becomes a problem. Third, the usability of the service devices can change with time, because they may experience crash-downs, or resource deprivation due to tasks overloading, but the state may not be disseminated in time.

Therefore, it is of importance to establish a mechanism that enables the matching and trading of computing power and the cooperation of multiple service devices from different serve providers, while guaranteeing the benefits received by the users and service providers. In this paper, we propose a "requirement-service" task allocation algorithm to tackle previous challenges. We discuss a mechanism to cooperated service devices from different providers through game theory and match compute tasks to devices considering the preference of users. We build the trading architecture on blockchain for its superiority in trust. Lastly, we conduct some experiments to examine the effectiveness of our algorithm.

There are three contributions we make:

- Through abstract analysis of features in compute first networking, this research conducted abstract modelling for the multi-agent, complex-cooperation process of computing power matching.
- This research proposed a "requirement-service" matching method based on game theory, which maximized utilities for multiple agents in a distributed, dynamic, and information-asymmetric setting.
- This research tested the proposed method on Matlab and Truffle architecture, and proved the method's applicabil-

ity.

The rest of this paper is organized as follows: Section II presents the literature review. Section III provides a detailed explanation of the task allocation method. Section IV expatiates the computing power trading system based on blockchain. Moreover, the experiments and results are discussed in Section V. Finally, Section VI gives conclusion.

II. Literature Review

A. Task Allocation Problem

In the cloud computing and marginal computing system, the task allocation problem can be divided into two parts: task matching problem and task unloading problem. For the task matching problem, a novel task regulation algorithm is proposed in [1], which contained 4 parameters, including task completion duration, resource costs, system security, and reliability. For the task unloading problem, by identifying the service devices and users into sellers and buyers, Gao et al.[3] converted the task unloading problem to a periodic auction problem, and formalized it as a restricted total profit optimization problem.

However, previous works employed heuristic algorithms in solving the task allocation problem, so they failed to take heed of the multi-agents, complex-cooperation characteristics of computing first networking, which includes the competition, cooperation, and interactions between many service providers. In this situation, it is impossible for the heuristic algorithms to obtain all the information required, so their solutions are far from prefect.

B. Game Theory

Several attempts had been made to incorporate game theory into the task allocation problem. Reference [6] focused on the cloud marginal cooperation task allocation scenario converting the problem to a multi-agent non-cooperative game. Du et al.[2] came up with a resource sharing mechanism based on Differential Stackelberg Game Model to incentivise the resource trade and sharing between a single cloud computing and fog service providers. Researchers in [5] proposed an algorithm based on game theory and auction theory to match the proposed prices of the buyers and sellers.

However, these solutions did not consider about the information-incomplete scenario, that it is impossible for different service providers to share sensitive information such as the remaining resources of service devices with each other. Also, unlike the situations in the these previous works, the participants in the computing first networking can only reach partial cooperation, while the bulk of their interactions remain competitions.

C. Blockchain

As a point-to-point oriented decentralized tool, blockchain has mechanisms including multi-agents simultaneous storage via ledger mechanism and programmable smart contracts that enables automatic interactions, so blockchain is considered as a trustworthy interaction tool in the decentralized scenario.

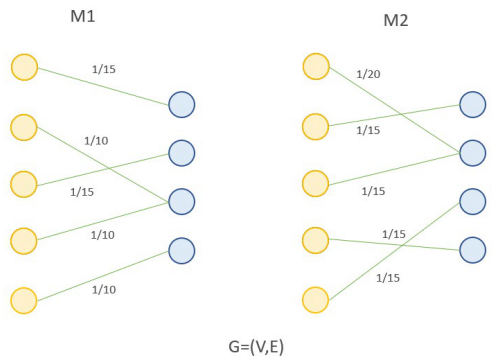

Fig. 1. Optimal bipartite matching

Research conducted by Xiong et al.[7] focused on the data trade market, in which the researchers constructed a concise data authentication and trade method by deploying smart contracts. However, to the extent of our knowledge, there are no systems based on blockchain that specifically target the trade of computing power, which is newly proposed in this paper.

III. Task Allocation Method

A. Problem Formation

In the problem of task allocation, there is a set of service providers S $= \{S_A, S_B, ...\}$ that implement M service devices in different locations, denoted by the set F $= \{F_{A1}, F_{A2}, F_{B1}, F_{B2}, ...F_M\}$. These service devices are used to satisfy the needs of users U $= \{U_1, U_2, ...\}$, who are also distributed over a wide geographical area. The task allocation problem can be formulated into a dichotomous optimal match problem containing a binary undirected graph $G = \{V, E\}$, in which the tasks of users and the service devices are formalized to be nodes, and the edges are formed if a user's task is allocated to a service device. Note that in G, one task can only be allocated to one service device. For G, we want to find a connection scheme that maximizes the sum of values on the edges, which is calculated by various service features such as transmission duration and processing duration. As shown in Figure 1, the sum of values on edges in M1 is greater than that of M2, so M1 is the better solution.

Like in previous research [8], we make the assumption that the user tasks remain unchanged in a typical time-step. In this time-step, a group of user tasks T $= T_1, T_2, ...T_n$ are to be allocated to solve the task allocation problem. Some objectives need to be formally defined before solving.

Before further expatiation, Table 1 is a variable table for reference.

In Table 1, R_{ser} and R_{SP} are respectively calculated by

$$R_{ser} = \sum T_i \cdot Com_i \cdot P_j \tag{1}$$

$$R_{SP} = \sum_{j=1}^{m} R_{ser} \tag{2}$$

TABLE I
NOTATIONS

Variable Table	Description
F	The set of all service devices
A	The set of all actions
Re_i	The amount resource spent on a task
R_{SP}	The reward received by a service provider
C_{sj}	The total cost of implementing all service devices
P_j	The ratio of provided resource of a service device in its coalition
C	The set of all cooperation coalition
T	The set of all types
RT	The total resources owned by all service device
Rev_i	The reward of a task
Com_i	Whether a task is completed (1 means true, 0 means false)
R_{Ser}	The total reward received by a service device
P	The budget proposed by the users for their tasks, known before the tasks are allocated

where m is the number of service devices a service provider has.

For the metrics we constructed, firstly, the average time of the users' required duration of completing their tasks is

$$\text{Time(T, F)} = \frac{1}{n} \sum_{T_i \in T, t(T_i) < time_i} t(T_i) \tag{3}$$

where n denotes the number of user tasks. Secondly, to quantify the extent for which the users' requirements are satisfied in a time-step, we formalized two other metrics

$$QoE_{service}\text{T, F} = \frac{s}{n} \tag{4}$$

$$QoE_{time}\text{T, F} = \frac{Time(T,F)}{QoE_{service}\text{(T,F)}} \tag{5}$$

where s is the number of tasks completed.

Thirdly, to take into account the benefits of the service devices, we constructed the following 4 metrics. The first metric is the utilization rate UR of a service device

$$\text{UR(T, F)} = \frac{\sum_{i \in (1,n)} Re_i}{RT}. \tag{6}$$

The second is the total reward R of the current task allocation system

$$\text{R(T, F)} = \frac{1}{n} \sum_{i=1}^{n} Com_i \cdot Rev_i. \tag{7}$$

where $Com_i = 1$ denotes whether the task is finished and otherwise $Com_i = 0$. The third is economical efficiency Eco_{ser} for a service provider S_i

$$Eco_{ser}\text{(T, S, F)} = \sqrt{\frac{1}{k} \sum (R_{SP} - \sum_{i=1}^{n} Rev_i \cdot \frac{C_{sj}}{\sum_{j=1}^{k} C_{sj}})^2}. \tag{8}$$

B. Cooperation Coalition Model

Due to physical area restrictions, service devices from different service providers can form coalitions to achieve better task allocation results as shown in Figure 2. Service devices from different service providers are not able to share sensitive information with each other, so the problem of forming coalitions among all the service devices can be described as a cooperation game with incomplete information. This game can further be described as a 5 unit tuple <F, C, A, T, B>: F is the set of all service devices, C is the set of all coalitions formed, A represents the set of all possible actions. Specifically, when forming coalitions in each iteration, there are two actions for each coalition, which are to remain to be itself, or to request to unite itself with another coalition. T = $\{T_0, T_1, ..., T_n\}$ is the set of types (storage, computation etc.) of the service devices. In the process of gaming, the service devices that belong to the same coalitions know types T_i of each others, but they do not know the types of service devices from other coalitions. The only two other information they know about the other coalitions is that the types of service devices of the other coalitions also belong to some T, and they can deduce a probability B of the types of the service devices that do not belong to their coalitions, or known as the Bayesian Belief[4]. B is calculated by

$$B = \text{p}(t = T_i|x) = \frac{(x|t = T_i)\text{P}(t = T_i)}{\text{P}(x)} \tag{9}$$

in each iteration of forming coalitions.

For a coalition to decide an action, it needs to further consider about its utility $U(C)$, calculated by

$$\text{U(C)} = \frac{\text{UR(T,F)} \cdot 100}{Eco_{ser}\text{(T,S,F)}}. \tag{10}$$

Only when the $U(\overline{C})$ of uniting itself with a new coalition is greater than the original $U(C)$, will the original coalition request the uniting action.

C. Coalition Formation Algorithm

The formation process of coalitions during each time-step is illustrated in Figure 3.

D. Distribution of Tasks within Coalition

After the formation of a typical coalition, tasks need to be allocated to the service devices within the coalition. The process can be described by the below steps.

1) Calculate a time priority Pri_i for each task according to Equation 11, and thereby separating all the tasks into two lists.
$$Pri_i = \begin{cases} 1, \text{if } time_i < \text{Pri\_divide} = \frac{1}{n} \sum_{i=1}^{n} time_i \\ 0, \text{if } time_i \geq \text{Pri\_divide} = \frac{1}{n} \sum_{i=1}^{n} time_i \end{cases} \tag{11}$$
where $time_i$ is the completing time restriction for a task.
2) Sort each list in descending order according to budget P of each tasks.
3) Remove a task at the top of the list where $Pri_i = 1$ and allocate it to the service device within the coalition

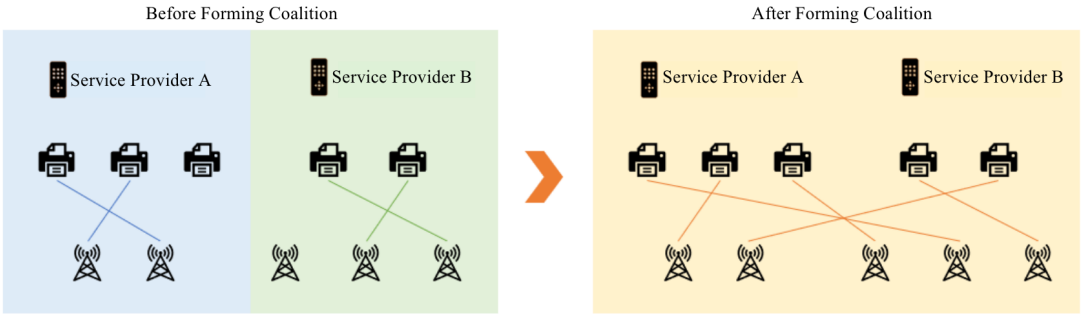

Fig. 2. Task Allocation with Coalition

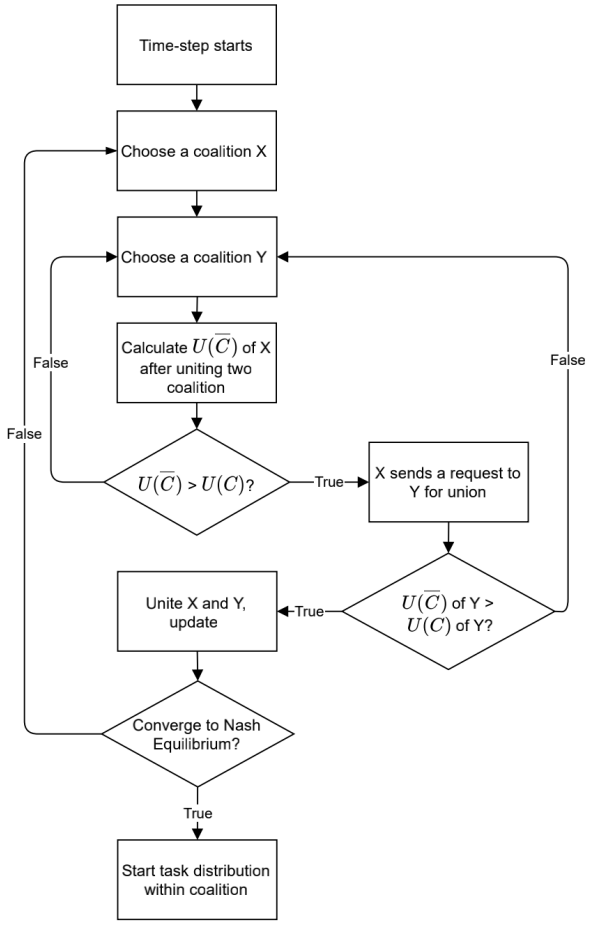

Fig. 3. Process of Forming Coalition

for which the transmission time is the smallest for the selected task. If the list is empty, remove and allocate a task at the top of the list where $Pri_i = 0$ instead.

4) Repeat Steps 1 to 3 until all tasks are allocated.

IV. Computing Power Trading System Based on Blockchain

A. Blockchain Architecture

Figure 4 demonstrates the architecture of the computing power trading system based on blockchain. The key components are

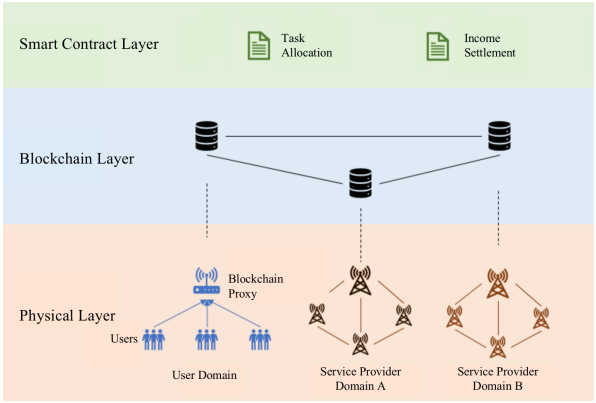

Fig. 4. Computing Power Trading System Architecture

1) User domain: The user domain composes of users who send the requirements for computation, storage, and network etc., and is willing to pay for the services. Note that because the users' devices are often limited in capabilities, the operation of user domain is accomplished by the blockchain proxies near the domain.
2) Service provider domain: Each service provider domain deploy a large amount of service devices, and these devices collectively form one or more service provider domains. Blockchain are deployed on some nodes (service devices) with high authority in the service provider domain.
3) Blockchain: The blockchain is composed of the blockchain proxies in the user domain and the blockchain in the service provider domain.
4) Smart contract: There are three types of smart contract in this study. The first is the task allocation smart contract; the second is the income settlement smart contract.

For the architecture of each block in blockchain, every block has two main components: the header and the trading data. The header is composed of the important metadata to ensure the completeness and correct sequence of blockchain, for instance, the hash value of the previous block, the unique identifier for a block, the timestamp, and the size of the block.

B. Consensus Algorithm

Because the number of participating nodes in the scenario of this study is relatively high, to ensure the efficiency and the scalability of the consensus algorithm, we used the DPoS algorithm of the Ethereum[9], and made an adjustment to increase its security and decrease its centralization.

In detail, we added the credibility rating for each node (device). For the selected committee members, we divide them by the ratio of $3:1:1$ to members who generate new blocks, the supervisors, and the substitute members. The supervisors are in charge of supervising the circumstances of the creation of the current blocks. If a member who is generating the new block misbehaves, he will be disqualified from generating new blocks, a substitute member will replace him in generating his new block, and his credibility rating will be decreased. The credibility rating of a member is calculated as

$$S_i(t+1) = 0.5 \cdot f \cdot S_i(t) + 0.5 \cdot S_i(t) \cdot (\frac{\mathrm{v}}{m} + \frac{\mathrm{c}}{n}) \quad (12)$$

where in a time-step t,

$$f = \begin{cases} 1 \text{ if the member succeed in generating his new block} \\ 0 \text{ otherwise,} \end{cases} \quad (13)$$

v denotes the number of times of validation success for a block, c denotes the number of communication success of the member's node, m denotes the total number of validation success for all blocks, n denotes the total number of communication success of all the members' nodes.

C. Smart Contract Description

The smart contract is considered as a safe, reliable, and automatic program. Currently, the developers can use Solidity language to develop and deploy their own smart contracts. Like other tradings in blockchain, deploying smart contract will send a trading to the blockchain. The information in the trading is the the code of the smart contract after compilation. If this trading can pass the consensus mechanism on the blockchain to become a block, than it will be able to accessed later solely via the address of the block. Similar to calling functions in programming, the users pass parameters to the smart contracts and accept the returned results, but the process of passing parameters to smart contracts is accomplished by the blockchain trade verification. The key information in a trading include the block address of the smart contract and the specific parameters.

Since the smart contract must be deployed on blockchain, its actual contents and calculation processes is transparent to all users using the blockchain. There are two smart contracts involved in this study as shown in Algorithm 1 and Algorithm 2.

Algorithm 1 Task Allocation Smart Contract

```
Function: task_allocation
Input: col[], list[]
for task_i in list[] do
    calculate time_pri
    if time_pri == 1 then
        add task_i to list1 sort by revenue DESC
    else
        add task_i to list2 sort by revenue DESC
    end if
end for
for task_i in list1, list2 do
    if s_min_tim(task_i) is available then
        add(task_i, server) to result
    else
        find s_min_time(task_i)
    end if
end for
return result
```

Algorithm 2 Income Settlement Smart Contract

```
Function: income_settlement
Input: add_task, add_ua, add_s, pay
if (add_task exist) && (ua, s in add_task == add_ua, add_s)
then
    if (Balance[add_ua] > 0) then
        Balance[add_ua] -= pay
        Balance[add_s] += pay
    else
        return false
    end if
end if
```

D. Computing Power Trading System Workflow

With the above information, we are now ready to present the complete workflow of the blockchain based computing power trading system, divided into 3 states, which are the task release state, task allocation state, and the task completion state.

1) Task Release State: The users submit tasks to the blockchain proxies for computing power services, along with information including their own operators, completion time requirements, and budgets. Then, the blockchain proxy publicizes these tasks. After these tasks are confirmed by the blockchain, they will be added to the task lists of the corresponding operators' service devices.

2) Task Allocation State: In this process, every service device update the members in its coalition, and utilized **task allocation smart contract** to obtain the task allocation results corresponding to the current coalitions, and conduct identity authentication mechanism. After all the coalitions reach Nash equilibrium, the task allocation completed. The final allocation result is a trading that connects to the history tradings on the blockchain, and this trading is signed by the users and the service providers.

3) Task Completion State: After the service providers complete the tasks, the service devices send the completion information to the blockchain. In the blockchain, the **income settlement smart contract** checks whether each task exists and is completed, and conducts the final payment transaction.

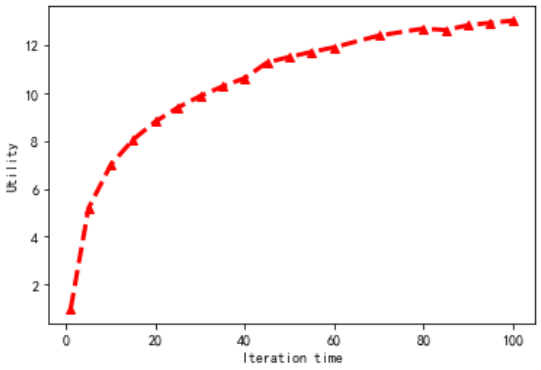

Fig. 5. Convergence Analysis of Coalition Formation Algorithm

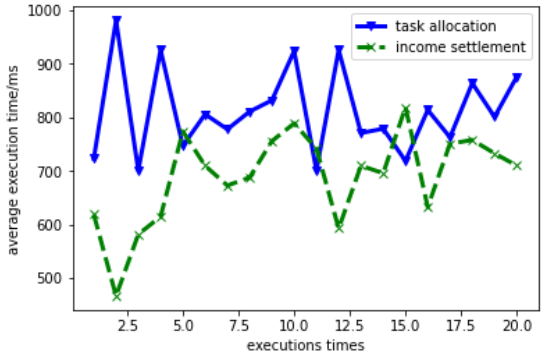

Fig. 6. Response Time of Task Allocation Smart Contract and Income Settlement Smart Contract

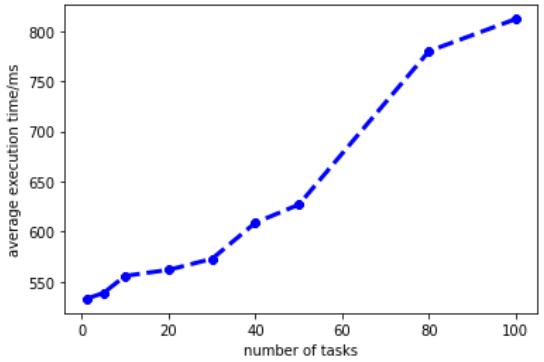

Fig. 7. Relationship between Number of Tasks and Response Time

V. Experiments & Results

A. Experiment on Task Distribution Method

1) Experiment Setup: The regional experiment contained 3 different service providers with 10 service devices in total, deployed in a 5×5 block matrix. In the current time-step, tasks each 5 belonging to a different service provider appear randomly in one block on uniform distribution. We suppose the income of each task fall in a random distribution between 1 and 10, and the distance from the task block to each service provider from 1 to 10.

2) Convergence Result: This study tested the convergence of the coalition formation algorithm in Section III-C for iterations of 1 to 100 times. As shown in Figure 5, after 80 iterations, the utility of the users approached to convergence.

3) Evaluation on Efficiency of Method: There are 4 metrics included to test the efficiency of the proposed method:

- The average time of completion of all tasks ($time_{avg}$)
- The utilization rate of the service devices, as calculated in Equation 6 in Section III-A (UR)
- The economy efficiency for the service providers, as calculated in Equation 8 in Section III-A (Eco_{ser})
- The execution duration of the proposed method in a time-step (dur)

Table II shows the experiment results. The method of involving coalition game had better results in the majority of the 4 metrics. Note that the method including coalition gaming had longer dur than its opponent, but this could be accounted by the higher complexity of the proposed algorithm, so the duration difference of 30ms is acceptable.

TABLE II
Experiment Result

Metric/Method	With Coalition	Without Coalition
$time_{avg}$	4.7	5.2
UR	0.74	0.66
Eco_{ser}	5.1	8.2
dur	127ms	93ms

B. Experiment on the Computation Power Trading System Based on Blockchain

1) Experiment Setup: In this study, we utilized Truffle to establish a consortium blockchain network with 10 nodes, among which 4 nodes were blockchain proxy nodes (represent users), and 6 nodes were the service device nodes belonging to 3 service providers, who each had 2 service devices.

We successfully established and executed the task allocation smart contract and income settlement smart contract in the network described in Section IV-C.

2) Performance Results: We conducted experiments on the performance of the proposed trading system, which were used to evaluate the response (execution) time and the relationships between the response time and various factors.

For the response time, firstly, Figure 6 illustrates the response time of the task allocation smart contract and income settlement smart contract. For the exact result, the average response time of task allocation smart contract is 812ms, and that of income settlement smart contract is 690ms. Comparing to the 300ms task allocation program response time in reference [10], and considering the higher complexity and greater security level of the methods in this study, the results should be acceptable.

For the relationship between the response time and the number of tasks, as illustrated in Figure 7, even when there

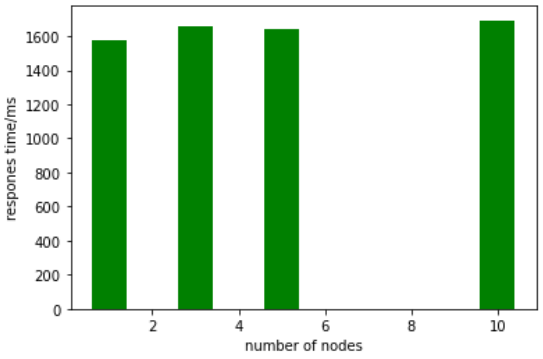

Fig. 8. Relationship between Number of Nodes and Response Time

was only one task, the response time approached 600ms. Thus, this is the delay due to the blockchain system. Also, when the number of tasks reached 50, the rate of increase in response time became high, so the optimal number of tasks allocated per time-step should be 50. Furthermore, Figure 8 demonstrates the relationship between the response time and the number of nodes in the blockchain network. As shown, the response time was stable with the increase number of nodes.

VI. Conclusion

Compute first networking is a kind of network architecture that focuses on integrating the ubiquitous and dynamically distributed network, storage and other computing resources in the network, and making them unified and intelligently scheduled. This study focuses on realizing the effective assignments of tasks to service devices and the effective interactions between multi-belonging service providers in the service transaction process while taking into consideration of the complex network cooperation game, and meeting the needs of service providers and the users.

In this paper, a formal abstract model is established for the compute first networking scenario, including a task allocation workflow based on Bayesian Coalition Game Theory, which enables the service devices to form coalitions to provide services for users. Another main work of this paper is to propose a computing power service trading system based on blockchain, with three smart contracts that ensure the trustworthiness and security of the system. To prove the effectiveness and applicability of the proposed methods, we conducted simulation experiments on Matlab platform and Ethereum Truffle framework.

For future work, we may integrate scalable blockchain consensus algorithms into our system; in addition, we may consider the application different blockchain architectures such as multi-link and DAG; furthermore, we may give further discussion of the security issues of our mechanism.

Acknowledgement

This work is supported by the National Science Foundation of China (NSFC 62072012), Key-Area Research and Development Program of Guangdong Province (2020B0101090003), Shenzhen Project (JSGG20191129110603831), and Shenzhen Key Laboratory Project (ZDSYS201802051831427)

References

[1] Yangyang Dai, Yuansheng Lou, and Xin Lu. "A Task Scheduling Algorithm Based on Genetic Algorithm and Ant Colony Optimization Algorithm with Multi-QoS Constraints in Cloud Computing". In: *2015 7th International Conference on Intelligent Human-Machine Systems and Cybernetics*. Vol. 2. 2015, pp. 428–431. DOI: 10.1109/IHMSC.2015.186.

[2] Jun Du et al. "Stackelberg Differential Game Based Resource Sharing in Hierarchical Fog-Cloud Computing". In: *2019 IEEE Global Communications Conference (GLOBECOM)*. 2019, pp. 1–6. DOI: 10.1109/GLOBECOM38437.2019.9013966.

[3] Guoju Gao et al. "Auction-based VM Allocation for Deadline-Sensitive Tasks in Distributed Edge Cloud". In: *IEEE Transactions on Services Computing* (2019), pp. 1–1. DOI: 10.1109/TSC.2019.2902549.

[4] Neeraj Kumar, Joel J. P. C. Rodrigues, and Naveen Chilamkurti. "Bayesian Coalition Game as-a-Service for Content Distribution in Internet of Vehicles". In: *IEEE Internet of Things Journal* 1.6 (2014), pp. 544–555. DOI: 10.1109/JIOT.2014.2374606.

[5] Sidra Malik et al. "TrustChain: Trust Management in Blockchain and IoT Supported Supply Chains". In: *2019 IEEE International Conference on Blockchain (Blockchain)*. 2019, pp. 184–193. DOI: 10.1109/Blockchain.2019.00032.

[6] Graham Romp. *Game theory: introduction and applications*. Oxford University Press on Demand, 1997.

[7] Wei Xiong and Li Xiong. "Data Trading Certification Based on Consortium Blockchain and Smart Contracts". In: *IEEE Access* 9 (2021), pp. 3482–3496. DOI: 10.1109/ACCESS.2020.3047398.

[8] Congying Yang et al. "Dynamic Allocation for Complex Mobile Crowdsourcing Task with Internal Dependencies". In: *2019 IEEE SmartWorld, Ubiquitous Intelligence Computing, Advanced Trusted Computing, Scalable Computing Communications, Cloud Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*. 2019, pp. 818–825. DOI: 10.1109/SmartWorld-UIC-ATC-SCALCOM-IOP-SCI.2019.00171.

[9] Fan Yang et al. "Delegated Proof of Stake With Downgrade: A Secure and Efficient Blockchain Consensus Algorithm With Downgrade Mechanism". In: *IEEE Access* PP (Aug. 2019), pp. 1–1. DOI: 10.1109/ACCESS.2019.2935149.

[10] Song Yang et al. "Survivable Task Allocation in Cloud Radio Access Networks With Mobile-Edge Computing". In: *IEEE Internet of Things Journal* 8.2 (2020), pp. 1095–1108.