Decentralized Caching for Content Delivery Based on Blockchain: A Game Theoretic Perspective

Wenbo Wang*, Dusit Niyato*, Ping Wang* and Amir Leshem†
*School of Computer Engineering, Nanyang Technological University, Singapore 639798
†Faculty of Engineering, Bar-Ilan University, Ramat-Gan, Israel, 52900

***Abstract*—Blockchains enable tamper-proof, ordered logging for transactional data in a decentralized manner over open-access, overlay peer-to-peer networks. In this paper, we propose a decentralized framework of proactive caching in a hierarchical wireless network based on blockchains. We employ the blockchain-based smart contracts to construct an autonomous content caching market. In the market, the cache helpers are able to autonomously adapt their caching strategies according to the market statistics obtained from the blockchain, and the truthfulness of trustless nodes are financially enforced by smart contract terms. Further, we propose an incentive-compatible consensus mechanism based on proof-of-stake to financially encourage the cache helpers to stay active in service. We model the interaction between the cache helpers and the content providers as a Chinese restaurant game. Based on the theoretical analysis regarding the Nash equilibrium of the game, we propose a decentralized strategy-searching algorithm using sequential best response. The simulation results demonstrate both the efficiency and reliability of the proposed equilibrium searching algorithm.**

***Index Terms*—Caching, blockchain, proof-of-stake, Chinese restaurant game**

I. Introduction

According to the latest Cisco VNI report [1], over the past 5 years the global mobile data traffic has grown 18-fold, which is largely driven by the explosive growth of the on-demand mobile video traffic. In contrast, it is predicted that the mobile network connection speed will only increase 3-fold in the next 5 years. Motivated by the fact that most mobile traffics are asynchronous but repetitive content delivery requests, the research communities have vastly investigated the technique of proactive caching in recent years [2]–[4]. In general, proactive caching consists of two stages. The first stage is the prefetching stage, where the edge servers such as Wi-Fi Access Points (APs) and Small-Cell Base Stations (SCBSs), or even Device-to-Device (D2D)-enabled nodes pre-download/replicate the contents from the Content Servers (CSs) before the content requests are posted by any mobile users. The second stage is the delivery stage, where the contents are delivered to the mobile users via the fronthaul or D2D links from the caching nodes. A recent line of works has shown that proactive caching is able to relieve the backhaul congestion [2], reduce the cost of the content providers [3] and/or improve the Quality of Experience (QoE) of the mobile users [4].

With the goals of offloading the heavy traffic and ensuring delivery in case of high volatility, proactive caching at the APs/SCBSs in wireless networks shares a lot of similarities with the commercial Content Delivery Networks (CDNs) [5]. However, due to the limited resource in storage, computation power and edge-to-end capacity, an AP/SCBS can only afford caching a subset of the contents from the Content Providers (CPs). Then, how to properly select the contents for proactive caching becomes a vital challenge for the edge nodes, namely, Cache Helpers (CHs). Moreover, due to the self-deployment nature of the CHs and the coexistence of multiple CPs, it is impractical to deploy or coordinate the accounting/broking servers which are required by the logical system of a traditional CDN [5] for transaction auditing and traffic estimation. As a result, no trusted entity in the network audits the content access/delivery or enforces the proper payments to the right parties (e.g., CPs and CHs). Therefore, new expectations are set for a decentralized, self-organized proactive caching system:

(1) The brokering and accounting processes between the CPs and CHs should be achieved in a decentralized market, with truthfulness being ensured for both parties.
(2) The CHs should be able to adapt to the changes of content demand/supply in the network without access to a centralized content delivery accounting server.
(3) The processes of cache selection and delivery allocation should remain transparent to the mobile users.
(4) Proper incentive mechanisms should be provided to sustain self-motivated content caching among the APs/SCBSs.

In this paper, we jointly address the challenges above by constructing an autonomous content-caching market with a smart contract-enabled blockchain. The blockchain serves as a public immutable ledger without any centralized mediator in a logical Peer-to-Peer (P2P) network [6]. The transaction records between blockchain users are jointly approved by the consensus nodes and digitally stored in their local blockchain replicas. With the blockchain, the content prefetching and delivering processes are self-organized in the form of smart contracts [7]. The blockchain also provides publicly accessible records about the demand and supply of contents in the network. By jointly considering the CHs' activities in smart contract execution and block consensus maintenance, we adopt a cross-layer design for the blockchain's consensus mechanism based on Proof-of-Stake (PoS) [8]. Compared with the existing centralized caching systems, the key advantages of the proposed mechanism are:

- Without a centralized auditor, the blockchain maintains a publicly auditable transaction record. By querying it, the CHs are able to autonomously learn the market state and adapt their caching strategies accordingly.

- The smart contracts use financial incentive to ensure the interests of different parties in a trustless caching market.
- The proposed consensus protocol financially incentivizes the CHs to stay active in their service while securing the blockchain consistency with little resource consumption.

We model the blockchain-based content caching market as a Chinese restaurant game [9]. Based on the analysis of the Nash Equilibrium (NE) of the game, we propose a decentralized NE algorithm using sequential best response. The simulation results demonstrate the efficiency and reliability of the proposed decentralized NE searching algorithm.

II. Network Model and Design Approach

A. Network Structure

We consider a hierarchical wireless network (e.g., an ultra dense small cell network), where N CPs offer a static, homogeneous catalog of contents, $\mathcal{K}=\{1,\ldots,K\}$, to mobile users. M edge nodes, i.e., the SCBSs and user-deployed Wi-Fi APs, work as the proactive CHs in the network. We assume that the mobile users are indifferent in choosing the services of any CP, and each user randomly subscribes to one of the CPs with a flat-rate subscription payment for the full access to $\mathcal{K}$. We also consider that the frequency of user demands for the contents remains the same for a sufficiently long period. Then, after arranging the contents in $\mathcal{K}$ in a descending order of their popularity, we can use the Zipf distribution to model the probability of a content being requested during a given time [10]:

$$p_k = \frac{k^{-\beta}}{\sum_{k=1}^{K} k^{-\beta}}, \qquad (1)$$

where the coefficient β reflects how skewed the distribution is.

We assume that a CH m chooses to cache only one content from one CP during a certain period, mainly due to the storage/computation limit. In return, the corresponding CP does not pay the CH for merely prefetching contents, but promises to offer reward according to the number of offloaded deliveries made by the CH to the mobile users. We also assume that the price of delivery offloading may differ for each CP. As shown in Figure 1, the transactions occur only from the mobile users to the CPs and from the CPs to the CHs. In this way, the caching process is kept transparent to the mobile users for most of the time, except the final content delivery stage from the CHs to the mobile users.

B. Blockchain for a Distributed Cache-delivery Market

We employ the virtual P2P blockchain network as the backbone of the decentralized content delivery market (see Figure 1). An entity (e.g., a CP, a CH or a mobile user) i in the wireless network maps itself as a node in the blockchain network based on a pair of asymmetric keys, namely, a secret key sk_i and a public key pk_i. Let $\mathcal{H}(\cdot)$ denote a collision-resistant, irreversible hash function. Node i identifies itself on the blockchain by a unique "transaction address", $\mathcal{H}(pk_i)$, using the hashcode of its public key. We consider that all payments are made in blockchain tokens (c.f., Ethers of the Ethereum

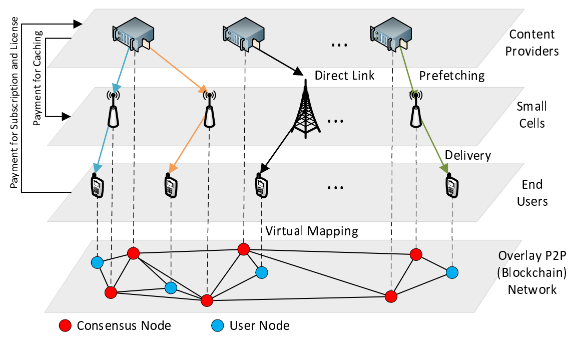


Fig. 1. Illustration of payment flow and blockchain network structure for proactive caching in a hierarchical wireless network.

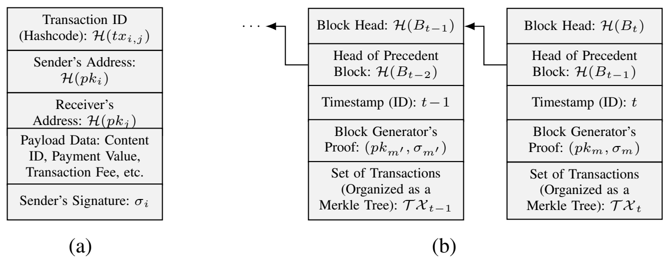

Transaction ID (Hashcode): $\mathcal{H}(tx_{i,j})$
Sender's Address: $\mathcal{H}(pk_i)$
Receiver's Address: $\mathcal{H}(pk_j)$
Payload Data: Content ID, Payment Value, Transaction Fee, etc.
Sender's Signature: σ_i
(a)
Block Head: $\mathcal{H}(B_{t-1})$
Head of Precedent Block: $\mathcal{H}(B_{t-2})$
Timestamp (ID): $t-1$
Block Generator's Proof: $(pk_{m'}, \sigma_{m'})$
Set of Transactions (Organized as a Merkle Tree): $\mathcal{TX}_{t-1}$
Block Head: $\mathcal{H}(B_t)$
Head of Precedent Block: $\mathcal{H}(B_{t-1})$
Timestamp (ID): t
Block Generator's Proof: (pk_m, σ_m)
Set of Transactions (Organized as a Merkle Tree): $\mathcal{TX}_t$
(b)

Fig. 2. Data structures for constructing a blockchain. (a) Data fields in a single transaction. (b) Blockchain as a hash linked list of blocks.

network [7]). Then, the two types of transactions in the caching market can be abstracted as follows (see also Figure 2(a)):

Definition 1 (Transaction). *A transaction $tx_{i,j}$ from node i to node j on the blockchain can be represented by a 4-tuple: $tx_{i,j} = \langle \mathcal{H}(pk_i), \mathcal{H}(pk_j), d_{i,j}, \sigma_i \rangle$, where*

- *$\mathcal{H}(pk_i)$ and $\mathcal{H}(pk_j)$ are the addresses of nodes i and j,*
- *$d_{i,j}$ is the payload data specifying the payment value and the auxiliary information (e.g., the content ID), and*
- *σ_i is the publicly verifiable digital signature of node i using the secret key sk_i, namely, $\sigma_i = Sign(sk_i, d_{i,j})$.*

A node issues a new transaction by broadcasting it to the entire blockchain network with an off-the-shelf P2P protocol (e.g., Whisper in Ethereum [7]). To further maintain a trustworthy and strictly-ordered log of transactions, the blockchain network records the transactions in a chain of "blocks", where each block contains a unique set of transactions. By abstracting away the implementation details, we define the data structure of the block and the blockchain as follows (see also Figure 2(b)):

Definition 2 (Block). *A block B_t can be represented by a 5-tuple: $B_t = \langle t, \mathcal{TX}_t, (pk_m, \sigma_m), H(B_t), H(B_{t-1}) \rangle$, where*

- *t is the ID (i.e., timestamp) of block B_t in the blockchain.*
- *$\mathcal{TX}_t = \{tx_1, \ldots, tx_l\}$ is a set of unique transactions that do not conflict with the transactions in B_0 to B_{t-1}.*
- *(pk_m, σ_m) contains the public key and the signature of node m that publishes the block. $\sigma_m = Sign(sk_m, \mathcal{TX}_t)$.*
- *$H(B_t)$ and $H(B_{t-1})$ are the heads (i.e., hashcodes) of the current block and the precedent block at $t-1$. $H(B_t) = \mathcal{H}(t \| \mathcal{TX}_t \| (pk_m, \sigma_m) \| H(B_{t-1}))$.*

Definition 3 (Blockchain). *A blockchain $\mathcal{C}(t) = \{B_0, \ldots, B_t\}$*

is a sequence (i.e., linked list) of blocks indexed in a strictly increasing time order. B_0 is an arbitrary block known as the genesis block of $\mathcal{C}(t)$, and B_t is known as the head of $\mathcal{C}(t)$.

With a chain of blocks linked by the block heads (i.e., hashcodes), a node have to modify every subsequent block if it wants to forge any transaction in a block stored at the local replica of the blockchain. Therefore, if the network adopts a reliable consensus mechanism to coordinate the states of the local replicas of the blockchain, the blockchain works as a tamper-proof public database of the transactions in the content delivery market [6]. Then, each CH can easily count the frequency of requests for a certain content $k \in \mathcal{K}$ by incrementally querying the blockchain about the related new transactions sent by the mobile users. The CHs may use the observed request frequencies to estimate the content popularity distribution in (1), and thus are able to autonomously choose the contents to cach according to the demand state in the market.

We consider that the CHs and CPs work as the full (consensus) nodes and participate in the consensus process of the blockchain network. The mobile devices work as lightweight nodes and only issue requests for content delivery. Instead of relying on the computation-intensive protocol using Proof of Work (PoW) [6], we adopt a PoS-based consensus mechanism from the Ouroboros protocol [8]. Without loss of generality, we assume that the consensus nodes are equipped with roughly synchronized clocks. A new block is generated in a slotted manner with a fixed time interval. The PoS scheme facilitates an uncompromisable random leader-election process at each time slot to designate a unique consensus node for block generation. To ensure the sustainability of the proactive caching eco-system, we make the following assumptions regarding the token supply mechanism of the blockchain:

- Each CP is assigned a sufficient amount of tokens at B_0.
- New tokens are supplied to the blockchain only through a fixed-amount reward for generating every new block.
- The CPs provide a fixed exchange rate between the tokens and the fiat money for the CHs and the mobile users.
- The transaction fee (c.f., *gas* in Ethereum [7]) is negligible.

From [8], we introduce the formal definition of the leader-election process for block generation with PoS as follows.

Definition 4. *Assume that at time slot s_t the stakes held by the $M+N$ consensus nodes are static and measured by $\mathbf{u}(s_t) = [u_1(s_t), \ldots, u_{M+N}(s_t)]^\top$. A leader-election process consists of a distribution $\mathcal{D}$ and a deterministic function $F(\cdot)$ such that, with a random seed $\rho \leftarrow \mathcal{D}$, $F(\mathbf{u}, \rho, s_t)$ outputs a unique leader index m ($1 \le m \le M+N$) for block generation with probability*

$$p_m^{win}(s_t) = \frac{u_m(s_t)}{\sum_{i=1}^{M+N} u_i(s_t)}. \qquad (2)$$

The random variable $m \leftarrow F(\mathcal{S}, \rho, s_t)$ is independent of s_t.

The PoS-based election scheme can be implemented by a standard Follow-the-Satoshi (FtS) algorithm [8]. The FtS resembles the process of biased coin-tossing and randomly selects a leader's address. This is achieved through indexing a sub-set of tokens controlled by the consensus nodes (i.e, stakeholders) and tracking the owner's address of a random token index[1]. The randomness of token selection is guaranteed by the seeding function $\rho \leftarrow \mathcal{D}$, which is implemented using the hashcode of the precedent block head, i.e., $\sigma \leftarrow \mathcal{H}(B_{s_t-1})$. Here, $\mathcal{H}(\cdot)$ is treated as a trusted uniform random oracle [8] and σ is used to determine the random token index (e.g., with a simple modulo operation). When all the consensus nodes are honest, the proposed PoS scheme is able to avoid the inherent inefficiency of PoW due to "block mining" competition [6] such as computational resource consumption and block orphaning.

Let an *epoch* define a fixed time interval consisting of T time slots. Instead of using the balance of each stakeholder as its stake in (2) (c.f., [8]), we measure the stake of each CH by its delivery reward collected in the latest epoch. Such design incentivizes the CHs to stay active online, since a CH no longer receives any profit by only holding the tokens. On the other hand, we consider that the CPs' stakes are proportional to the amount of unconfirmed payment for content delivery in the last epoch. When the CHs fail to complete the delivery tasks or when pending transactions accumulate due to the Denial of Service (DoS) attacks by malicious CHs, such design ensures the CPs to take control of the block generation process with high probability. Thereby, the blockchain lays the foundation for implementing a secured decentralized caching market.

C. Caching and Delivery Based on Smart Contracts

Now, we address the issues of autonomous content delivery and truthfulness enforcement with smart contracts [7]. From Section II, we identify two stages, namely, the prefetching and the delivering stages, in the content delivery process and implement for each stage a group of smart contracts [11]. At the prefetching stage, the smart contract is used by the CPs and CHs for negotiating about the assignment of offloaded contents. As shown in Figure 3, a CP n posts for each content $k \in \mathcal{K}$ an *order for caching* by deploying a corresponding smart contract, which specifies the price, $o_{n,k}$, for one successfully offloaded delivery in the future. An interested CH m may respond to CP n by calling the function *offer for caching* for content k and sending a deposit to the smart contract. Consequently, an event *response* is fired by the smart contract to notify CP n about the response from CH m. We consider that a CP is able to register more than one CHs for the delivery task (see the "crowdfund contract" in Ethereum [11] for example). CP n confirms to choose CH m by calling a registering function of the smart contract. Then, it transfers a copy of content k to CH m with third-party methods. To get refunded, CH m has to provide CP n an interactive Proof of Retrievability (PoR) [12] for content k in the form of a series of Merkle proofs [12]. After verifying the PoR, CP n orders to refund the deposit to CH m. At the end of each epoch, the existing contracts are destroyed by the

[1]FtS can be performed in advance at $s_t - 1$ by a smart contract through executing a random search in the Merkle tree of the related stake deposits. See https://github.com/Realiserad/fts-tree for a simplified implementation.

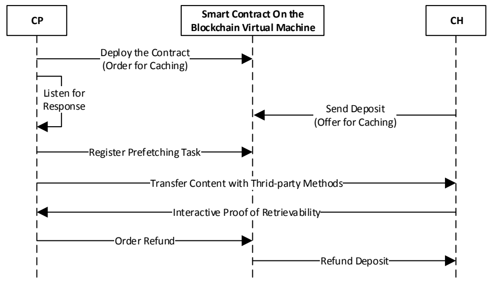

CP
Smart Contract On the Blockchain Virtual Machine
CH
Deploy the Contract (Order for Caching)
Listen for Response
Send Deposit (Offer for Caching)
Register Prefetching Task
Transfer Content with Thrid-party Methods
Interactive Proof of Retrievability
Order Refund
Refund Deposit

Fig. 3. Sequence diagram of the content prefetching contract.

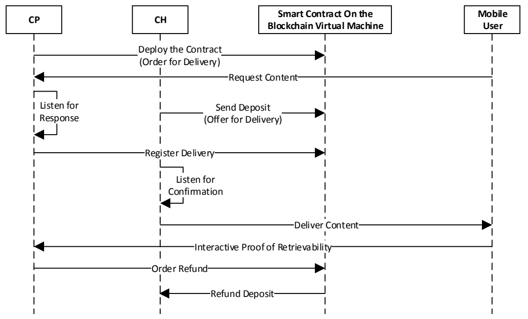

CP
CH
Smart Contract On the Blockchain Virtual Machine
Mobile User
Deploy the Contract (Order for Delivery)
Request Content
Listen for Response
Send Deposit (Offer for Delivery)
Register Delivery
Listen for Confirmation
Deliver Content
Interactive Proof of Retrievability
Order Refund
Refund Deposit

Fig. 4. Sequence diagram of the content delivery contract.

CPs to avoid unresolved transactions. A set of new contracts in the same form are deployed at the beginning of the next epoch.

At the delivering stage, CP n posts for each content $k \in \mathcal{K}$ an *order for delivery* by deploying a smart contract, which works as an escrow account to ensure that the payment by CP n is made only when the delivery is successful. When a CH m responds by calling the function *offer for delivery*, it sends a security deposit to the smart contract. CP n evenly distribute the delivery tasks among the responding CHs and notifies the smart contract to register each CH m for the delivery task. On the other hand, the payment of CP n for the delivery, $o_{n,k}$, is also held by the smart contract until the target mobile user provides the interactive PoR of content k to CP n. To prevent the CHs and the flat-rate mobile users jointly cheating for the delivery reward without actually delivering the content, the smart contracts require that the mobile users provide the PoR within a certain delay to confirm the content reception. Otherwise, the contract will rollback and CP n will receive CH m's security deposit as a punishment. Such design ensures that without a locally stored content copy, a mobile user is not able to obtain the PoR from the CH due to transmission delay. Finally, the content delivery contract is also updated at the beginning of each epoch to remove the unfinished transactions.

III. Blockchain-Based Content Caching Game

A. Caching Strategies with PoS-Based Block Generation

We consider that during each epoch, a total number of L users request content delivery in the network. Since the users are indifferent in subscribing to any CP for the same content service, from (1), the expected number of requests for content k in CP n during the epoch can be written as $L(n,k)=LN^{-1}p_k$. Assume that a subset of CHs, $\mathcal{M}_{n,k}$, have cached the ready-to-deliver copies of content k provided by CP n. Again, if each CP evenly distributes the delivery tasks among the responding CHs, the expected number of delivery tasks assigned to CH $m \in \mathcal{M}_{n,k}$ can be expressed as $L_m(n,k) = LN^{-1}M^{-1}_{n,k}p_k$, where $M_{n,k}$ is the size of $\mathcal{M}_{n,k}$ and $\sum_{n=1}^{N}\sum_{k=1}^{K} M_{n,k} = M$. By adopting the strategy of best-effort delivery, CH m offloads all the delivery tasks assigned by CP n. Then, for a given sub-set of CHs caching content k for CP n, $\mathcal{M}_{n,k}$, the expected delivery reward received by CH m ($m \in \mathcal{M}_{n,k}$) can be expressed as:

$$r^d_m(n,k) = o_{n,k}L_m(n,k) = o_{n,k}LN^{-1}M^{-1}_{n,k}p_k. \quad (3)$$

If CH m is selected as the block generation leader at one epoch, it also receives a fixed reward for generating T new blocks. Let λ denote the total block generation reward in an epoch. According to Section II-B, the probability for CH m to be elected in the next epoch is determined by the ratio between its own delivery reward and the sum of the delivery reward collected by all the CHs in the current epoch. By (2) and (3), for CH m caching content k from CP n, the expected block generation reward to be collected in the next epoch is

$$r^m_m(n,k) = \lambda \frac{o_{n,k}L_m(n,k)}{\sum_{i=1}^{N}\sum_{j=1}^{K}\sum_{l\in\mathcal{M}(i,j)} o_{i,j}L_l(i,j)}, \quad (4)$$

where $L_l(i,j)$ is the expected number of tasks assigned to CH l for delivering content j for CP i.

Now, we can define the action of CH m in an epoch as the pair of the selected CP and content, i.e., $a_m = (n,k)$. We note that the expected reward provided by CP n for delivering content k is evenly distributed to the set of corresponding CHs, $\mathcal{M}_{n,k}$. Then, the payoff for CH m to take action $a_m = (n,k)$ can be obtained by adding up (3) and (4) as follows:

$$r_m(a_m) = \frac{o_{n,k}Lp_k}{NM_{n,k}} + \lambda \frac{o_{n,k}p_k}{M_{n,k}\sum_{i=1}^{N}\sum_{j=1}^{K} o_{i,j}p_j}, \quad (5)$$

where owning to the adjusted stakes based on the unspent reward held by the CPs, $\sum_{i=1}^{N}\sum_{j=1}^{K} o_{i,j}p_j$ is independent of $M_{i,j}$. Following the proposed mechanism of stake evaluation, the payoff of a CH in (5) only depends on the joint action of the CHs at the current epoch. Thereby, we are ready to investigate the caching strategy selection of the CHs through modeling the caching market as a non-cooperative game.

B. Caching Market as a Chinese Restaurant Game

Let $\mathcal{A} = \{(n,k) : 1 \le n \le N, 1 \le k \le K\}$ denote the set of actions (i.e., CP-content choices) for CH $m \in \mathcal{M} = \{m : 1 \le m \le M\}$, and $\mathbf{a} = [a_1, \ldots, a_M]^\top$ denote the vector of the CHs' joint actions. We can define the non-cooperative caching game among the CHs as a 3-tuple: $\mathcal{G} = \langle \mathcal{M}, \mathcal{A}, \{r_m(\mathbf{a})\}_{m\in\mathcal{M}, a_m\in\mathcal{A}} \rangle$. We define the *grouping of actions* as the vector of the numbers of the CHs choosing the same action given a joint action $\mathbf{a}$: $\mathbf{g}_{\mathcal{M}}(\mathbf{a}) = [M_{1,1}, \ldots, M_a, \ldots, M_{N,K}]^\top$. From (5), we define the *reward potential* of each action as $R_{n,k} = o_{n,k}p_k$. Then, for CH m choosing $a = (n,k)$, when fixing the other CHs' actions, we can rewrite (5) as:

$$r_m(a) = r(R_a, M_a) = \frac{L}{N}\frac{R_a}{M_a} + \frac{\lambda}{\sum_{i=1}^{N}\sum_{j=1}^{K} R_{i,j}}\frac{R_a}{M_a}. \quad (6)$$

Since $r(R_a, M_a)$ is a decreasing function of M_a, M_a reflects the impact of the negative network effect on CH m's payoff for choosing action a. By inspecting the first order derivative of $r(R_a, M_a)$ with respect to R_a, we can easily show that $r(R_a, M_a)$ is an increasing function of R_a. Then, game $\mathcal{G}$ is a typical Chinese restaurant game [9], where $\mathcal{A}$ can be compared to the table set in a restaurant, and R_a and M_a can be compared to the size and the customer number of table a, respectively.

Let a_{-m} denote the adversaries' actions with respect to CH m. Let $\mathbf{g}_{\mathcal{M}}(a_{-m}) = [M_{1,1}^{-m}, \ldots, M_{N,K}^{-m}]^\top$ denote the outcome of action grouping for a_{-m}, where $M_{n,k}^{-m}$ is the number of CHs except CH m choosing action (n, k). Then, we can define the Best Response (BR) of CH m to a_{-m} as:

$$BR_m(a_{-m}) = BR_m\left(\mathbf{g}_{\mathcal{M}}(a_{-m})\right) = \arg\max_{a \in \mathcal{A}} r(R_a, M_a^{-m} + 1). \tag{7}$$

Based on (7), we can define the NE of game $\mathcal{G}$ in the form of the simultaneous best response as follows.

Definition 5. *A joint CH action $\mathbf{a}^* = (a_m^*, a_{-m}^*)$ is an NE of game $\mathcal{G}$, if $\forall m \in \mathcal{M}$, the following holds:*

$$a_m^* = BR_m\left(\mathbf{g}_{\mathcal{M}}^*(a_{-m})\right) = \arg\max_{a \in \mathcal{A}} r(R_a, M_a^{*,-m} + 1), \tag{8}$$

where $\mathbf{g}_{\mathcal{M}}^*(a_{-m})$ is the outcome of action grouping for a_{-m}^*.

C. Nash Equilibrium of the Caching Game

After reformulating the payoff of a CH m with action $a = (n, k)$ as a function of R_a and M_a in (6), we note that the CHs choosing the same action receive an identical payoff. Then, we can show that game $\mathcal{G}$ is an exact potential game.

Theorem 1. *$\mathcal{G}$ is an exact potential game and admits at least one pure-strategy NE.*

Proof. We define the following potential function (i.e., Rosenthal's potential function [13]) of $\mathbf{a} = (a_m, a_{-m})$ in game $\mathcal{G}$:

$$\phi(\mathbf{a}) = \sum_{a \in \mathcal{A}} \sum_{i=1}^{M_a} r(R_a, i), \tag{9}$$

where M_a is given by the action grouping $\mathbf{g}_{\mathcal{M}}$ of $\mathbf{a} = (a_m = a, a_{-m})$. If CH m unilaterally switches its action from $a_m = (n, k)$ to $a'_m = (n', k')$, the change only affects the CHs that adopt actions a_m and a'_m. Thus, we obtain a new grouping of actions $\mathbf{g}_{\mathcal{M}}(\mathbf{a}') = [M'_{1,1}, \ldots, M'_{N,K}]^\top$, where the only difference from $\mathbf{g}_{\mathcal{M}}(\mathbf{a})$ is that $M_{a_m} = M'_{a_m} + 1$ and $M_{a'_m} = M'_{a'_m} - 1$. Then, we obtain

$$\begin{aligned}
&\phi(a_m, a_{-m}) - \phi(a'_m, a_{-m}) \\
&= \left(\sum_{i=1}^{M_{a_m}} r(R_{a_m}, i) + \sum_{i=1}^{M_{a'_m}} r(R_{a'_m}, i)\right) - \left(\sum_{i=1}^{M_{a_m}-1} r(R_{a_m}, i)\right. \\
&+ \left.\sum_{i=1}^{M_{a'_m}+1} r(R_{a'_m}, i)\right) = r(R_{a_m}, M_{a_m}) - r(R_{a'_m}, M_{a'_m} + 1) \\
&= r_m(a_m, a_{-m}) - r_m(a'_m, a_{-m}).
\end{aligned} \tag{10}$$

By Definition 52 in [13], game $\mathcal{G}$ is an exact potential game. Then, Theorem 1 immediately follows Theorem 53 in [13]. □

Algorithm 1 Equilibrium Strategy Searching

```
Require: Randomly initialize $\mathbf{a}(0) = [a_1(0), \ldots, a_m(0)]^\top$;
1: while $\mathbf{a}(t)$ not converged do
2:   for CH $m \in \mathcal{M}$ do
3:     $r_m \leftarrow 0$;
4:     for $a \in \mathcal{A}$ do
5:       if $r(R_a, M_a^{-m} + 1) > r_m$ then
6:         $a_m \leftarrow a$, $r_m \leftarrow r(R_a, M_a^{-m} + 1)$;
7:       end if
8:     end for
9:     $a_m(t) \leftarrow a_m$;
10:  end for
11:  $t \leftarrow t + 1$;
12: end while
```

Corollary 1. *For game $\mathcal{G}$, the asynchronous/sequential best-response dynamics converge with probability one to a pure NE.*

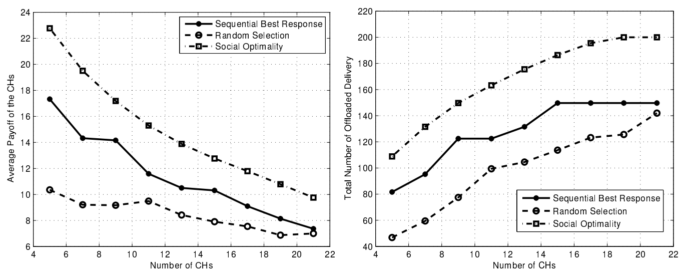
Proof. Given Theorem 1, namely, $\mathcal{G}$ is an exact potential game, Corollary 1 follows Theorem 143 in [13]. □

Considering the asynchronous nature of the caching choice selection based on the blockchain, we propose in Algorithm 1 the asynchronous best-response algorithm for equilibrium strategy searching in the content market game $\mathcal{G}$. By Corollary 1, the proposed strategy searching scheme in Algorithm 1 is guaranteed to converge to a pure-strategy NE.

IV. Simulation Results

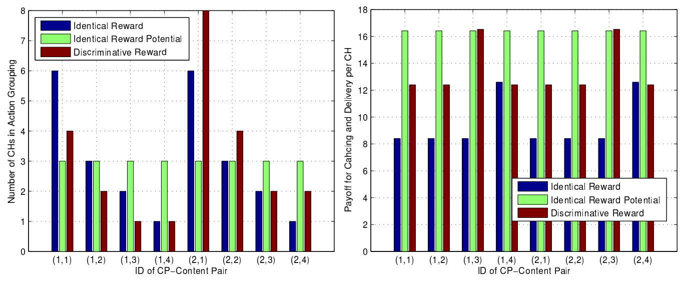
We first consider a network of $N = 3$ CPs with a catalog of $K = 6$ contents. There are $L = 200$ mobile users in the network and the skewness of the popularity distribution in (1) is set to $\beta = 1$. We first consider that the delivery rewards are identical for all the CP-content pairs. In Figure 5, we compare the performance of the proposed best-response strategy searching scheme in Algorithm 1 with that of the random content selection and that of the centralized payoff optimization, respectively. The result for random content selection is obtained with Monte Carlo simulation. As we can observe in Figures 5(a) and 5(b), the performance of the proposed caching strategy searching algorithm significantly outperforms that of random content selection in both the CHs' average payoff and the total number of offloaded deliveries. However, the gap of performance between the centralized optimum and the proposed algorithm indicates that the "pure price of anarchy" is unnegligable for the caching game $\mathcal{G}$. Theoretically, from the individual payoff function given by (6), we can re-interpret the caching game $\mathcal{G}$ as a singleton congestion game [13]. Then, without coordination, the CHs may be reluctant to offer caching services to the CP-content pairs that pay less for delivery, although by doing so they are able to increase the social welfare of the CHs. Instead, sharing the delivery demands with other CHs for the CP-content pairs with higher reward potential will lead to a higher individual payoff.

In Figure 6, we study the impact of the CPs' rewarding scheme on the CHs' NE strategies. For ease of exposition, we



(a) (b)

Fig. 5. (a) Average payoff of the CHs vs. number of CHs in the network. (b) Number of offloaded content deliveries to the CHs vs. number of CHs.



(a) (b)

Fig. 6. (a) Action grouping results with different rewarding schemes. (b) Payoff comparison of action groups with different rewarding schemes.

consider a network of 2 CPs with 4 contents, 24 CHs and 200 mobile users. We consider 3 pricing schemes: (a) the CPs offering uniform reward for each content, i.e., $o_{n,k}=o_{n',k'}$ for all feasible n, n', k, k' (the blue bars in Figure 6); (b) the CPs offering different reward to make the reward potential identical, namely, $o_{n,k}p_k=o_{n',k'}p_{k'}$ for all feasible n, n', k, k' (the green bars in Figure 6); (c) discriminative reward with $o_{2,k}=2o_{1,k}$ for all feasible k (the red bars in Figure 6). As shown in Figure 6, the reward pricing scheme with identical reward potential is able to achieve better fairness among the CHs, and the CHs are more willing to serve the caching and delivery tasks for the less popular contents. On the other hand, by increasing the level of delivery reward, a CP is able to attract more CHs for its own traffic offloading (see the red bars in Figures 6(a)). We also note in Figure 6(b) that when one CP increases its delivery reward, the payoffs of the CHs at the NE will ultimately increase. This indicates that the CPs are at a higher cost if one of them unilaterally tries to improve its delivery efficiency.

V. Conclusion

We have designed an decentralized proactive caching system in a hierarchical wireless network based on blockchains, which enables the cache helpers to autonomously adapt their strategies without a centralized auditing body. We have designed several smart contacts to enable autonomous caching-delivery task assignment and enforce truthfulness of different parties in the network. We have designed an incentive-compatible consensus mechanism for the blockchain based on proof-of-stake to encourage the cache helpers to stay active in service online. We have modeled the caching system as a Chinese restaurant game and further shown that it is an exact potential game. We have proposed a decentralized strategy searching algorithm based on asynchronous best response for the cache helpers to learn their NE strategies. The numerical simulation results have demonstrated both the efficiency and the reliability of the proposed proactive caching scheme.

VI. Acknowledgement

This work was supported in part by Singapore MOE Tier 1 under Grant 2017-T1-002-007, MOE Tier 2 under Grant MOE2014-T2-2-015 ARC4/15, NRF2015-NRF-ISF001-2277 and ISF-NRF grant number 2277/16.

References

[1] Cisco, "Cisco visual networking index: Global mobile data traffic forecast update, 2016-2021," Tech. Rep., Mar. 2017.

[2] B. Chen, C. Yang, and A. F. Molisch, "Cache-enabled device-to-device communications: Offloading gain and energy cost," *IEEE Transactions on Wireless Communications*, vol. 16, no. 7, pp. 4519–4536, Jul. 2017.

[3] J. Tadrous and A. Eryilmaz, "On optimal proactive caching for mobile networks with demand uncertainties," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2715–2727, Oct. 2016.

[4] M. Dehghan, B. Jiang, A. Seetharam, T. He, T. Salonidis, J. Kurose, D. Towsley, and R. Sitaraman, "On the complexity of optimal request routing and content caching in heterogeneous cache networks," *IEEE/ACM Transactions on Networking*, vol. 25, no. 3, pp. 1635–1648, Jun. 2017.

[5] A.-M. K. Pathan and R. Buyya, "A taxonomy and survey of content delivery networks," Grid Computing Distributed System Lab, University of Melbourne, Melbourne, Australia,, Tech. Rep., 2007.

[6] F. Tschorsch and B. Scheuermann, "Bitcoin and beyond: A technical survey on decentralized digital currencies," *IEEE Communications Surveys Tutorials*, vol. 18, no. 3, pp. 2084–2123, third quarter 2016.

[7] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger (eip-150 revision)," *Ethereum Project Yellow Paper*, vol. 151, 2017.

[8] A. Kiayias, A. Russell, B. David, and R. Oliynykov, "Ouroboros: A provably secure proof-of-stake blockchain protocol," in *CRYPTO 2017: 37th Annual International Cryptology Conference, Part I*, Santa Barbara, CA, Aug. 2017, pp. 357–388.

[9] C. Y. Wang, Y. Chen, and K. J. R. Liu, "Chinese restaurant game," *IEEE Signal Processing Letters*, vol. 19, no. 12, pp. 898–901, Dec. 2012.

[10] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and zipf-like distributions: evidence and implications," in *Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 1, Mar. 1999, pp. 126–134 vol.1.

[11] "Solidity documentation, release 0.4.20," Ethereum, Tech. Rep., Jan. 2018. [Online]. Available: http://solidity.readthedocs.io/en/latest/

[12] H. Kopp, D. Mdinger, F. Hauck, F. Kargl, and C. Bsch, "Design of a privacy-preserving decentralized file storage with financial incentives," in *2017 IEEE European Symposium on Security and Privacy Workshops (EuroS PW)*, Paris, France, Apr. 2017, pp. 14–22.

[13] S. Lasaulce and H. Tembine, *Game theory and learning for wireless networks: fundamentals and applications*. Academic Press, 2011.