Cloudchain: A Blockchain-Based Coopetition Differential Game Model for Cloud Computing

Mona Taghavi[1(✉)], Jamal Bentahar[1], Hadi Otrok[1,2], and Kaveh Bakhtiyari[3,4]

[1] Concordia Institute for Information System Engineering, Concordia University, Montreal, Canada
m_tag@encs.concordia.ca, bentahar@ciise.concordia.ca, hadi.otrok@kustar.ac.ae

[2] Department of ECE, Khalifa University, Abu Dhabi, UAE

[3] Interactive Systems, University of Duisburg-Essen, Duisburg, Germany
kaveh.bakhtiyari@uni-due.de

[4] Department of Electrical Engineering, The National University of Malaysia, Bangi, Malaysia

Abstract. In this paper, we introduce, design and develop *Cloudchain*, a blockchain-based cloud federation, to enable cloud service providers to trade their computing resources through smart contracts. Traditional cloud federations have strict challenges that might hinder the members' motivation to participate in, such as forming stable coalitions with long-term commitments, participants' trustworthiness, shared revenue, and security of the managed data and services. Cloudchain provides a fully distributed structure over the public Ethereum network to overcome these issues. Three types of contracts are defined where cloud providers can register themselves, create a profile and list of their transactions, and initiate a request for a service. We further design a dynamic differential game among the Cloudchain members, with roles of cloud service requesters and suppliers, to maximize their profit. Within this paradigm, providers engage in coopetitions (i.e., cooperative competitions) with each other while their service demand is dynamically changing based on two variables of gas price and reputation value. We implemented Cloudchain and simulated the differential game using Solidity and Web3.js for five cloud providers during 100 days. The results showed that cloud providers who request services achieve higher profitability through Cloudchain compared to those providers that supply these requests. Meanwhile, spending high gas price is not economically appealing for cloud requesters with a high number of requests, and fairly cheaper prices might cause some delays in their transactions during the network peak times. The best strategy for cloud suppliers was found to be gradually increasing their reputation, especially when the requesters' demand is not significantly impacted by the reputation value.

Keywords: Cloud service federation · Smart contract · Blockchain Ethereum · Differential game

C. Pahl et al. (Eds.): ICSOC 2018, LNCS 11236, pp. 146–161, 2018.
https://doi.org/10.1007/978-3-030-03596-9_10

1 Introduction

To mitigate the issue of underutilized and over provisioned computing resources, cloud providers scaled their pool of resources by forming cloud federations to maximize their profit and provide guaranteed Quality of Services (QoS) [4,6,11]. In spite of the prominent federation advantages, cloud providers are reluctant to participate in due to some strict challenges, mainly: 1- The stability of a federation is a key factor for the cloud providers to ensure their profitability [6]. Such a stability requires long-term commitments from the providers, which is very hard to obtain. 2- A federation needs to address the complications of a fair revenue sharing model to warrant that each cloud provider will gain a revenue according to the amount of computational resources contributed to the federation. 3- The presence of unknown and untrusted participants in a federation can degrade the QoS of the federated services [16]. The trust issue limits conventional federations to enroll only trusted providers and disregard the new ones. 4- Having a large pool of computing resources in a grand coalition might increase the opportunity of botnet attacks. Meanwhile, forming a small federation might hinder the revenue maximization of its participants [2]. 5- There are some security and privacy concerns regarding the managed data and services as well as the creation and management of the cloud federation itself. All the necessary information to manage a federation is usually maintained in a centralized trusted third party. This implies that a federation must maintain roles concerning the authorization to manage the participants' information that yields, which makes not only a single point of failure, but also raises trustfulness concerns [11].

Contributions: This research overcomes the traditional cloud federation issues by contributing a novel architecture and an innovative strategic game model:

1. To provide a practical cooperative solution that any cloud provider can embrace regardless of their market position and trustworthiness, we advocate a fully distributed architecture with a democratic governance structure, called *Cloudchain*. To effectively enforce such a structure, Cloudchain proposes an innovative exploitation of blockchain to prompt and support interoperability and coopetition among the cloud providers over the public Ethereum network. Within Cloudchain, cloud providers endeavor to overcome the resource limitation in their local infrastructure by outsourcing their customers' requests to other members of the Cloudchain. Moreover, it allows providers to access underutilized resources and lease them at cheaper prices. By leveraging blockchain-enabled smart contracts [17], we eliminate the need for trust in the federation and reduce barriers of entry [9].
2. To incentivize the cloud providers and help them make wise decisions about the utilization of Cloudchain, a dynamic differential game is designed, solved and simulated. This game aims to maximize the profit of the Cloudchain members who cooperatively compete while their service demand is dynamically changing. Two variables are considered to impact the cloud provider's revenue, the demand variability and the quality of the provided service: gas cost and reputation value. Gas is a proportional amount that Ethereum pays

to motivate the miners to participate in the mining process and to supply a fair compensation for their computation effort [12]. Reputation value is defined to assign a credibility proportional to the quality that a Cloudchain member provides.

We implement the Cloudchain prototype using Solidity and Web3.js which is available open source in Github[1]. We further simulated the differential game using the Gratner's rating dataset[2] where five real-world providers trade their services. Despite being costlier to transact for cloud-service providers who request a service rather than supply, the obtained results proved it is economically justified to adopt Cloudchain.

2 Related Work

The literature about cloud-providers cooperation focuses on federation formation as coalitional games where capacity and revenue are shared [15]. Coronado et al. had an intensive investigation on federation-formation variables among cloud providers, including revenue sharing mechanisms, capacity and cost disparity, and the presence of a big competitor [5]. They defined revenue sharing mechanisms as the most important factor. Among these mechanisms, shapely value and outsourcing models had the least and best performance, respectively. They indicated that collaborating cloud providers can implement a mechanism in which a provider outsources some of its business and gets a percentage of the revenue. The outsourcing model allows the provider to keep some of the revenue of its secured business, even though it is not able to fulfill that business alone. The authors had an insight through the demand peaks and concluded that cloud providers tend to stay in outsourcing collaboration when the demand is high. However, interoperability, trust among cloud providers and service quality or SLA are not considered in their study. The findings from this study confirm the superiority of outsourcing in terms of maximizing the profit of cloud providers, which is what we are proposing in this paper in addition of having the advantage of coopetition among cloud providers. The fact that providers tend to collaborate when they face a hike in their demand, reinforces the consideration of a dynamic and long/short-term federation like Cloudchain. The challenges of interoperability and trust issues among cloud providers are also addressed by the blockchain platform we propose in this paper. Another cloud outsourcing model has been performed by Chen et al. [4] who analyzed the interrelated workload factoring and coalition formation game among private clouds. The authors integrated two types of federations: (1) vertical (outsource workload to public clouds), ad (2) horizontal (share resources with other private clouds. Their experiments found this approach to be promising to improve the cloud's service quality and decrease the delay by 11%. However, their research was limited to service quality and economic aspects of stable cooperation patterns without

[1] https://github.com/kavehbc/Cloudchain.
[2] https://www.gartner.com/reviews/market/public-cloud-iaas.

considering other challenges of a traditional federation explained in the previous section.

Very few efforts have been made to study the potential of blockchain in real-world applications despite its great potential for businesses to share data and collaborate in a secure and customized manner [13]. According to Tractica, a market research firm, the annual revenue for enterprise applications of blockchain is estimated to increase to $19.9 billion by 2025 [8]. The majority of studies about blockchain's application have focused on finance [19], energy [14] and IoT applications [21]. In cloud computing and service industry, to the best of our knowledge, there has been only one academic initiative that proposed a cloud marketplace based on the blockchain technology. Klems et al. designed Desmaa, a conceptual framework for trustless intermediation in service marketplaces using blockchain [9]. This conceptual framework modeled the interactions between a service provider and a service consumer and tried to overcome problems of conventional marketplace systems, such as barriers of entry and transaction costs. Yet, the outsourcing model with collaboration and competition among cloud providers themselves are not considered in their research. Moreover, the providers' profit and the best strategies for utilizing this marketplace is not elaborated nor modeled. Even though the authors developed a prototype, no evaluation and validation against real-world's scenarios were provided.

3 Cloudchain Architecture

Cloudchain incorporates three types of smart contracts including a set of executable functions and state variables. Similar contracts are proposed in [1] in the context of medical data management. *Contract 1* (*C1*) or Cloudchain Registery (CCR) is a global contract that maps cloud providers identification values (including *Name*, *Reputation Value*, *Computing Capacity* and *Storage Capacity*) to their Ethereum address identities (equivalent to public keys). The reputation values can be computed from the customers' ratings given to each provider through online rating platforms. Policies coded into the contract can regulate registering new providers or changing the mapping of the existing ones. The cloud provider registration can be restricted only to certified providers. CCR also maps identities to the Cloudchain Contract (CCC) address on the blockchain, where a special contract regarding each provider profile and list of services is recorded.

Contract 2 (*C2*) denotes Cloudchain Profile (CCP). It holds a list of references to CCC, representing all the participants' previous and current engagements with other nodes in the system. CCP also implements a functionality to enable provider notifications. Providers should register their requests in this contract. Each transaction list stores a status variable. This indicates whether the transaction is newly established, awaiting pending updates and has or has not been completed. This contract is important as it stores the address of all new CCC contracts, without which Cloudchain can simply lose the track of all the contracts.

Contract 3 (*C3*) represents the Cloudchain Contract (CCC). It is issued between two nodes in the system when one node accepts and provides the requested service for the other. The beneficiaries can also complete, or cancel the contract. Once the contract is completed or canceled, the contract balance would be transferred to the supplier-, or requester address respectively, and the contract status would also be updated. There are two approaches to reduce the size of the data as well as the cost of transactions over Cloudchain. The first approach is a common practice for data storage in smart contracts and consists of storing raw data off-chain, and meta-data, small critical data, and hashes of the raw data on-chain [20]. However, the selection of off-chain data storage has some concerns regarding the interaction between the blockchain and the off-chain data storage. The other approach is to provide a common glossary among cloud providers to define the generic terms and policies to be referred to in the contract.

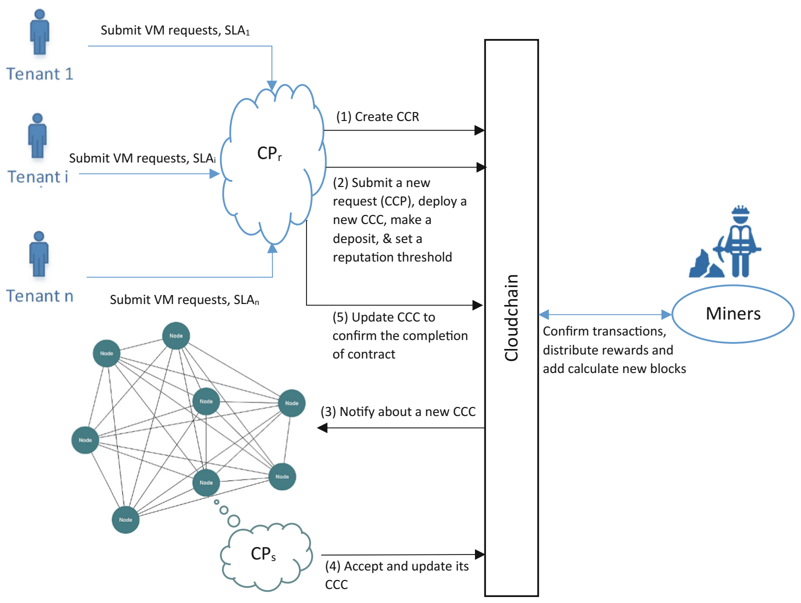
Tenant 1
Submit VM requests, SLA$_1$
Tenant i
Submit VM requests, SLA$_i$
Tenant n
Submit VM requests, SLA$_n$
CP$_r$
(1) Create CCR
(2) Submit a new request (CCP), deploy a new CCC, make a deposit, & set a reputation threshold
(5) Update CCC to confirm the completion of contract
(3) Notify about a new CCC
(4) Accept and update its CCC
CP$_s$
Node
Cloudchain
Miners
Confirm transactions, distribute rewards and add calculate new blocks

Fig. 1. Cloudchain interactions

Figure 1 provides the steps taken by the Cloudchain members to register and establish their requirements by interacting via Cloudchain. In step 1, a provider registers in CCR. Each registered user is assigned with a public key pair. Guaranteed SLA tenants require performance consistency and scale predictability. When a member faces a computing-resources deficiency to meet its end users' demand with guaranteed SLA, it can submit a request for a service using CCP to deploy a CCC to the blockchain in step 2. Requesters are required to pay a deposit in advance and it is stored in the contract. Meanwhile, a rule for providers is set by the requesters to ensure that qualified providers could ultimately receive the task, e.g. reputation value threshold. Function calls on contracts are transactions, and those which update the contract storage need to be validated by miners. Once a new block is mined with the newly linked CCC, it would be broadcasted to other nodes in step 3. Through step 4, the first node that accepts the request should update the respective CCC contract. Each provider who accepts a task should deposit some coins or its reputation value to guarantee the quality of the task. The contract termination and delivery of the requested service have to be confirmed by the service requester in step 5. The requester is required to rate the supplier based on the received service quality.

4 Cloudchain Members' Revenue Optimization

A true blockchain-led federation will not happen unless cloud providers are widely engaged and able to manage the costs and properly play their role. The use of a differential game [3] is motivated by the need to model the time constrained and dynamic strategies of selfish cloud providers willing to maximize their own revenue. Let us consider two typical cloud providers (CP) over Cloudchain, CP_r as the provider who is facing a peak time and is going to request some VMs from other Cloudchain members, and CP_s as the Cloudchain member that has some idle servers and is willing to rent them out with the price offered by CP_r. For simplicity and without losing generality, we will focus on a single VM type, with ϕ denoting the capacity and the process rate of VM instances that can be hosted by a typical CP_j that can be CP_r or CP_s.

To make a request and create a contract, CP_r has to define the price of gas G_r for the created transaction (e.g. 5 gwei). If the price is high enough, the transaction will be executed sooner, since miners will execute transactions with the highest gas price first. If the price is set too low, CP_r may end up waiting longer for execution of its transaction and distribution of its request. This waiting time may degrade the service quality for its users and hinder its profit. On the other hand, setting a high gas price for every single transaction and update incurs higher costs. So, the gas price is a decisive factor in profit optimization and we define it as the control path at time t for CP_r, denoted by $G_r(t)$. In this game, VM price is assumed to be given by CP_r. To be qualified to supply a cloud service, the provider CP_s has to maintain a good reputation R_s which is given based on the quality of service for end users and the quality of collaboration (e.g. speedy communication) with the cloud provider that requested the service, CP_r. Even though the reputation value is given by CP_r and not CP_s itself, yet it has a control over this value through the service quality and gas price of its own transactions. Therefore, $R_s(t)$ is considered as the control path of CP_s to coopete with CP_r within Cloudchain to gain higher profit. Table 1 provides a summary of the notations used in our model.

In order to capture the demand elasticities and variations specific for each user, we define the user demand using the Cobb-Douglas function that models well these elasticity aspects in terms of price and reputation, adopted from [18]. It is assumed that the user will have the opportunity to check the cloud provider

Table 1. Notations used in Cloudchain

u	End user index
$r, s \in \{1, 2, \dots, j, \dots, k\}$	Requester and supplier in the set of k cloud providers in Cloudchain
G	Cost of gas
M	The amount of the cumulated gas for each block
$M^{'}$	The amount of required gas for each transaction
X	Number of the transactions occurring over Cloudchain
ω	Rate of transactions arrival for a CP in $[0 - T]$
$\varGamma$	Rate of the block generation for miners in Cloudchain
$\phi/\phi^{'}/\phi^{''}$	Provider's active/idle/mining capacity
$p/p^{'}$	Price per VM for the end user/for the members of Cloudchain
R	Reputation value of cloud provider in Cloudchain
Rw	Reward value of mining
τ	Block propagation time
η	Rate of the impact of M over τ
$\check{\theta}$	Rate of CP_r demands rise due to the higher quality services of CP_s
$\hat{\theta}$	Rate of $D^{'}_r$ demands increase due to higher reputation of CP_s
ψ	Rate of CP_r demands increase due to higher gas and higher quality
α_u/β_u	CP price/rating variation for user u
δ	Demand decay rate
μ	The amount of VMs
$C/C^{'}$	Cost of the primary/outsourced capacity

rating that represents the actual user satisfaction level and reputation value defined through Cloudchain. The user demand function is defined as follows:

$$D_u = \mu \, p^{-\alpha_u} \; R^{\beta_u} \tag{1}$$

In the mining race, miners have to compete to solve proof of work and propagate the block to reach consensus. The new blocks' generation follows a Poisson process with a constant rate $\frac{1}{\varGamma}$ throughout the whole Cloudchain network [10]. Before the race, miners collect their selected pending transactions into their blocks with a total gas amount of $\sum_{j=1}^{k} M_j$. When miner j propagates its block to Cloudchain for consensus, the time for verifying each transaction is affected by the size of transactions M_j. The first miner j who successfully has its block achieves consensus will be rewarded based on the amount of the assigned capacity $\phi^{''}_j$. Thus, miner j's expected reward $Rw_j(\phi^{''}_j)$ is:

$$Rw_j(\phi^{''}_j) = Rw_j \mathbb{P}_j(\phi^{''}_j, M_j) \tag{2}$$

where $\mathbb{P}_j(\phi^{''}_j, M_j)$ is the probability that miner j receives the reward by contributing a block. To win the reward, provider must perform a successful mining and instant propagation. The miner may fail to obtain the reward if its new block does not achieve consensus as the first. This kind of mined block that cannot be added to the blockchain is called orphaned block. The block containing a larger size of transactions has a higher chance of becoming orphaned since a larger block requires more propagation time, thus, causing a higher delay for consensus. As the arrival of new blocks follows a poisson distribution, miner j's orphaning probability, $\mathbb{P}_j^0$, can be approximated as:

$$\mathbb{P}_j^0 = 1 - exp(-\frac{1}{\Gamma})\tau_j \tag{3}$$

Here, we assume miner j's block propagation time τ_j is linear with the size of transactions in its block, $\tau_j = M_j \eta_j$, where η_j is a constant that reflects the impact of M_j over τ_j. Therefore, we obtain the reward probability as follows:

$$\mathbb{P}_j(\phi_j^{''}, M_j) = 1 - \mathbb{P}_j^0 = \phi_j^{''} e^{-\frac{1}{\Gamma} M_j \eta_j} \tag{4}$$

Substituting Eq. 4 into Eq. 2 provides an estimation of total revenues that CP_j may obtain by attending the mining tournament. To model the transactions' distribution, we use the compound Poisson process, which is a generalization of the Poisson process where each arrival is weighted according to a distribution. The compound Poisson process represents better the transactions dynamics. In this case, the assumption is that transactions sent to Cloudchain follow a Poisson process, but the amount of gas they require follows a compound Poisson process. The reason is that the difference between the amount of gas is based on the complexity of the transaction, for example, the creation of a contract requires a much higher amount of gas than updating the contract. Therefore, the probability of the required gas by X_j transactions occurring in $[0-T]$ follows an exponential distribution based on the compound Poisson process as follows:

$$\mathbb{P}_j(X) = \frac{e^{-\omega T}(\omega T)^{X_j}}{X_j!} \tag{5}$$

4.1 Cloud Provider as a Requester

Here we explain the scenario from the perspective of CP_r that has to optimize its profit CPP_r while requesting VMs as follows:

$$\begin{aligned} CPP_r(G_r(t), D_r^{'}(t), t) &= (p_r - \ \phi_r \ C_r) \ D_r + (p_r - p_s^{'} \phi_s^{'}) D_r^{'}(t) \\ &- \frac{e^{-\omega T}(\omega T)^{X_r}}{X_r!} M_r^{'} G_r(t) + R w_r \phi_r^{''} e^{-\frac{1}{\Gamma} M \eta} \end{aligned} \tag{6}$$

$M_r^{'}$ represents the amount of required gas that depends on the complexity of the transaction a provider wants to initiate. The transaction fees go to the miner that mines the block, so if a provider attends a mining process, it will be rewarded according to Eqs. 2 and 4. $D_r^{'}(t)$ is the demand that CP_r intends to outsource to obtain the idle capacity of $\phi_s^{'}$ for a secondary price of $p^{'}$. Considering the time-dependent profit functions of CP_r in Eq. 6, the objective function is the total discounted cloud provider's payoff over the planning horizon $[0-T]$:

$$\begin{aligned} \text{maximize} \quad & \int_0^T e^{\rho t}\{CPP_r(G_r(t), D_r^{'}(t), t)\}dt \\ \text{subject to} \quad & \dot{D}_r^{'}(t) = G_r^{\beta_u}(t)\psi_r + \check{\theta} R_s^{\beta_u}(t) - \delta_r D_r^{'}(t) \\ & D_r^{'}(0) = D_{0r}^{'} \end{aligned} \tag{7}$$

The users' demands evolution over time is represented as $\dot{D}_r^{'}(t)$ for CP_r that increases when the service quality rises. The service quality is aggregated through two factors of gas price that CP_r pays and the reputation of CP_s. The demand decays at a certain rate of δ_r. It is important to note that Eq. 7 formulates an optimal control problem with the gas price as a control variable and the cumulative demand of CP_r as a state variable. The analysis of differential games relies profoundly on the concepts and techniques of optimal control theory [7]. To study the dynamics of the payoff function and the path of control variable, we leverage the Hamiltonian systems. Equilibrium strategies in the open-loop structures can be found by solving a two-point boundary value problem for ordinary differential equations derived from the Pontryagin maximum principle in Hamiltonian functions. The Pontryagin maximum principle gives the necessary condition for a control path to be optimal open-loop control. To acquire the optimal control, we first formulate the Hamiltonian system of the cloud provider's payoffs:

$$\begin{aligned} H_r(G_r(t), D_r^{'}(t), \lambda_r(t), t) = (p_r - \ \phi_r \ C_r) \ D_r(t) + (p_r - p_s^{'}\phi_s^{'})D_r^{'}(t) \\ - \frac{e^{-\omega T}(\omega T)^{X_r}}{X_r!} M_r^{'} G_r(t) + R w_r \phi_r^{''} e^{-\dfrac{1}{\varGamma} M\eta} \\ + \lambda_r(t)(G_r^{\beta_m}(t)\psi_r + \check{\theta} R_s^{\beta_m}(t) - \delta_r D_r^{'}(t)) \end{aligned} \tag{8}$$

According to the control theory, the optimal control strategy of the original problem must also maximize the corresponding Hamiltonian function. Thus, based on the Pontryagin maximum principle, the candidate optimal strategy has to satisfy the following necessary conditions:

$$\frac{\partial H_r(t)}{\partial G_r(t)} = -\frac{e^{-\omega T}(\omega T)^{X_r}}{X_r!} M_r^{'} + \lambda_r(t)\beta_m G_r^{\beta_m - 1}(t)\psi_r = 0 \tag{9}$$

$$\dot{\lambda}_r(t) = \rho\lambda_r(t) - \frac{\partial H_r(t)}{\partial D_r^{'}(t)} = (\rho + \delta_r)\lambda_r(t) - p_r + p_s^{'}\phi_s^{'}, \lambda_r(T) = 0 \tag{10}$$

When only one boundary condition is specified as $D_r^{'}(0) = D_{0r}^{'}$, the free-end condition is used as $\lambda_r = 0$ at $t = T$. The formulated differential equation Eq. 10 can lead us to the adjoint variable:

$$\lambda_r(t) = \frac{p_r - p_s^{'}\phi_s^{'}}{\rho + \delta_r}(1 - e^{(\rho+\delta_r)(t-T)}) \tag{11}$$

Replacing Eq. 11 in Eq. 9 gives us the optimal gas price control path as follows:

$$G_r^*(t) = \left(\frac{M_r^{'} e^{-\omega T}(\omega T)^{X_r}(\rho+\delta_r)}{X_r!(p_r - p_s^{'}\phi_s^{'})(1-e^{(\rho+\delta_r)(t-T)})\beta_m \psi_r}\right)^{\frac{1}{\beta_m - 1}} \tag{12}$$

4.2 Cloud Provider as a Supplier

Cloud provider as a supplier has a different scenario. CP_s observes the total demand of its own users, D_s, and the capacity preserved for the mining process to determine the remaining capacity $\phi_s^{'}$, to optimize its profit as follows:

$$\begin{aligned} CPP_s(R_s(t), D_r^{'}(t), t) &= (p_s - \ \phi_s \ C_s) \ D_s + (p_s^{'} - \ \phi_s^{'} \ C_s^{'}) D_r^{'}(t) \\ &- \frac{e^{-\omega T}(\omega T)^{X_s}}{X_s!} M_s^{'} G_s(R_s(t)) + R w_s \phi_s^{''} e^{-\frac{1}{\Gamma} M\eta} \end{aligned} \tag{13}$$

$G(R_s(t))$ denotes the gas cost that the suppliers pay to earn higher reputation for having prompt communication. Considering the time-dependent profit functions of CP_s in Eq. 13, the objective function is the total discounted cloud provider's payoff over the planning horizon $[0 - T]$:

$$\begin{aligned} \text{maximize} \quad & \int_0^T e^{\rho t}\{CPP_s(R_s(t), D_r^{'}(t), t)\}dt \\ \text{subject to} \quad & \dot{D}_r^{'}(t) = \hat{\theta}_r R_s(t)^{\beta_n} - \delta_s D_r^{'}(t) \\ & D_r^{'}(0) = D_{0r}^{'} \end{aligned} \tag{14}$$

The demand dynamics of CP_s is defined based on the demand that it receives from CP_r that evolves with its own reputation and decays at a rate δ_s. By solving a corresponding Hamiltonian system of Eq. 14, similar to Eq. 8, the optimal reputation control path is obtained as follows:

$$R_s^*(t) = \left(\frac{M_s^{'} e^{-\omega T}(\omega T)^{X_s} G_s(\rho+\delta_s)}{X_s!(p_s^{'} - \phi_s^{'} \ C_s^{'})(1-e^{(\rho+\delta_s)(t-T)})\beta_n \hat{\theta}_r}\right)^{\frac{1}{\beta_n - 1}} \tag{15}$$

5 Implementation, Simulation and Discussion

We implemented the coopetitive Cloudchain prototype on Ethereum using Solidity (version 0.4.24), the script language on Ethereum, to test our proposed framework and the effect of gas price and reputation values on cloud providers revenues. This program is available open source in Github (See footnote 1). The program was written with the main concern of the minimum consumption of gas per each transaction and was tested using remix[3], an online IDE for Solidity.

[3] http://remix.ethereum.org/.

The gas price unit is in gwei, which is 1×10^{-9} ether. Ethereum stores arbitrary data in smart contracts in two ways. The first option is to store the data as a variable in a smart contract. The cost of storing data in the contract storage is based on the number of SSTORE operations on the contract variable. The second option is to store arbitrary data as a log event. There are also memory variables such as contract arguments and defined memory variables, which are not stored permanently inside the contracts. Memory variables are disposed after the function execution is complete. In our implemented prototype, we used solidity structures and variables to store provider's data and requests inside the contracts. Meanwhile, each transaction is logged with a summary using an event to make it easily accessible for the other providers (blockchain nodes) to track new transactions. Once a new transaction with a specific event (e.g. New Request) is created, other providers can call the contract to get more information and/or change contract stored data (e.g. to accept a new request). Calling a contract and retrieving data are expensive transactions, the stored data as events can provide enough information without any retrieval cost. The events are retrieved and filtered using the Web3.js platform to notify the providers on important changes (e.g. New registration, updates, deactivations, new requests, etc.) in Cloudchain. CCR and CCP contracts are deployed once, but CCC would be deployed every time a new request is registered.

Table 2. Provider's estimated transactions and costs on Cloudchain based on the proposed scenarios

	Amazon EC2	Microsoft Azure	Rackspace	Century Link	Alibaba cloud
Reputation value	88	82	84	60	82
Price per hour (p)	0.0058	0.005	0.084	0.025	0.0125
Price per hour (p')	0.003	0.0025	n/a	n/a	n/a
Requests[a]	0	0	8	15	17
Supplies[a]	23	17	0	0	0
Cancellations [a]	0	0	0	3	2
Total gas	1,290,668	953,972	15,292,736	34,310,286	36,254,668
Gas price (gwei)[b]	15	15	15	12	11
Gas cost (gwei)[c]	19,360,020	14,309,580	229,391,040	411,723,432	398,801,348
Gas cost (USD)[c]	\$12.06	\$8.91	\$142.91	\$256.50	\$248.45
Transaction delay (s)[d]	27-66	27-66	27-66	27-4000	27-5459

[a]Quantity [b]Total Gas×Gas Price [c]Average [d]Time range of each transaction in seconds

For the sake of representation, we assumed a small number of 5 cloud providers (Amazon, Microsoft, Rackspace, Alibaba cloud, and Century Link) using Cloudchain for a duration of 100 days to investigate their economic gain through the differential game. The scalability of our system for higher number of cloud providers is not questioned since the Ethereum platform is proven to be scalable. We simulated Rackspace, Alibaba and Century Link as cloud requesters who make 8, 17 and 15 requests of service, respectively. Meanwhile, Amazon accepts Rackspace and Century Links requests with a reputation threshold of 75, and Microsoft takes Alibaba's orders, which were set for a minimum reputation of 85. Due to the limitation of Solidity in defining float numbers, we scaled the reputation values collected from Gartner to [0–100]. The on-demand cloud services' prices are borrowed from the providers' websites with an assumption of the secondary price of Amazon and Microsoft to be two times less for the Cloudchain members. The collected real-world data (e.g. reputation and price), simulated number of requests and supplies, as well as the simulated results of total gas consumption, gas price and transactions delays are shown in Table 2. Since there is no time-dependent profit maximization model similar to our proposal, not even in traditional centralized federations or related experiments to be compared to, only the results of our model are reported.

In our simulated scenario, three cloud providers of Amazon, Microsoft and Rackspace are supposed to be miners and collect their rewards. To make the simulation more realistic, we followed up all the contract transactions from registering in the Cloudchain up to confirmation of the contract completion, depositing the payment and assigning a reputation. Century Link and Alibaba are assumed to cancel their requests for few times after making the contract before acceptation. As Table 2 depicts, the obtained gas consumptions of cloud service requesters are much higher than those that answer these requests and supply these services. This is why Alibaba has the most and Microsoft the least gas consumption.

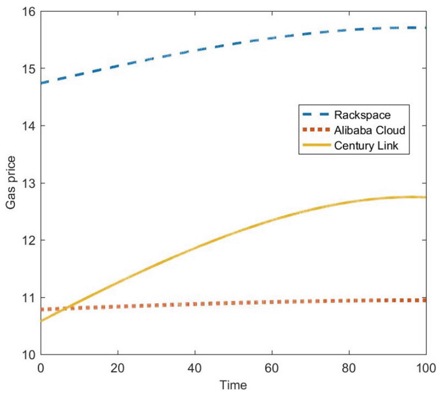

Fig. 2. Gas prices of the three cloud service requesters

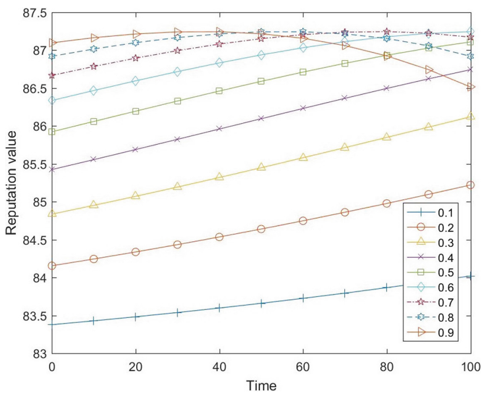

Fig. 3. Microsoft's optimal reputation with different values of $(0.1 \leq \hat{\theta}_r \leq 0.9)$

The gas price of Amazon and Microsoft are considered as constant inputs and they are set to 15. This price guarantees a fast execution of transactions to avoid tarnishing their reputation and will not impose them huge cost due to their minimal gas required as the role of suppliers. To estimate the time delay for each transaction, we tested different prices in different time slots to obtain an

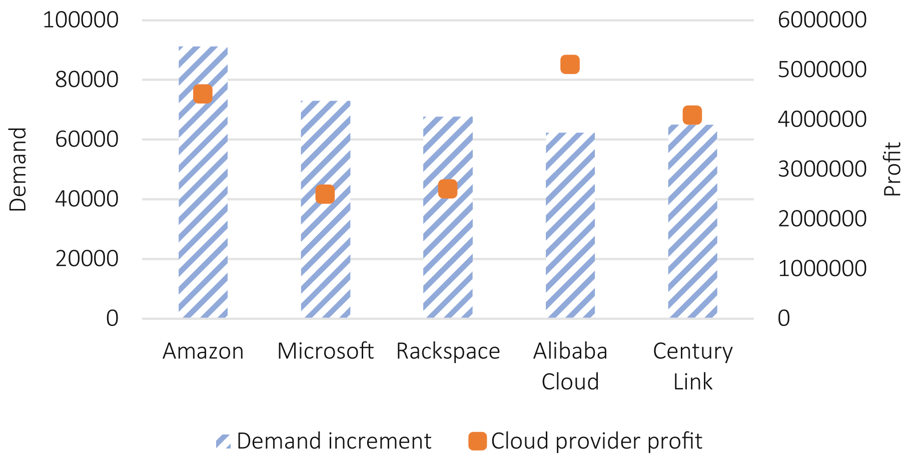

Fig. 4. Average of cloud providers' profit and demands' evolution in Nash Equilibrium

approximate range of delay depending on the traffic of the Ethereum network. The obtained optimal gas prices for the three cloud requesters are shown in Fig. 2. Alibaba has to pay the minimal price, which is almost 11 gwei for the whole period of time. This cloud provider has the highest number of requests, so it is not profitable if it invests more money over gas. With this price, Alibaba has to pay almost \$248.45, at the time of writing this paper. However, because of the cheap gas price, Alibaba has a delay of 27 to 5459 s for each transaction (refer to Table 2). Even though high traffic happens not very often, yet, it would be advisable to predict its demand in advance to avoid the delays that can cause user dissatisfaction. Century Link also has to pay cheap gas price, but not as cheap as Alibaba. It is reasonable since this cloud provider has less gas consumption, higher end-users' prices and lower reputation. To win the users' satisfaction proportional to its service's price, Century Link has to increase the gas price sharply, to speed up the communication and avoid major delays. Based on the results, Rackspace has to pay the highest price for the gas among the cloud requesters. The main reason can be the highest end users' price, the low amount of transactions and gas consumption. The participation in the mining process could also add up to its wealth to afford higher price and higher quality with minimum delays. It worth to note that even though the gas is costly for all cloud providers, it is a one-time cost for a permanent storage.

Figure 3 depicts Microsoft's optimal reputation value during these 100 days as obtained in our experiment. It is worth mentioning that Amazon showed a similar pattern. To investigate the behavior of cloud requesters' demand over these reputation values, we considered the demand rate $\hat{\theta}_r$ varying from 0.1 to 0.9. As the effectiveness of reputation over demands' rate raises, the provider has to aim for a higher reputation at the beginning to earn the eligibility for more demands. However, these optimums do not follow the same trend. In the case of lower effectiveness, the provider has to increase the service quality and gas price leading to a higher reputation over time, but as effectiveness gets intense, the reputation starts to decline. This is where the provider has established its credibility at first and made the major profit halfway through the period, and the increase of reputation is not profitable anymore. This confirms that keeping a high reputation is costly and not always economically justified.

Figure 4 presents a comparative analysis of the average of profit and demands' evolution for the Cloudchain members. The demands' evolution $\dot{D}_r^{'}(t)$ for cloud service suppliers have noticed a higher spike. Yet, interestingly, cloud service requesters have received a higher profit from Cloudchain due to fulfilling their initial demand and selling to their own end-users. The cloud service requesters could obtain cheaper prices from the suppliers and sell at their own prices. However, it should be noted that they can face the risk of not fulfilling their commitments to the end-users if none of the suppliers have the required preserved capacity to rent out. Although it seems that Cloudchain benefits more the cloud requesters, yet it is not true. The main profit of cloud suppliers is from their own market and users, and they only rent the partial idle computing resources, which are not being used. As the number of cloud service requesters elevates, their share of profit from the outsourced demand and the mining rewards increases.

6 Conclusion

In this paper, we introduced a new distributed blockchain-based framework for cloud providers federation to overcome the limitations of conventional centralized federations. Due to the coopetitive environment of Cloudchain, and high expense of public smart contracts, we further designed and solved a differential game. This game modeled the best strategies of cloud providers to make a request with an optimal transaction cost and time, as well as, to optimize their reputation value to receive the requests from other providers. Cloudchain was implemented using Solidity over the Ethereum network and the differential game was simulated for a sample of five cloud providers during 100 days. The findings can be summarized from two perspectives of the cloud service requesters and suppliers. For cloud requesters with a high number of requests, spending high gas price is not economically appealing. With cheaper gas prices, they might face some delays in peak times, which needs to be predicted in advance. Although requesters incurred higher costs from Cloudchain, yet they gained a significantly high income by outsourcing some parts of their customers' demands that could not be fulfilled by their own. The results showed that cloud suppliers have minimal gas consumption, which makes it more affordable for them to pay higher prices and enhance their communication and reputation. Though increasing the reputation was not always the best strategy for highly reputed cloud providers, a gradual increase is recommended when the requesters' demand is not significantly impacted. The end-user's service price is found to be a very decisive factor in deciding the level of quality and gas/reputation values for both of the cloud service requesters and suppliers.

References

1. Azaria, A., Ekblaw, A., Vieira, T., Lippman, A.: MedRec: using blockchain for medical data access and permission management. In: International Conference on Open and Big Data, pp. 25–30 (2016)
2. Bairagi, A.K., Alam, M.G.R., Talukder, A., Nguyen, T.H., Hong, C.S., et al.: An overlapping coalition formation approach to maximize payoffs in cloud computing environment. In: 2016 International Conference on Information Networking, pp. 324–329 (2016)
3. Basar, T., Olsder, G.: Dynamic Noncooperative Game Theory, 2nd edn. Society for Industrial and Applied Mathematics, Philadelphia (1998)
4. Chen, H., An, B., Niyato, D., Soh, Y.C., Miao, C.: Workload factoring and resource sharing via joint vertical and horizontal cloud federation networks. IEEE J. Sel. Areas Commun. **35**(3), 557–570 (2017)
5. Romero Coronado, J.P., Altmann, J.: Model for incentivizing cloud service federation. In: Pham, C., Altmann, J., Bañares, J.Á. (eds.) GECON 2017. LNCS, vol. 10537, pp. 233–246. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68066-8_18
6. Hassan, M.M., Alelaiwi, A., Alamri, A.: A dynamic and efficient coalition formation game in cloud federation for multimedia applications. In: Proceedings of the International Conference on Grid Computing and Applications, p. 71 (2015)
7. Hocking, L.M.: Optimal Control: An Introduction to the Theory with Applications. Oxford University Press, Oxford (1991)
8. Jiao, Y., Wang, P., Niyato, D., Xiong, Z.: Social welfare maximization auction in edge computing resource allocation for mobile blockchain. arXiv preprint arXiv:1710.10595 (2017)
9. Klems, M., Eberhardt, J., Tai, S., Härtlein, S., Buchholz, S., Tidjani, A.: Trustless intermediation in blockchain-based decentralized service marketplaces. In: Maximilien, M., Vallecillo, A., Wang, J., Oriol, M. (eds.) ICSOC 2017. LNCS, vol. 10601, pp. 731–739. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-69035-3_53
10. Kraft, D.: Difficulty control for blockchain-based consensus systems. Peer-To-Peer Netw. Appl. **9**(2), 397–413 (2016)
11. Lee, C.A.: Cloud federation management and beyond: requirements, relevant standards, and gaps. IEEE Cloud Comput. **3**(1), 42–49 (2016)
12. Luu, L., Chu, D.H., Olickel, H., Saxena, P., Hobor, A.: Making smart contracts smarter. In: Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, pp. 254–269 (2016)
13. Mendling, J., et al.: Blockchains for business process management-challenges and opportunities. ACM Trans. Manag. Inf. Syst. **9**(1), 4 (2018)
14. Münsing, E., Mather, J., Moura, S.: Blockchains for decentralized optimization of energy resources in microgrid networks. In: Conference on Control Technology and Applications, pp. 2164–2171 (2017)
15. Niyato, D., Vasilakos, A.V., Kun, Z.: Resource and revenue sharing with coalition formation of cloud providers: game theoretic approach. In: 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, pp. 215–224 (2011)
16. Ray, B., Saha, A., Khatua, S., Roy, S.: Quality and profit assured trusted cloud federation formation: game theory based approach. IEEE Trans. Serv. Comput. (2018). https://doi.org/10.1109/TSC.2018.2833854

17. Szabo, N.: Formalizing and securing relationships on public networks. First Monday **2**(9) (1997). https://doi.org/10.5210/fm.v2i9.548
18. Taghavi, M., Bentahar, J., Otrok, H., Wahab, O.A., Mourad, A.: On the effects of user ratings on the profitability of cloud services. In: International Conference on Web Services (ICWS), pp. 1–8 (2017)
19. Underwood, S.: Blockchain beyond bitcoin. Commun. ACM **59**(11), 15–17 (2016)
20. Xu, X., et al.: A taxonomy of blockchain-based systems for architecture design. In: International Conference on Software Architecture, pp. 243–252 (2017)
21. Zhang, Y., Wen, J.: The IoT electric business model: using blockchain technology for the internet of things. Peer-To-Peer Netw. Appl. **10**(4), 983–994 (2017)