CryptoArcade: A Cloud Gaming System With Blockchain-Based Token Economy

Sizheng Fan, *Graduate Student Member, IEEE*, Juntao Zhao, Rong Zhao, Zehua Wang, *Member, IEEE*, and Wei Cai, *Senior Member, IEEE*

Abstract—Cloud gaming is a novel service provisioning technology that offloads parts of game software from terminals to powerful cloud infrastructures. However, the commercial charging model for cloud gaming is still in its infancy. In this article, we reveal the deficiencies of existing cloud gaming pricing models and propose CryptoArcade, a token-based cloud gaming system that adopts cryptocurrency as a payment method. Using cryptocurrency, CryptoArcade provides a transparent and resource-aware pricing method, enabling a time irrelevant silent payment on the floating price to protect players' interests, which avoids the Quality of Experience (QoE) degradation caused by traditional dynamic models. While CryptoArcade can solve the problem of pricing strategies, players still face decision headaches caused by having commission overhead and pre-deposit amounts on blockchains. To better understand players' trading behaviors in this decision-making, we consider a marketplace where players trade tokens through smart contracts before gaming sessions. Considering the uncertainty of future token consumption, we use Prospect Theory (PT) in modeling and obtain the optimal solution in closed form. When comparing with the benchmark expect utility theory (EUT), we show that with the same external factors, EUT players are more likely to buy tokens than PT ones.

Index Terms—Cloud gaming, pricing, blockchain, token, prospect theory

1 Introduction

Cloud gaming, services that offload the game programs from the traditional consoles to the cloud, executes the core game logic and game runtime on the cloud and conveys the game content to the players via video stream, which reduces the hardware resource requirement in the thin clients. We are now getting a more solid version of the cloud gaming future landscape from the recent announcement of several big companies. During the Game Developers Conference (GDC) 2019 conference, Google offered Stadia, a cross-platform cloud gaming platform, aiming to provide cloud gaming service through the browser. Meanwhile, Tencent Cloud released its cloud gaming solution at ChinaJoy 2019. Recently, Oppo provided a cloud gaming experience over 5G at Mobile World Congress (MWC) 2019, while Microsoft will also test the xCloud game streaming service in Korea over the 5G soon. Forsaken World, the new massively multiplayer online role-playing game (MMORPG) from Perfect World, also launched a cloud version on China Telecom's cloud gaming platform in 2020. Worldwide game and tech firms are exploring cloud gaming as a new way to deliver game services, and the dawn of 5G provided solutions to the pain point of network problems faced with cloud gaming in the past few years, which also fueled up this field.

Extensively studies have been conducted to optimize cloud gaming services, including graphical rendering [2], edge allocation [3], bandwidth allocation [4], server resource management [5], and dynamic streaming [6]. In contrast, few researchers investigated novel cloud gaming pricing strategies, which adopt playing time as their pricing criteria. The existing cloud gaming pricing strategy follows a traditional *time granularity pricing* in other cloud computing services. For example, PlayStation Now[1], the most popular operating cloud gaming platform, charges its customers with a monthly subscription policy. The players need to pay the subscription fee in advance at the beginning of a month to access their cloud gaming services. However, this method implies a high pre-paid price, which means the players need to play sufficient time to make their payment worthwhile. Therefore, the players with high service stickiness may benefit from the monthly subscription, while others may suffer from over-pay loss because of their limited playing time. At the same time, the subscription needs resource provision for all the subscribed players on a large time scale, which leads to cloud computing resources idle and wasted. Another method is the spot price, as dynamic pricing is used in cloud gaming, solving the previous issues nicely in many

- *Sizheng Fan, Juntao Zhao, Rong Zhao, and Wei Cai are with the School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen, Guangdong 518172, China, and also with the Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen 518129, China. E-mail: {sizhengfan, juntaozhao, rongzhao}@link.cuhk.edu.cn, caiwei@cuhk.edu.cn.*
- *Zehua Wang is with the Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, BC V6T 2G9, Canada. E-mail: zwang@ece.ubc.ca.*

Manuscript received 25 February 2022; revised 22 September 2022; accepted 24 September 2022. Date of publication 0 2022; date of current version 0 2022.
This work was supported by Shenzhen Science and Technology Program (Grant No. JCYJ20210324124205016).
(Corresponding author: Wei Cai.)
Recommended for acceptance by P. Spirakis.
This article has supplementary downloadable material available at https://doi.org/10.1109/TCC.2022.3210013, provided by the authors.
Digital Object Identifier no. 10.1109/TCC.2022.3210013

1. https://www.playstation.com/en-us/explore/playstation-now/

discontinuous computing services, which is fine-grained and sensible to the market demand. For example, Parsec[2] applies an hourly spot pricing model, where the players pay \$0.5 to \$0.8 per hour according to the host. However, new issues emerged in cloud gaming when using the spot price. Cloud gaming services have high requirements for the quality-of-service experience over a long continuous time. Floating prices will directly force players to change service demands, which may devastate players' service experience. For example, if a silent payment method is used, the players will be charged pay-as-you-go with a floating price. This will make players concerned about their payment during the gaming session, as they need to estimate the current service price and their balance to determine how much time they should or will play. If using payment requests instead, as the fine-grained pricing model requires a cost based on a certain time unit, the frequent payment requests will also affect the players' gaming experience. Another problem with the spot price is that the floating price in the payment process will be non-transparent, which introduces price discrimination risks to the players. As players can only have a rough estimation of the service price, the transparent problem in payment allows the service provider to arbitrarily control the price with no protection for players' utility. Hence, none of these pricing strategies is good in practice [7], [8].

To mitigate the above issues, a new business model or pricing strategy should be established. First, the newly proposed model should protect the user's payment, no matter what they already paid or will pay in the future. This requires the new model to keep the paid value while making the payment process transparent. Second, to utilize the computing resource, the service price of this new model should be floated with the market demand. Third, to protect the game experience from the worry of price and interruption of the payment request, a silent payment way on floating price without concern should be applied.

Motivated by the tokenization and transparency of the blockchain, we propose and implement *CryptoArcade* by borrowing the idea from the traditional amusement arcade, which installs coin-operated machines to provide the cloud gaming service. Specifically, it is a novel cloud gaming system that employs cryptocurrency as the coin, a.k.a. token, to start the cloud gaming service, which consists of two parts: *token issue protocol* and *CloudAracde*. Token issue protocol, a smart contract deployed by the service provider (SP) on the blockchain, can enable automatic price determination and an autonomous liquidity mechanism for tokens. Players can buy or sell tokens for participating in cloud gaming by calling it. Unlike the time-based rental in traditional cloud pricing models, CloudArcade sells gaming content not by the length of gaming periods but by challenging opportunities (e.g., a limited three lives in Contra). The tokens store the payment value during the exchange and service purchase process. Players can consume tokens with their needs, thus, the over-paid problem caused by the coarse granularity pricing can be solved. From the players' perspective, arbitrary price manipulation by SPs can be prevented because transactions are transparent and traceable on the blockchain.

At the same time, CryptoArcade publishes the price of a game on smart contracts and represents the price with a relatively constant number of tokens, which is determined by the game content and estimated demanded resources. Since the instant price of a token directly reflects the number of tokens in circulation, the actual price for the game will be a dynamic index of market demand. As the token price will silently manipulate the players' purchase behavior, CryptoArcade can leverage it to optimize the resource consumption of the cloud gaming system. The pricing scheme in CryptoArcade is not related to the gaming period. For example, if you buy ten tokens in advance, it will still be ten tokens after minutes. Also, as token stores, a floating price using tokens to pay the service reflects the state of market conditions. Thus, the time anxiety and disturbance introduced by the spot model can be eliminated, promoting the player's gaming experience. To overcome the performance issue of the blockchain when players use tokens to purchase cloud gaming services, we also integrate the payment channel technique to provide players with more credible, lower-cost, and higher-frequency transactions.

Although the CryptoArcade can solve the problems of pricing strategies in cloud gaming, players confront new decision-making problems. 1) *High commission*: Purchasing tokens on blockchains requires a significant commission, known as the gas fee[3] that is not related to the number of tokens purchased at one time. 2) *Advance payment*: Players need to attach enough tokens to the pre-deployed payment channel smart contract before the cloud gaming service to ensure that the game does not end due to a lack of tokens. Therefore, before starting a cloud gaming on CryptoArcade, players need to make a careful consideration on how many tokens to buy or sell at a time.[4]

Located at the player's premises, we focus on one player's trading behavior under the future token consumption uncertainty, given the current token price and quantities of remaining tokens in her/his wallet. Specifically, we need to solve how many token she/he sells or buys to maximize her/his utility? To answer the above question, we calculate the maximum expected utility for holding the different number of tokens, considering his future consumption uncertainty, the current token price, and gas fee[10]. However, substantial empirical evidence has indicated that predictions based on Expected Utility Theory (EUT) can be significantly inconsistent with observations from reality due to the psychological complexity in humans' decision-making mechanisms. Hence, prospect theory (PT) has been proposed to provide a user-centric view to address this issue, considering three significant aspects: reference points, asymmetric value function, and probability distortion [11]. To better understand the players' trading behaviors before participating in CryptoArcade, we formulate the trading decision problem as an optimization problem, where the

2. https://parsecgaming.com/

3. Because of the boom of decentralized finance (DeFi) since 2020, the evolution of gas price during the second half of 2020 has been increasing in an unprecedented manner. For example, in September 2019, the average price was 0.0225ETH (\$4.8 at the time), and one year later, it was 0.193ETH (\$74.9 at the time) [9].

4. Players can purchase multiple tokens at once to reduce the number of purchases and thus reduce the gas fee consumed during the purchase process.

player will decide her/his selling or purchasing token quantity. Moreover, We discuss and compare the practical insights by comparing the analysis under PT and EUT. The significant contributions of this paper are shown as follows:

- *System Design and Implementation*: We propose and implement the first cloud gaming system named CryptoArcade. Specifically, we adopt cryptocurrency to solve the problem of traditional time granularity pricing and adopt the special silent payment method to protect players' game experience while utilizing computing resources. In addition, we leverage the payment channel to address the performance issues of the blockchain, providing low-cost, faster transactions between players and the SP.
- *Prospect Theory-based player behavior model and analysis*: Due to the uncertainty of future token demand faced by players in CryptoArcade when they buy or sell tokens, we model players' behavior based on prospect theory considering the effect of token price and gas fee on both EUT and PT players' strategies. Compared with the benchmark EUT, PT players are more likely to buy tokens under the same external conditions.

The rest of this paper is organized as follows. We review related work in Section 2 and illustrate the overview and design of the proposed cloud gaming system in Section 3. The EUT and PT player's behavior models are presented in Section 4. We then formulate and solve the optimization problem in Section 5. Afterward, we illustrate the system implementation and numerically evaluate the sensitivity of the player's optimal decision for several model parameters in Section 6. Finally, section 7 concludes this paper.

2 Related Work

2.1 QoE of Cloud Gaming and Dynamic Pricing

As a kind of cutting-edge cloud computing paradigm, cloud gaming shows promise to the economic landscape of computing. The pricing is a critical issue for cloud gaming because it directly affects players' budgets, influencing players' QoE [12]. Though static pricing is the dominant strategy today, dynamic pricing has been widely studied and discussed in the past few years. It tries to solve the problems in static pricing by adjusting prices according to the demands in the cloud service market. Dynamic pricing schemes such as real-time pricing, auction-based pricing, and job scheduling pricing are discussed and proposed [13], which are adopted in some real-world applications, including cloud computing [14], [15], [16], edge computing [17], [18], and power control [19]. These pricing schemes usually develop a floating price algorithm based on both users and service providers and indirectly use price to manage the demands on the users' side, leading to a devastation of users' service experience. Meanwhile, due to opaque pricing, service providers can manipulate prices to gain more significant benefits. A typical example is the spot price proposed by the Amazon Web Service (AWS) [20]. Xu et al. [12] conduct an empirical study on Amazon's spot price history to show that, in contrast to the common belief [21], Amazon's spot price is unlikely to be set according to market supply and demand. Rather, price oscillates within a narrow band most of the time, which is more likely to be controlled by Amazon.

Here, we apply cryptocurrency to CryptoArcade mainly because it fulfills our critical needs: 1) its price reflects the demand in the market, which is not controlled by the companies; 2) it provides a secure and transparent payment process.

2.2 Token Issuing Problem

Many different decentralized exchanges (DEXs) have been proposed using different market marker mechanisms, ranging from classic order book mechanism [22] to other more complicated approaches with particular bonding curve [23]. Directly applying the classic order book mechanism on service pricing can bring the low liquidity problem [24]. Specifically, token transactions need to match the buyer and seller, leading to liquidity loss and hindering the transactions.

To mitigate the above issues, early automated market maker-based DEXs (AMM-based DEXs) such as Bancor [23] used bonding curve model for pricing assets: in this models, the function specifies the cost of an asset based on the total available supply. Another possible model for pricing assets named constant product market marker (CPMM), first introduced by Uniswap [25], [26], does not require the ability to change the supply of an asset in order to measure its price. Instead, Uniswap holds assets whose relative price we wish to measure in its reserves. Uniswap specifies a pricing function that maps the assets' quantities in reserves to their marginal price. Although the CPMM-based AMMs are similar in spirit to bonding curve-based AMMs, we will distinguish them as a separate class of AMMs because of relatively distinct range of applicability.

2.3 PT-Based Players' Behavior Analysis

The research of using behavioral economics (and PT in particular) to understand user decisions in networking is at its infancy stage. Li et al. [27] considered a linear value function with the probability distortion and compared the equilibrium strategies of a two-user random access game under EUT and PT. Xiao et al. [28], and Wang et al. [29] considered a linear value function with the probability distortion and characterized the unique Nash Equilibrium of an energy exchange game among microgrids under PT. Yu et al. [30] considered the general S-shaped value function in studying a secondary wireless operator's spectrum investment problem.

3 Design of CryptoArcade

In this section, we first introduce the system overview of CryptoArcade. Then, we illustrate the *token issue protocol* and *CloudArcade*, respectively.

3.1 System Overview

In this subsection, we present an overview of our proposed system, composed of the game store, cloud gaming service, and blockchain platform. In our system, games are run in virtual machines (VMs) in the cloud and configured by a cloud gaming service, which uses a uniform token for access and game time continuation. Tokens in the system are used for unlocking the game by unlocking the control panel, which can be purchased by calling the token issue protocol. When the asset in the game is run out, the control panel will be locked,

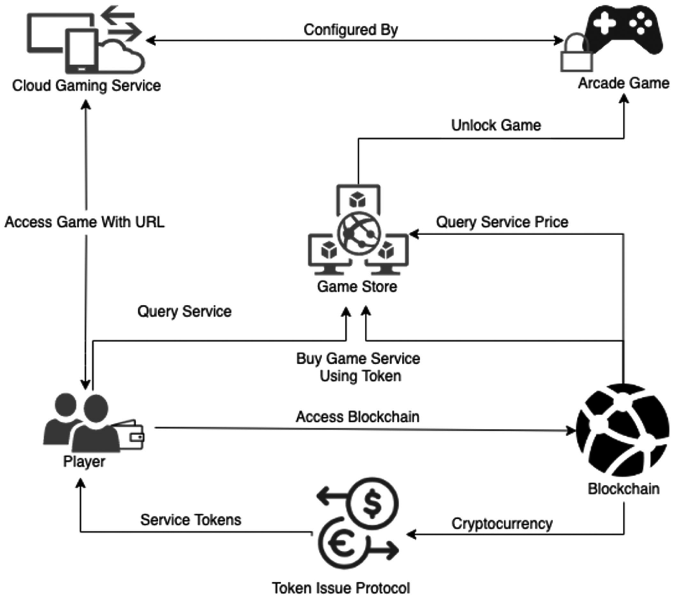


Fig. 1. System overview of CryptoArcade.

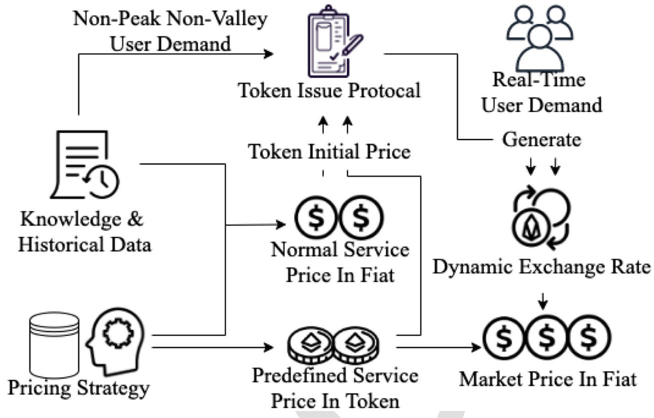


Fig. 2. Token issue protocol.

or the game cannot continue. To access or continue gaming, tokens should be paid. Our design is depicted in Fig. 1.

Unlike traditional cloud gaming services, here, we only choose to serve coin-op arcade games, which require coins to start or continue the gaming sessions in the cloud gaming. The arcade game is time irrelevant but only related to the avatar life quantity or the time limits in the game design. A new payment request is needed if the lifetime or any other finite representative regarding the payment is exhausted. This particular character eliminates all the pricing concerns relevant to the playing period. In CryptoArcade, we set a game service price as a certain number of tokens written in a smart contract. This first provides a transparent payment process that helps players to have a clear understanding of the cost they need to pay. Second, though the number of tokens is relatively constant, the price of tokens can always reflect the actual market demand. The game service price is automatically adjusted according to the market condition, and resource optimization based on dynamic pricing manipulation can be achieved in that sense.

3.2 Token Issue Protocol

Integrating the dynamic pricing ability of cryptocurrency into the real market, liquidating the token, maintaining the token price in a reasonable range, and keeping the price elasticity to reflect the supply and demand of the market are crucial to the success of the CryptoArcade. The AMM-based DEXs seem to be proper for our case. However, the CPMM-based DEXs, such as Uniswap and Sushiswap, define a relationship between two or more tokens [31]. The price of tokens changes on a fixed “bonding curve”, depending on the ratio of tokens in the pool, which can be dramatically affected by arbitrageurs’ behaviors.[5] To keep the token price be relatively stable, we introduce the Bancor protocol. Besides providing autonomous liquidity for tokens on the smart contracts [23], Bancor defines a relationship between token price and supply. Specifically, the token’s price is continuously recalculated according to not only the balance of the connector’s[6] value but also the total supply of the tokens. The SP can tune the three critical parameters, namely, the connector weight, initial connector balance, and initial token supply, to determine the initial token price and price-supply relation of the token issuing method [33]. This method has been used widely to maintain relatively stable cryptocurrency prices in several protocols, such as Aavegotchi[7] and FEI.[8]

As shown in Fig. 2, the SP uses the historical data and knowledge of the service, with its designed pricing strategy to determine the normal *price in fiat*. Normal *price in fiat* is an estimated service price presented by fiat currency using the non-peak non-valley demand data. Then, SP determines the service *price in token*, claiming the relationship between the token number and service access (e.g., one token a service). Based on the normal *price in fiat*, *price in token* and knowledge of the service demand, SP can tune the parameters for the token issue contract. After the token issue protocol is determined, players who interact with the protocol can generate a real-time dynamic exchange rate between the issued token and fiat currency. According to this dynamic exchange rate, the *price in token* can be appropriately mapped to a market price in fiat, achieving the dynamic pricing for service.

3.3 CloudArcade

After trading tokens via calling token issue protocol, players can buy cloud gaming services in CloudArcade. We illustrate our design of CloudArcade in Fig. 3. Video games are executed in the VMs hosted by cloud gaming services and have their corresponding game service URLs. These services are registered in a local database of the cloud server. From the perspective of players, players need to first log in to their cryptocurrency wallets to access the game store. Then they can query game prices through the interaction with the smart contract deployed by the CloudArcade. The game store

5. The price of the cryptocurrency are volatile. For example, the price of ETH at the beginning of 2017 was roughly $10; a year later, it was over $1,400 [32]. And now, it has been over $4,000 in 2021.

6. The connectors are other frequently-used cryptocurrencies, such as USDT, USDC, DAI, and ETH

7. https://aavegotchi.com/

8. https://docs.fei.money/protocol/bondingcurve

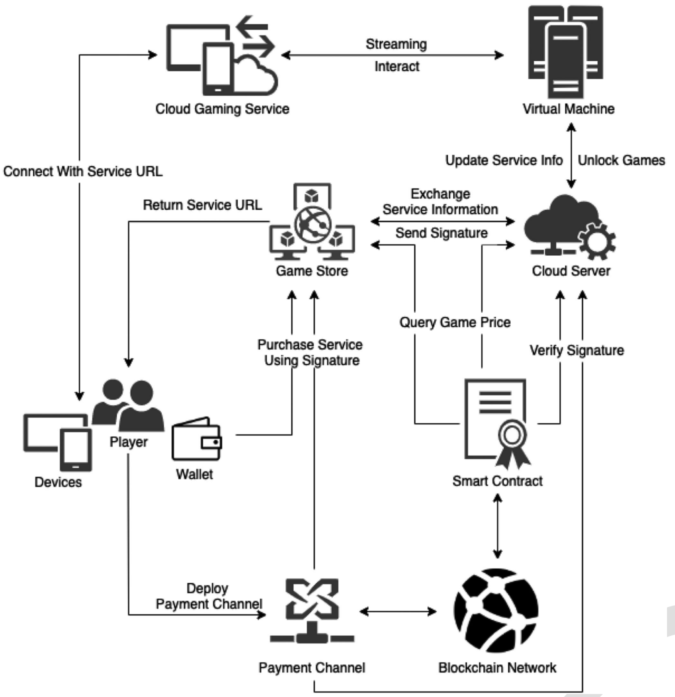


Fig. 3. CryptoArcade architecture.

automatically retrieves other game services information from the cloud server, as specified in Section 3.3.2.

3.3.1 Game Service Setup

Game services provided in the CloudArcade should maintain games that have outside control over their primary processes. A game process should be blocked or terminated when no activation signal is received. Meanwhile, it should also be a non-roguelike genre that players can pick up anytime to continue playing. To this end, games run in VMs should be modified to fulfill the following requirements: 1) a game's process can only be run or continued when activated; 2) the game can be only activated by the central server; 3) the game's process should be blocked again when current service is over.

3.3.2 Game Service Information Fetching

We use a centralized cloud server to store the basic information of the game services and a smart contract deployed on the Ethereum platform to store the price of the games. The game store will automatically query the game service information from the cloud server, like the name and degree of crowdedness. It will also open up a bi-directional communication channel to receive the server's latest price and queue information. To completely use the game store, players first need to inject their wallet accounts. Players may use self-hosted wallet plugins in their browser and authorize wallet accounts in the store. The store will automatically trigger the account address, balance, and other essential information of the wallet. If the wallet did not login yet, a warning message would be generated. After the wallet information is successfully detected, players can fetch the game services information by clicking the query button on a particular game card. The game store will then query the price through the interaction with the smart contract we previously mentioned.

3.3.3 Service Purchase

After receiving the information from the smart contract and cloud server, the player can make a transaction when available services exist for a specific game. However, the average playing time for an arcade game is relatively low, and the purchase requests will be very frequent. Besides, the throughput of blockchain pales in comparison to centralized payment systems such as VISA [34]. This pending time will result in too much waiting for a service like that. Moreover, because there exists an emission for every transaction, there will finally be many total events for all the payment processes, which adds difficulty for the cloud server to search the latest block that matches up with the need. The searching process will become slower as the event blocks grow up. Also, there is a problem with frequent interactions of smart contracts in this system, resulting in some delays.

To solve the problems mentioned above, we integrate the payment channel[9] – a second layer protocol into CloudArcade, as illustrated as Fig. 3. The payment channel is widely studied and utilized by researchers in solving such problem [18], [35], [36]. Players now use the payment channel instead of directly committing transactions to the blockchain to perform purchase actions. A payment channel is a pre-payment offline transaction model designed to allow players to make multiple transactions without committing these transactions to the blockchain. Here, instead of directly calling the smart contract deployed by CloudArcade to make a transaction, the player first deploys a smart contract by himself, which is then called the payment channel. The player needs to attach enough tokens to the contract to make further transactions. Every time a payment channel is created, the game store will send its address and the player's account address to the central cloud server. The server will then update the record in the local database if there already exists a payment channel in the local database for the corresponding player's wallet account. The old address will be replaced with the new one, and CloudArcade will record the address for further claims and fund release.

When a player makes a transaction, he needs to authorize a payment by signing the message with the newest cumulative payment and the payment channel address, then sending it to the cloud server. After receiving the signature, the cloud server will deconstruct the signed message to check whether the signature is valid. The following checks are performed: *1)Address verification:* That means the contract address and player address inside the signature will be validated. The player's address will be confirmed to see whether it is matched up with the player that sends this signature to the cloud server. The contract address will be validated to avoid a replay attack. The request will be rejected if there exists any wrong in the previous check process. *2) New Total Amount Verification:* The cloud server can get the newly paid fees by comparing the new accumulated price inside the signature, and the record new verified

9. https://solidity.readthedocs.io/zh/stable/solidity-by-example.html\#micropayment-channel

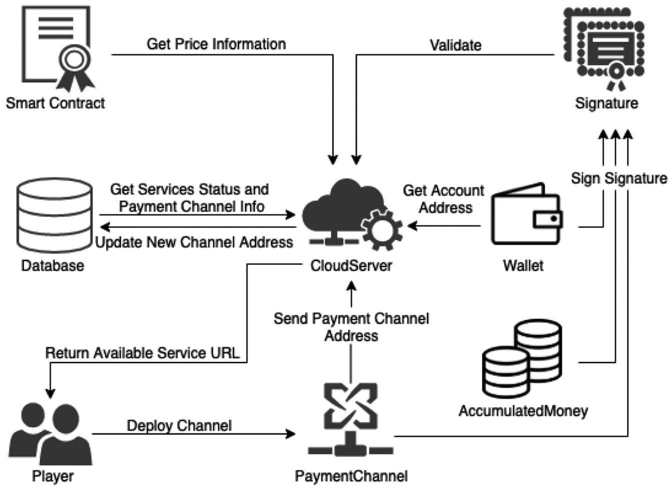


Fig. 4. Validation in CloudArcade.

accumulated price to the local database. If the newly added price is not matched with the current game price demonstrated in the smart contract, the request will be rejected. *3) Total Amount Exceeding Verification:* As the payment channel always has a ceiling for the pre-paid ether amount, the cloud server needs to check whether the new total amount already exceeds the maximum. If so, the request will be rejected. The check process can be graphed as Fig. 4. If all checks are correct, the service allocation process will be conducted. And the latest signature and the total amount of the given wallet address will be updated. When CryptoArcade decides to withdraw money, it only needs to present a signed message to the smart contract. After the authenticity of the message is verified, the fund will be released. Because the payment is offline and does not operate on the blockchain network, it eliminates the pending problems in CryptoArcade.

3.3.4 Service Allocation

After receiving the *txhash* in the previous step, the player can now use it to exchange the corresponding game service from the cloud server. The player sends his *txhash* together with the account address to the central cloud server, and the cloud server will fetch all the GamePayoutSuccess events from the smart contract. The central server can find the latest event performed by the account address and check whether the *txhash* is valid. The local database will also be used to make verification. The following checks are performed in the verification process: 1) Whether the *txhash* has been used. That is whether or not the *txhash* has already been recorded in the local database for this account address. If the *txhash* is already used, the allocation requests will be rejected. 2) Whether the *txhash* is the latest. That is if the *txhash* matches the latest GamePayoutSuccess event that cloud servers retrieved from the smart contract or not. If not, the allocation requests will be rejected. If all checks pass through, the cloud server will derive the game ID from the data part of the GamePayoutSuccess event and check whether available resources exist to provide the game service for the particular game identified by the game ID. If not, the activation process will still be rejected. If there are enough resources on a cloud server, the server will unlock the corresponding game in the local VM by rewriting the lock file and sending the service URL back to the player. The latest *txhash* for this account address will be updated. Then the updated service information will be broadcast to all players.

3.3.5 Game Service Access

After receiving the valid service URL, the player can access the game service. All games are run in the VMs with a lock file inside. The game is only runnable when the lock file is false, and only the cloud server can unlock these files. These files will be reset true after a game service ends like life end or time is up. The cloud gaming service hosts all games, and they will have their corresponding configuration file that determines their streaming and control properties and the service URLs. And all service URLs and lock files will be registered in the local database of the cloud server. The cloud gaming service will start streaming these game content when a configuration is run, and the service URL can be used to access these games. In this sense, the player can only access the service whose inside game has already been set unlocked. In practice, the service URL will be generated randomly to ensure the game experience's safety, which can be easily done by changing the configuration file of specific game service.

4 PT and EUT-Based Player's Model

Although CryptoArcade can solve the problems of price fluctuation and opaque pricing in traditional cloud game pricing, due to the payment channel[10] and transaction fee during the token purchase process, players need to estimate the tokens needed for the cloud gaming before purchasing tokens. To better understand the players' decisions when buying or selling tokens, we use EUT and PT to model player behavior, respectively.

As illustrated in Fig. 1, we consider a cloud gaming service market consisting of a SP and an extensive set $\mathcal{J}$ of players. Each player is associated with a wallet and can obtain tokens by calling *token issue protocol* before cloud gaming. The token price π is volatile and depends on the demand for tokens at the current moment. Hence, a player can buy or sell tokens based on the current token price and future token consummations. We consider the operation for an extended period divided into T session slots. For notational convenience, we normalize the length of each session slot to be one. Moreover, we assume that total N players are at tth session slot. Since the number of players in the cloud gaming service market is large, a single player's choice will have a negligible impact on the market.[11]

4.1 Player's Modeling

In this subsection, we define the player's specific costs incurred to participate in CryptoArcade.

10. Before the cloud gaming service, players need to add enough tokens to their pre-deployed payment channel smart contract address to ensure that the game does not end for lack of tokens.

11. The impact here refers to the effects of player decisions on the current token price.

- Token fee: Token fee refers to the cost for buying tokens at a price π_t in the tth session slot. We consider a static game and denote the strategy of player j by σ_j, with $\sigma_j \in [-R_j, +\infty)$, which is the number of tokens that player j buys or sells, where R_j is the number of remaining tokens in player j's wallet. Specifically, positive values of σ_j indicate that player j purchases tokens via calling *token issue protocol*, and negative values of σ_j represent that the player j sells tokens back to the *token issue protocol*. $\sigma_j \geq -R_j$ implies that the player can not sell more tokens than the remaining tokens in the wallet.
- Transaction fee: Transaction fee, also known as the gas fee, refers to the transaction cost for token transfers or the execution of smart contract code in a blockchain. Gas fees are paid in the native currency of Ethereum, ether (ETH), which is calculated as the gas used multiplied by the gas price. Specifically, gas G refers to the unit that measures the amount of computational effort required to execute specific operations on the Ethereum network.[12] For instance, token transfers always take up 21,000, and a transaction involving smart contracts would take up a greater amount of gas, with the exact value determined by the complexity of the transaction. Gas price μ is charged by miners to use the computational power of Ethereum, which is generally quoted in "gwei". The conversion factor between ether and "gwei" is represented as $g = 10^{-9}$. Hence, the total transaction fee can be calculated as $G\mu g P_{eth}$, where P_{eth} represents the price of ETH at the current time slot. For simplicity, we use f to denote the transaction fee. Hence, the transaction fee for participation in cloud gaming services can be summarized as follows:

$$c(\sigma_j) = \begin{cases} f, & \text{if } \sigma_j \neq 0, \\ 0, & \text{if } \sigma_j = 0. \end{cases} \tag{1}$$

  If player j decides to buy or sell tokens (i.e., $\sigma_j \neq 0$), the transaction fee is f, if player j decides to use the remaining tokens without buying and selling operations (i.e., $\sigma_j = 0$), the transaction fee is 0.
- Satisfaction Loss: Another critical factor that the player j needs to consider is the future token consumption uncertainty in the current session slot. If his token consumption exceeds his total tokens, she/he will incur a satisfaction loss. For simplicity, we consider a linear satisfaction loss function in the specific interval,

$$L(\sigma_j) = \begin{cases} 0, & \text{if } R_j + \sigma_j - d_j \geq 0, \\ k_j(R_j + \sigma_j - d_j), & \text{if } R_j + \sigma_j - d_j < 0, \end{cases} \tag{2}$$

  where k_j is the satisfaction coefficient to represent satisfaction-seeking level of player j and d_j represents the token consumption of player j in this time slot. Moreover, R_j, k_j and d_j are constants for player j.

Next, we derive the player's expected utilities under both EUT and PT.

12. https://ethereum.org/en/developers/docs/gas/

4.2 Utilities Under EUT

We focus on a single player's decision-making problem and ignore the player index j for notional convenience. Hence we will write the future token demand of player j as d, the strategy as σ, the remaining tokens, and the satisfaction coefficient as R and k, respectively. We assume that the player's token consumption in the current session slot has I possible values $d_i : i = 1, 2, \cdots, I$, with the corresponding probabilities $p_i : i = 1, 2, \cdots, I$ such that $\sum_{i=1}^{I} p_i = 1$. Hence, if a player buys σ tokens, his utility under EUT will be represented as follows:

$$U_{EUT}(\sigma) = \sum_{i=1}^{I} p_i[-\pi\sigma + L(\sigma) - c(\sigma)]. \tag{3}$$

4.3 Utilities Under PT

In this subsection, we formulate the PT player's utility considering the three parts of PT, namely S-shaped value function $v(x)$, probability distortion function $w(p)$, and reference point u_{ref} [37]. Moreover, we discuss the impact of the three features on a PT player's utility.

Reference point u_{ref} [37] refers to players' personal benchmark to evaluate their final utilities, which varies from person to person. Specifically, the player will consider obtaining a gain if the actual outcome is higher than the reference point. Otherwise, she/he will think that she/he suffers from a loss. Hence, players with high reference points always have high expectations of the outcome. Players with low reference points always have low expectations. The reference point will significantly affect the player's subjective valuation of the outcome and strategies.

Subjective valuation $v(u)$ can be calculated by the actual outcome u. Behavioral studies show that an S-shape asymmetrical valuation function can better capture practical human psychological loss and risk preference. Specifically, the function $v(u)$ is concave in the gain region (i.e., $u > 0$) and convex in the loss region (i.e., $u < 0$). Moreover, the impact of the gain is smaller than the loss, i.e., $|v(-u)| \geq v(u), \forall u > 0$. For a special case in Fig. 5 (i.e., the blue line), $v(u)$ increases with increasing u.

$$v(u) = \begin{cases} (u - u_{ref})^{\beta}, & u \geq u_{ref}, \\ -\lambda(u_{ref} - u)^{\beta}, & u < u_{ref}, \end{cases} \tag{4}$$

where $0 < \beta \leq 1$ and $\lambda \geq 1$. We use β to represent the risk aversion parameter. A smaller β indicates that the value function is more concave in the gain region (i.e., $u > 0$) and convex in the loss region (i.e., $u < 0$), which represents that the player is more risk-averse in gains and risk-seeking in losses. Besides, the loss penalty parameter λ also significantly affects the PT player's utility. A larger λ indicates that the player is more loss averse.

Probability distortion function $w(p)$ represents humans' psychological over-weighting of low probability events, and under-weighting of high probability events [38], as shown in Fig. 6. A commonly used probability distortion function is

$$w(p) = \exp(-(-\ln p)^{\alpha}), 0 < \alpha \leq 1, \tag{5}$$

where α is the probability distortion parameter, which depicts how a player's subjective evaluation distorts the

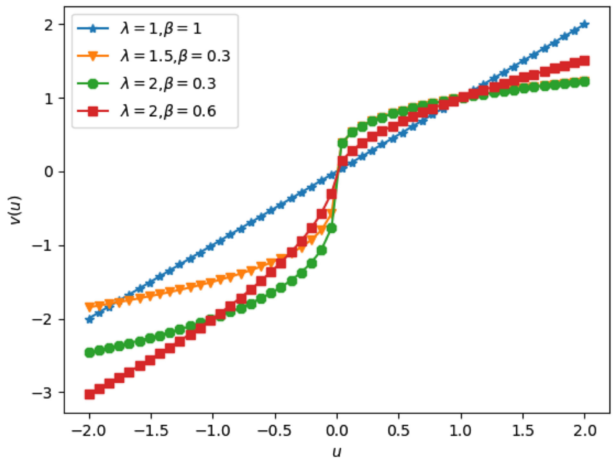


Fig. 5. The S-shaped asymmetrical value function in PT.

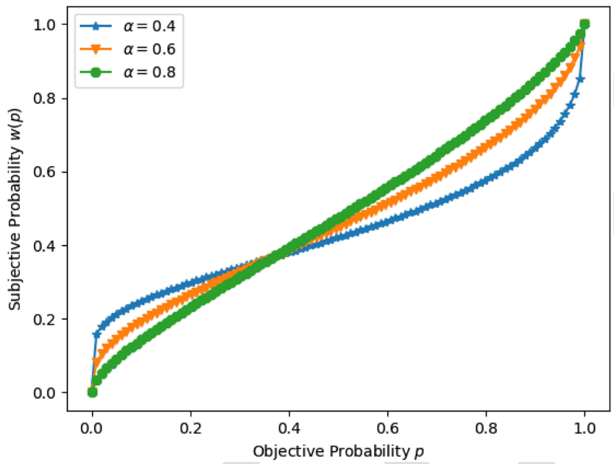


Fig. 6. The probability distortion function $w(p)$ in PT.

real probability. A larger α means a smaller probability distortion [38]. p refers to the real probability, and $w(p)$ represents the corresponding subjective probability under PT.

Considering the above three features in PT, a player's expected utility under PT is

$$u_{PT} = \sum_{i=1}^{I} w(p_i) v(-\pi\sigma + L(\sigma) - c(\sigma)]. \tag{6}$$

Combined Eqs. (3) with (6), we can obtain that the players' utility function under EUT is a special case of utility function under PT, with the parameter choices of $\lambda = \beta = \alpha = 1$ and $u_{ref} = 0$.

5 Solving the EUT and PT-Based Optimization Problem

To simplify the presentation and better illustrate the insights, we assume $I = 2$ for the rest of the paper. More specifically, we consider two possible future token consumption in the current time slot: d_h and d_l, with $d_h > d_l > R > 0$. We use p to represent the probability of low token consumption and $1-p$ to represent the probability of high token consumption d_h.

5.1 Performance of the EUT-Based Cloud Gaming Service Game

To solve the EUT-based optimization problem mentioned in Section 4.2, we first consider the player's utility maximization problem and formulate Eq. (3) as follows:

$$\begin{aligned} &\max \sum_{i=1}^{I} p_i[-\pi\sigma + L(\sigma) - c(\sigma)], \\ &s.t. \quad \sigma \in [-R, +\infty). \end{aligned}$$

Theorem 1. *The optimization problem under EUT is a piecewise function, which is not convex, so we discuss different cases and calculate the corresponding optimal strategy. The player's optimal strategy under EUT is summarized as follows:*

- *If* $0 < \pi \leq (1-p)k$ *and* $f \leq (k-\pi)(d_h - R) - kp(d_h - d_l)$, *the player's optimal trading strategy is* $\sigma^* = d_h - R$.
- *If* $(1-p)k < \pi \leq k$ *and* $f \leq (k-\pi)(d_l - R)$, *the player's optimal trading strategy is* $\sigma^* = d_l - R$.
- *If* $\pi > k$ *and* $f \leq (\pi - k)R$, *the player's optimal trading strategy is* $\sigma^* = -R$.
- *For any other conditions, the player's optimal trading strategy is* $\sigma^* = 0$.

The proof of Theorem 1 is given in Appendix A, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TCC.2022.3210013.

5.2 Performance of the PT-Based Cloud Gaming Service Game

To solve the PT-based optimization problem mentioned in Section 4.3, we first consider the player's utility maximization problem and formulate Eq. (6) as follows:

$$\begin{aligned} &\max \sum_{i=1}^{I} w(p_i) v(-\pi\sigma + L(\sigma) - c(\sigma) - u_{ref}] \\ &s.t. \quad \sigma \in [-R, +\infty) \end{aligned}$$

Here we set the reference point $u_{ref} = \pi R$, which means the player's high expectation utility is her/his existing asset. So the player's utility is:

$$U(\sigma) = \sum_{i=1}^{I} w(p_i)(\pi R + \pi\sigma - L(\sigma) + c(\sigma))^{\beta}. \tag{7}$$

We compute its first-order derivative as follows:

$$\begin{aligned} \frac{\partial U}{\partial \sigma} =& \beta w(p_k)\left(\pi - \frac{\partial L(\sigma)}{\partial \sigma} + \frac{\partial c(\sigma)}{\partial \sigma}\right) \\ & (\pi R + \pi\sigma - L(\sigma) + c(\sigma))^{\beta - 1}. \end{aligned} \tag{8}$$

One of the key challenges of computing the root of $\frac{\partial U}{\partial \sigma} = 0$ is due to the $(\pi - \frac{\partial L(\sigma)}{\partial \sigma} + \frac{\partial c(\sigma)}{\partial \sigma})$ and the $(\pi R + \pi\sigma - L(\sigma) + c(\sigma))^{\beta-1}$. To avoid a fractional order, we set the risk aversion parameter $\beta = 1$ to compare the player's utility under PT with the player's utility under EUT.

Theorem 2. *The optimization problem under PT is a piecewise function, which is not convex, so we discuss different cases and* *calculate the corresponding optimal strategy. The player's optimal strategy under PT is summarized as follows:*

- *If* $0 < \pi \leq \frac{w(1-p)k}{w(p)+w(1-p)}$ *and* $f \leq (\pi R - \pi d_h - kR) + \frac{kd_l w(p)+kd_h w(1-p)}{w(1-p)+w(p)}$, *the player's optimal trading strategy is* $\sigma^* = d_h - R$.
- *If* $\frac{w(1-p)k}{w(p)+w(1-p)} < \pi \leq k$ *and* $f \leq (k-\pi)(d_l - R)$, *the player's optimal trading strategy is* $\sigma^* = d_l - R$.
- *If* $\pi > k$ *and* $f \leq (\pi - k)R$, *the player's optimal trading strategy is* $\sigma^* = -R$.
- *For any other conditions, the player's optimal trading strategy is* $\sigma^* = 0$.

The proof of Theorem 2 is given in Appendix B, available in the online supplemental material.

From the above two theorems, we can observe that a player's optimal token buying quantity is discontinuous. This is due to the linearity of the utility function in the EUT case and the convexity of the utility function in the PT case with u_{ref}. Details are given in Appendix A and B, available in the online supplemental material. Besides, we have the following facts.

Fact 1. *When the gas fee* f *is small enough, the players under PT and EUT have the same threshold gas fee*[13] *of the token price* π.

Proof. From the *Theorems 1* and *2*, we know when the gas fee f is small, all of the EUT players under the range of token price $(0, (1-p)k]$ will buy token $(d_h - R)$ and the range of token price $((1-p)k, k]$ will buy token $(d_l - R)$. So when the range of token price is $(0, (1-p)k] \cup ((1-p)k, k] = (0, k]$, the EUT players will choose the strategy to buy tokens.

Similarly, all of the PT players under the range of token price $(0, \frac{w(1-p)k}{w(p)+w(1-p)}]$ will buy token $(d_h - R)$ and the range of token price $(\frac{w(1-p)k}{w(p)+w(1-p)}, k]$ will buy token $(d_l - R)$. So, the buying token strategy for the range of token price is $(0, \frac{w(1-p)k}{w(p)+w(1-p)}] \cup (\frac{w(1-p)k}{w(p)+w(1-p)}, k] = (0, k]$. Therefore, the players under PT and EUT have the same threshold gas fee of the token price π. □

Fact 2. *When the risk aversion parameter* $\beta = 1$, *for both PT and EUT players, they are more likely to reach high token demand* d_h *with decreasing* p.

Proof. Proof.

When the $\beta = 1$, we can easily find in the EUT condition, the token price and gas fee thresholds of the trading strategy $d_h - R$ are $(1-p)k$ and $(k-\pi)(d_h - R) - kp(d_h - d_l)$, which are increasing with the decreasing p. Similarly, in the EUT condition, the token price and gas fee thresholds of the trading strategy $d_h - R$ are $\frac{w(1-p)k}{w(p)+w(1-p)}$ and $(\pi R - \pi d_h - kR) + \frac{kd_l w(p)+kd_h w(1-p)}{w(1-p)+w(p)}$, which also are increasing with the decreasing p. Therefore, both PT and EUT players are more likely to reach high token demand d_h with the decreasing p. □

13. The threshold gas fee determines whether a player operates or not.

Fact 3. *When the risk aversion parameter* $\beta = 1$ *and probability distortion parameter* $\alpha = 1$, *a PT player with reference point* $u_{ref} = \pi R$ *has the same threshold conditions with an EUT player.*

Fact 4. *When the token price* π *is high enough, the player under PT and EUT has the same threshold gas fee* $\hat{f}$. *When the token price* π *is large, which is higher than satisfaction coefficient* k, *both EUT and PT players have one condition. If the gas fee is smaller than* $(\pi - k)R$, *they will sell all tokens. Otherwise, they will choose no operations.*

6 System Implementation and Evaluation

In this section, we present the implementation of a prototype to demonstrate our proposed CryptoArcade system.

6.1 Enabling Technologies

We select a series of packages to fulfill the prototype development requirements. For the blockchain platform, we employ Ethereum[14] due to its popularity in the decentralized application community. To this end, solidity[15] becomes our smart contract programming language. For the client, we adopt vue-cli[16] and webpack[17] framework to support the fast development of the front-end. And we make a wallet injection in the game store with the support of the Metamask,[18] a web browser plug-in to run Ethereum DApps without running a full Ethereum node. The smart contract is invoked by web3.js,[19] which is a JavaScript interface for contract interaction.

6.2 System Deployment

We deploy our smart contract on Rinkeby Testnet,[20] an Ethereum testnet that developers use to test and perfect their decentralized applications to conduct empirical experiments. This is because that when we deployed CryptoArcade, ETH[21] adopted PoW, which is one of the most decentralized and secure blockchains. Although the blockchain based on PoS and delegate Proof-of-Stake (DPoS) consensus models has lower costs, its security and centralization are controversial. The smart contract is deployed on Etherscan.[22] We designed two different smart contracts using solidity. The first smart contract provides the interface for the price query of the game services. It also provides the ability to directly use Ethereum as the payment method for the cloud gaming services. The second smart contract is the payment channel contract provided by the solidity. Its bytecode and API will be stored in the front-end, and players can use them to deploy the payment channel with the help of the Metamask. After successful deployment, the player can sign the transaction using the address of the smart contract.

14. https://www.ethereum.org/
15. https://github.com/ethereum/solidity
16. https://cli.vuejs.org/
17. https://webpack.js.org/
18. https://metamask.io/
19. https://web3js.readthedocs.io/en/v1.2.0/
20. https://www.rinkeby.io/
21. Indeed, ETH executed the *merge* on September 15, 2022, which completed the transition of Ethereum to Proof-of-Stake (PoS), officially deprecating Proof-of-Work (PoW). Still, the merger is controversial, and PoW has considerable followers in the Ethereum community.
22. https://rinkeby.etherscan.io/

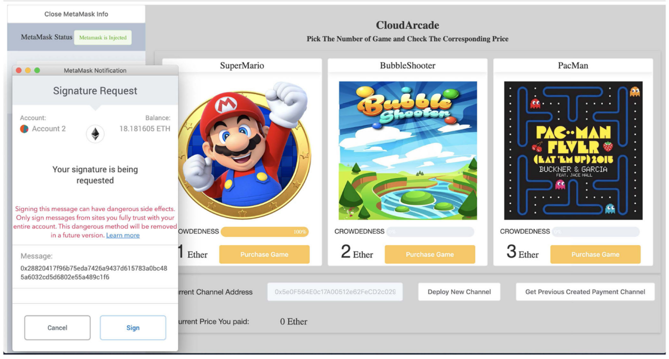


Fig. 7. Payment in CryptoArcade.

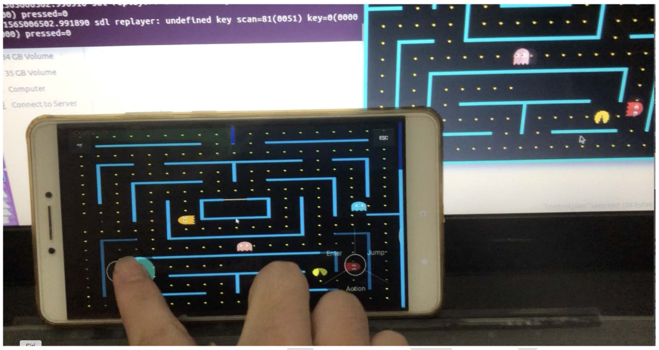


Fig. 8. Demonstration of game play with CryptoArcade.

To deploy the CryptoArcade system, we set up the open-source GamingAnywhere [39] platform as SP. Three open-source games, including Mario,[23] Bubble shooter,[24] and Pac-man,[25] are retrieved from the GitHub repositories to be executed in the CryptoArcade. We design a simple lock and unlock procedure for our game services. We define the game service status: true for unlocked and false for locked and store the status inside the JSON file lockfile.json under the root of the game directory. The service ID will also be stored in it. All lock files' file paths and their corresponding service identifiers will be stored in the database. The game will continually scan the lock file, only when the status is true, the game process can be run normally. So we initialize all the game status as false. When a successful transaction is confirmed, the server will unlock the allocated game service and return the service URL as the response. Players can use it to access the game service. The server also needs to mark the given service as occupied in the database. For example, when a service is over, the lifetime comes to zero. The game process will rewrite the status to false and send the modification request to mark the status of the service in the database as available.

6.3 Demonstration

For CryptoArcade, the service price will be automatically shown on the game card. Players can click the button at the bottom of the page to deploy a new payment channel or click the other button to get the payment channel address they created in the past. After a channel is selected, a player can now click the button on the game card to send transaction requests. Paid request validation triggers a signature warning from Metamask, as shown in Fig. 7. Upon confirmation, the signature will be sent to the server. If the signature is verified, the game store will notify the service URL. The cumulative cost within the payment channel will appear at the bottom of the page.

After getting the service URL from the server, the game process can be visited and controlled via the support of the GamingAnywhere clients, as demonstrated in Fig. 8. Besides, we evaluate our system performance, and complexity in our previous work [1].

23 https://github.com/justinmeister/Mario-Level-1
24. https://github.com/justinmeister/bubbleshooter
25. https://github.com/CharlesPikachu/Games/tree/master/Game14

6.4 Simulation and Results

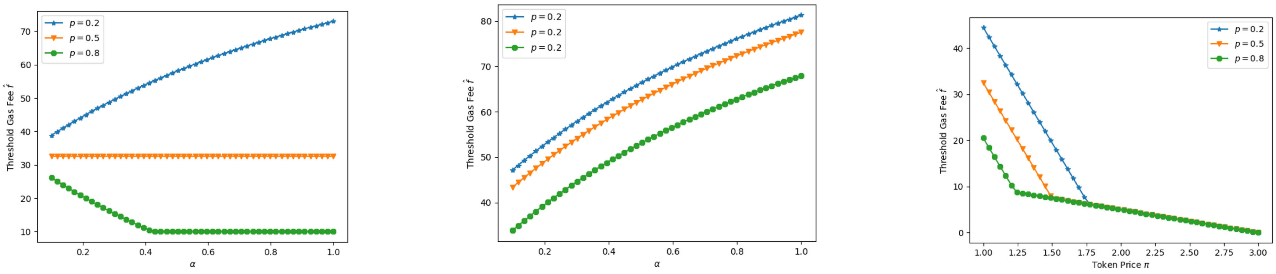
In this section, we provide numerical results to illustrate a player's behavior, and analyze the impact of PT model on players' optimal decisions.

6.4.1 Effect of Parameters on Player's Threshold Gas Fee

We first illustrate the impact of PT model parameters, market parameters, and demand uncertainty parameters on the players' optimal decision. We use Python as the tool to evaluate the player's behaviors in the CryptoArcade system. We mainly focus on the price and revenue change with different parameters input. From the previous part, we know the reference point $U_{ref} = \pi R$, which means the value of tokens in players' payment smart contract before cloud gaming. We assume $\beta = 1$ and $\lambda = 2$ [40]. Besides, we set $R = 10$, high demand $d_h = 60$, low demand $d_l = 15$, token price $\pi = \$1$, and $k = 3$.

Impact of the probability distortion parameter α on a player's threshold gas fee. Fig. 9a considers three different probabilities of low demand: low ($p = 0.2$), medium ($p = 0.5$), and high ($p = 0.8$). The token price varies from 1.00 to 3.00 with an increment of 0.25. We can observe that with the probability distortion parameter α increases, the threshold gas fee $\hat{f}$ decreases and then keep stable when $p = 0.8$. This is because a smaller α means a player will over underestimate the probability of low demand, and it is more risk-seeking. Hence, it will choose to meet its high demand when α is small. Since the probability of low demand under PT rises with the increasing α, the player will choose to meet the low demand when α is around 0.3, and the threshold gas fee will keep stable eventually. Besides, we can find that $\hat{f}$ is independent of α when $p = 0.5$. The reason is that low demand and high demand probability are the same under PT. Moreover, the threshold gas fee $\hat{f}$ increases in α when $p = 0.2$. The player chooses to meet the high demand because of the high probability. Since a smaller α means that a player will overestimate the low probability more, it becomes more risk-averse when p is small. Under this condition, the probability of low demand will decrease with the increasing α under PT.

Impact of the Remaining Tokens R in the Wallet on a Player's Threshold Gas Fee. Fig. 9b illustrates how the player's threshold gas fee $\hat{f}$ changes with the different remaining token R and the probability distortion parameter α. We assume that $p = 0.2$. The remaining tokens R are 3, 5, and 10, respectively, and the

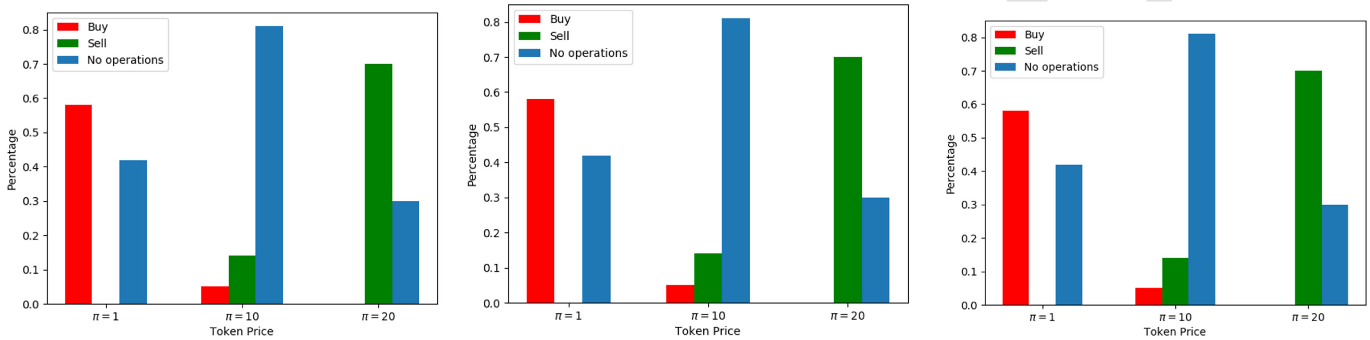


(a) player's threshold gas fee $\hat{f}$ vs. probability distortion parameter α with different p

(b) player's threshold gas fee $\hat{f}$ vs. probability distortion parameter α with different R

(c) player's threshold gas fee $\hat{f}$ vs. token price π with different p

Fig. 9. The effect of α, R and π on a player's threshold gas fee $\hat{f}$.



(a) $Var(\beta) = 0.2$

(b) $Var(\beta) = 0.5$

(c) $Var(\beta) = 0.8$

Fig. 10. The effect of β and token price π on PT players' behaviors.

token price varies from \$1 to \$3 with an increment of 0.25. From Fig. 9b, we can observe that $\hat{f}$ increases accordingly as α increases in three different values of R, respectively. This is because as α increases, the probability of high demand under PT will be larger, and the player will gain more revenue from satisfying high token consumption. Hence, the threshold gas fee raises with increasing α. Furthermore, we can find that $\hat{f}$ decreases with increasing R. If a player holds more tokens in his wallet, it will gain more revenue without buying extra tokens to token part in cloud gaming. Hence, when the gas fee is more significant, the player with more remaining tokens prefers to use the remaining tokens rather than pay an extra gas fee to buy tokens.

Impact of the Token Price π on a Player's Threshold Gas Fee. Fig. 9c considers three different probabilities of the low demand and illustrates how the threshold gas fee $\hat{f}$ changes with the token price π. We assume that $\alpha = 0.2$ and $p = 0.2$. The probabilities of low demand are 0.2, 0.5, and 0.8, respectively, and the token price varies from 1.0 to 3.0 tokens with an increment of 0.25. Under these parameter settings, the player chooses to buy extra tokens to meet the high demand under three different probabilities of low demand when the token price π is low. In contrast, it chooses to meet the low demand as π increases. Hence, the three threshold gas fees $\hat{f}$ are the same when π is larger than 1.75. Besides, the threshold gas fee $\hat{f}$ grows along with the increase in the probability of low demand when the token price π is low.

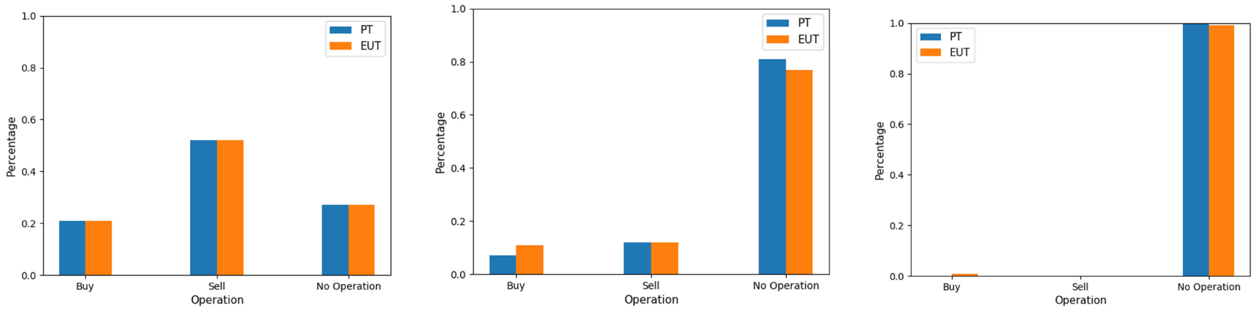
6.4.2 Effect of Parameters on Players' Behaviors

Previous analysis and simulations in Sections 6.4.1 focus on single player's strategies. Here we conduct a numerical simulation to discuss a more realistic scenario where different players may have other behaviors under different external factors[41], [42], [43], [44].

To better illustrate the insight in real life, we adopt the distribution of PT parameters that comes from the literature in psychology and behavioral economics. The literature investigated the PT parameters of each subject in real life experiments [41], [42], [43], [44]. According to the data fitting results in [40], the parameters λ follows a Gamma distribution with a shape parameter $s_\lambda = 3.2433$ and a scale parameter $\theta_\lambda = 0.6018$ ($p = 0.4768$), and that the parameter β follows a Gamma distribution with a shape parameter $s_\beta = 12.8662$ and a scale parameter $\theta_\beta = 0.0583$ ($p = 0.1278$). Besides, we collect the practical gas fee of swapping from 2021-12-11 to 2021-12-18 with a interval 30s from *crypto.com*[26] to decide the range of f (i.e., $\$28 < f < \$325$). Unless otherwise stated, we set the average practical gas fee of calling smart contracts as \$80. Besides, we assume there is a total of 100 players participating in the CryptoArcade at the current epoch. We assume that some parameters of players' utility function to follow the uniform distribution, with $d_l \sim \mathcal{U}(15, 30)$, $d_h \sim \mathcal{U}(30, 60)$, $R \sim \mathcal{U}(5, 15)$, $p \sim \mathcal{U}(0, 1)$. Moreover, we consider different players have different sensitivity to games and assume that the satisfaction coefficient $k \sim \mathcal{N}(5, 8)$.

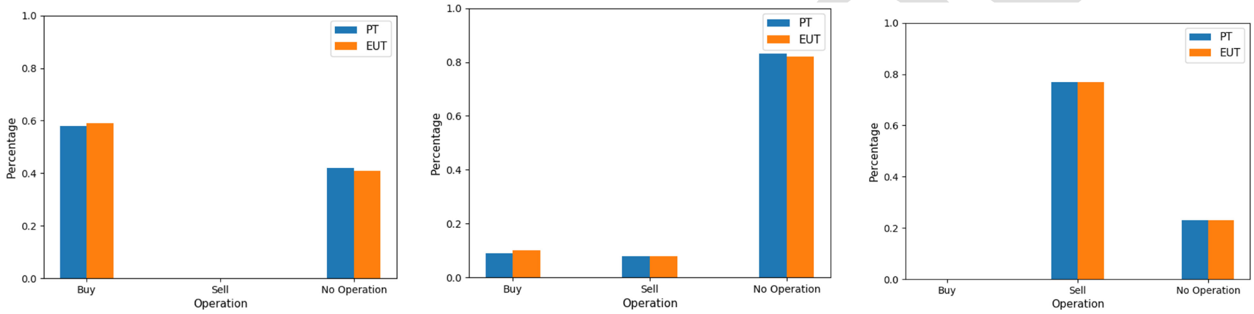
Impact of the Risk Aversion Parameter β on Players' Optimal Strategies. Utilizing the above empirical data, we will study the impact of the heterogeneity of parameter β. We generate this parameter β through the Gamma distribution with a fixed mean ($\theta_\beta = s_\beta \times \theta_\beta = 0.75$)[27]. Figs. 10a, 10b and 10c

26. https://crypto.com/defi/dashboard/gas-fees
27. The mean of the Gamma distributed random variable is the product of the shape parameter s and the scale parameter θ, i.e., $s \times \theta$.



(a) $f = \$28$ (b) $f = \$80$ (c) $f = \$358$

Fig. 11. The comparison of EUT and PT players under different gas fee f.



(a) $\pi = \$1$ (b) $\pi = \$10$ (c) $\pi = \$20$

Fig. 12. The comparison of EUT and PT players under different token price π.

consider three different distribution of risk aversion parameter β, and illustrate the effect of β and token price π on players' behaviors. From the figure, we can obtain that the percentage of players' behaviors is independent of the distribution of β. The fact is that the value function $v(u)$ in Eq. (4) is monotone increasing function with u, hence, the optimal strategies of players only depends on the comparison of u under different strategies, rather than the value of β. Besides, we can notice that increasing token price π makes more players choose to sell tokens, while fewer players buy tokens. In this case, the token price can keep stable. We use the Bancor protocol as token issue protocol, and the token price is related to the number of tokens in circulation. Specifically, When the token price is high, more players choose to sell tokens, leading to an increase in the number of tokens in the token issue protocol and thus, a decrease in the token price. Similarly, When the token price is low, more players choose to buy tokens, leading to a decline in the number of tokens in the secondary market and thus an increase in the token price. Moreover, we can observe that when the token price is small (i.e., $\pi = 1$) and the token price is large (i.e., $\pi = 20$), the token market is more active. In other words, more players buy or sell tokens via the token issue protocol.

Comparison of EUT and PT Players Under Different Gas Fee f. Fig. 11 compares the EUT and PT players' optimal strategies under the gas fee of \$28, \$80, and \$385, which represents the free Internet, standard Internet, and busy Internet. As we can see, if the gas fee is small (free Internet), the players with EUT and PT have the same strategies. When the gas fee is small, the EUT players and PT players only need to consider the token price, and the players under PT and EUT have the same threshold of the token price. When the Internet is standard, compared with the PT players, the EUT players are more likely to buy tokens than no operations, and the number of players to sell tokens is the same. The reason is that when the range of token price is the same, the EUT players have a more extensive range of gas fees f to operate (i.e., including buying and selling). Also, We notice that with the gas fee increases, both EUT and PT players change the buying and selling strategies to the no operation. As a result, when the Internet is busy, almost all players will choose no operation. The reason is that the higher gas fee decreases the utility of operation.

Comparison of EUT and PT Players Under Different Token Price π. Fig. 12 compares the EUT and PT players' optimal strategies under the token price of \$1, \$10 and \$20. Fig. 11 compared with the PT players, we can easily find that under the same lower token price, the EUT players are more likely to buy tokens rather than no operations, and the number of players by selling tokens is the same. This is because, as we discussed earlier, when the token price is small, under the same range of gas fee f, the EUT players have a larger range of token price π to buy the token and the same range of token price π to sell the token. However, when the token price π is higher, like \$20, the EUT and PT players' optimal strategies are the same. When the token price π is large, both EUT and PT players have the same threshold gas fee f for the buying and selling strategies.

7 Conclusion

We present CryptoArcade, a new cloud gaming business model based on the blockchain-empowered token. It provides a new landscape of the commercial cloud gaming business model, which tackles the various problem of the current cloud gaming business model and pricing strategy. The service on CryptoArcade is paid by token, whose price reflects the market demand. By purchasing and using tokens, players pay the floating price in a silent and time irrelevant way, which protects the players' utility and service experience. On the other hand, the floating token price also utilizes cloud computing resources via manipulating consumers' behaviors. By exploiting the smart contract, we also ensure the transparency of the payment process. The transparency payment builds up players' trust in the platform, implicitly increasing the number of players.

Located at the player's premises, we use PT to formulate the player's decision problems under future token consumption uncertainty to understand her/his realistic trading strategies. We have highlighted several key insights. Specifically, we explore external factors such as token price and gas fee on a PT player's strategy. Besides, we provide numerical results showing that the EUT players are more likely to buy tokens than no operations under the same external factors.

8 Future Vision

In our future work, we consider three main aspects. One is the *deepening and improvement of the current model*. Specifically, we consider extending PT and EUT from the special case (i.e., $I = 2$) to more general cases. Also, we consider the inclusion of myopic players in the model comparison. Another one is *token pricing*. We will price the tokens based on the resource consumption in cloud gaming from the SP's perspective.

Acknowledgments

A preliminary version of the article appears as 'CloudArcade: A Blockchain Empowered Cloud Gaming System' in ACM BSCI 2020 [1]. This article has made a significant extension.

References

[1] J. Zhao, Y. Chi, Z. Wang, V. C. Leung, and W. Cai, "CloudArcade: A blockchain empowered cloud gaming system," in *Proc. 2nd ACM Int. Symp. Blockchain Secure Crit. Infrastructure*, 2020, pp. 31–40.

[2] H. E. Dinaki and S. Shirmohammadi, "GPU/QoE-aware server selection using metaheuristic algorithms in multiplayer cloud gaming," in *Proc. IEEE 16th Annu. Workshop Netw. Syst. Support Games*, 2018, pp. 1–6.

[3] Y. Deng, Y. Li, R. Seet, X. Tang, and W. Cai, "The server allocation problem for session-based multiplayer cloud gaming," *IEEE Trans. Multimedia*, vol. 20, no. 5, pp. 1233–1245, May 2018.

[4] I. Slivar, L. Skorin-Kapov, and M. Suznjevic, "QoE-aware resource allocation for multiple cloud gaming users sharing a bottleneck link," in *Proc. IEEE 22nd Conf. Innov. Clouds Internet Netw. Workshops*, 2019, pp. 118–123.

[5] Y. Han, D. Guo, W. Cai, X. Wang, and V. C. M. Leung, "Virtual machine placement optimization in mobile cloud gaming through QoE-oriented resource competition," *IEEE Trans. Cloud Comput.*, vol. 10, no. 3, pp. 2204–2218, Jul.–Sep. 2022.

[6] B. Tandianus, H. S. Seah, T. D. Vu, and A. T. Phan, "Cloud-based dynamic streaming and loading of 3D scene," in *Proc. IEEE Int. Conf. Cyberworlds*, 2018, pp. 409–414.

[7] W. Cai et al., "The future of cloud gaming [point of view]," *Proc. IEEE*, vol. 104, no. 4, pp. 687–691, Apr. 2016.

[8] W. Cai et al., "A survey on cloud gaming: Future of computer games," *IEEE Access*, vol. 4, pp. 7605–7620, 2016.

[9] Y. Faqir-Rhazoui, M.-J. Ariza-Garzón, J. Arroyo, and S. Hassan, "Effect of the gas price surges on user activity in the DAOs of the Ethereum blockchain," in *Proc. CHI Conf. Hum. Factors Comput. Syst.*, 2021, pp. 1–7.

[10] I. Gilboa, "Expected utility with purely subjective non-additive probabilities," *J. Math. Econ.*, vol. 16, no. 1, pp. 65–88, 1987.

[11] D. Kahneman and A. Tversky, "Prospect theory: An analysis of decision under risk," in *Handbook of the Fundamentals of Financial Decision Making: Part I*. Singapore: World Scientific, 2013, pp. 99–127.

[12] H. Xu and B. Li, "Dynamic cloud pricing for revenue maximization," *IEEE Trans. Cloud Comput.*, vol. 1, no. 2, pp. 158–171, Jul.–Dec. 2013.

[13] M. Al-Roomi, S. Al-Ebrahim, S. Buqrais, and I. Ahmad, "Cloud computing pricing models: A survey," *Int. J. Grid Distrib. Comput.*, vol. 6, no. 5, pp. 93–106, 2013.

[14] Z. Xiong, S. Feng, W. Wang, D. Niyato, P. Wang, and Z. Han, "Cloud/fog computing resource management and pricing for blockchain networks," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4585–4600, Jun. 2019.

[15] Z. Xiong, D. Niyato, P. Wang, Z. Han, and Y. Zhang, "Dynamic pricing for revenue maximization in mobile social data market with network effects," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 1722–1737, Mar. 2020.

[16] C. Luo, Y.-F. Huang, and V. Gupta, "Stochastic dynamic pricing for EV charging stations with renewable integration and energy storage," *IEEE Trans. Smart Grid*, vol. 9, no. 2, pp. 1494–1505, Mar. 2018.

[17] S. Fan, H. Zhang, Z. Wang, and W. Cai, "Mobile devices strategies in blockchain-based federated learning: A dynamic game perspective," *IEEE Trans. Netw. Sci. Eng.*, early access, Apr 4 2022 doi: 10.1109/TNSE.2022.3163791.

[18] S. Fan, H. Zhang, Y. Zeng, and W. Cai, "Hybrid blockchain-based resource trading system for federated learning in edge computing," *IEEE Internet Things J.*, vol. 8, no. 4, pp. 2252–2264, Feb. 2021.

[19] D. T. Nguyen, H. T. Nguyen, and L. B. Le, "Dynamic pricing design for demand response integration in power distribution networks," *IEEE Trans. Power Syst.*, vol. 31, no. 5, pp. 3457–3472, Sep. 2016.

[20] J. He, Y. Wen, J. Huang, and D. Wu, "On the cost–QoE tradeoff for cloud-based video streaming under Amazon EC2's pricing models," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 4, pp. 669–680, Apr. 2014.

[21] J. Chen, C. Wang, B. B. Zhou, L. Sun, Y. C. Lee, and A. Y. Zomaya, "Tradeoffs between profit and customer satisfaction for service provisioning in the cloud," in *Proc. 20th Int. Symp. High Perform. Distrib. Comput.*, 2011, pp. 229–238.

[22] W. Warren and A. Bandeali, "0x: An open protocol for decentralized exchange on the ethereum blockchain," 2017, [Online]. Available: https://github.com/0xProject/whitepaper

[23] E. Hertzog, G. Benartzi, and G. Benartzi, "Bancor protocol," *Continuous Liquidity for Cryptographic Tokens Through Their Smart Contracts*, 2017. Accessed: Jun. 06, 2020. [Online]. Available: https://storage.googleapis.com/website-bancor/2018/04/01ba8253-bancor_protocol_whitepaper_en.pdf

[24] K. H. Al-Yahyaee, W. Mensi, H.-U. Ko, S.-M. Yoon, and S. H. Kang, "Why cryptocurrency markets are inefficient: The impact of liquidity and volatility," *North Amer. J. Econ. Finance*, vol. 52, 2020, Art. no. 101168.

[25] H. Adams, N. Zinsmeister, and D. Robinson, "Uniswap v2 core. 2020," 2020. [Online]. Available: https://uniswap.org/whitepaper.pdf

[26] S. Fan, T. Min, X. Wu, and C. Wei, "Towards understanding governance tokens in liquidity mining: A case study of decentralized exchanges," *World Wide Web*, pp. 1–20, 2022.

[27] T. Li and N. B. Mandayam, "Prospects in a wireless random access game," in *Proc. IEEE 46th Annu. Conf. Inf. Sci. Syst.*, 2012, pp. 1–6.

[28] L. Xiao, N. B. Mandayam, and H. V. Poor, "Prospect theoretic analysis of energy exchange among microgrids," *IEEE Trans. Smart Grid*, vol. 6, no. 1, pp. 63–72, Jan. 2015.

[29] Y. Wang, W. Saad, N. B. Mandayam, and H. V. Poor, "Integrating energy storage into the smart grid: A prospect theoretic approach," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2014, pp. 7779–7783.

[30] J. Yu, M. H. Cheung, and J. Huang, "Spectrum investment under uncertainty: A behavioral economics perspective," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 10, pp. 2667–2677, Oct. 2016.

[31] G. Angeris, H.-T. Kao, R. Chiang, C. Noyes, and T. Chitra, “An analysis of uniswap markets,” 2019, *arXiv:1911.03380*.

[32] L. Kiffer, D. Levin, and A. Mislove, “Analyzing Ethereum’s contract topology,” in *Proc. Internet Meas. Conf.*, 2018, pp. 494–499.

[33] E. Hertzog, G. Benartzi, and G. Benartzi, “Bancor protocol,” *Continuous Liquidity Cryptographic Tokens Smart Contracts*, 2017. Accessed: Jun. 6, 2020. [Online]. Available: https://storage. googleapis. com/ website-bancor/2018/04/01ba8253-bancor_protocol_whitepaper _en. pdf

[34] H. Zhang, S. Fan, and W. Cai, “Decentralized resource sharing platform for mobile edge computing,” in *Proc. Int. Conf. 5G Future Wireless Netw.*, 2020, pp. 101–113.

[35] P. Joseph and D. Thaddeus, “The bitcoin lightning network: Scalable off-chain instant payments,” 2016.

[36] D. Christian and W. Roger, “A fast and scalable payment network with bitcoin duplex micropayment channels,” in *Proc. Symp. Self-Stabilizing Syst.*, 2015, pp. 3–18.

[37] D. Prelec and G. Loewenstein, “Decision making over time and under uncertainty: A common approach,” *Manage. Sci.*, vol. 37, no. 7, pp. 770–786, 1991.

[38] J. Yu, M. H. Cheung, J. Huang, and H. V. Poor, “Mobile data trading: A behavioral economics perspective,” in *Proc. IEEE 13th Int. Symp. Model. Optim. Mobile Ad Hoc Wireless Netw.*, 2015, pp. 363–370.

[39] C. Huang, C. Hsu, Y. Chang, and K. Chen, “GamingAnywhere: An open cloud gaming system,” in *Proc. 4th ACM Multimedia Syst. Conf.*, 2013, pp. 36–47.

[40] G. Liao, X. Chen, and J. Huang, “Prospect theoretic analysis of privacy-preserving mechanism,” *IEEE/ACM Trans. Netw.*, vol. 28, no. 1, pp. 71–83, Feb. 2020.

[41] M. Abdellaoui, O. l’Haridon, and C. Paraschiv, “Experienced vs. described uncertainty: Do we need two prospect theory specifications?,” *Manage. Sci.*, vol. 57, no. 10, pp. 1879–1895, 2011.

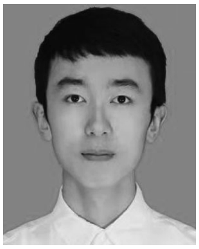
[42] M. Abdellaoui, H. Bleichrodt, and O. L’Haridon, “A tractable method to measure utility and loss aversion under prospect theory,” *J. Risk Uncertainty*, vol. 36, no. 3, pp. 245–266, 2008.

[43] M. Abdellaoui, H. Bleichrodt, and C. Paraschiv, “Loss aversion under prospect theory: A parameter-free measurement,” *Manage. Sci.*, vol. 53, no. 10, pp. 1659–1674, 2007.

[44] H. Fehr-Duda, M. De Gennaro, and R. Schubert, “Gender, financial risk, and probability weights,” *Theory Decis.*, vol. 60, no. 2, pp. 283–313, 2006.

Sizheng Fan (Graduate Student Member, IEEE) received the BEng degree in automation from the Beijing Institute of Technology, China, in 2018. He is currently working towards the PhD degree in computer and information engineering with the Chinese University of Hong Kong, Shenzhen, China. He is working as a research assistant with Human-Cloud Systems Laboratory. His current research interests include blockchain, DeFi, and Web 3.0. He is a student member of the ACM.

Juntao Zhao received the BEng degree in computer science and engineering from the Chinese University of Hong Kong, Shenzhen, China, in 2020. He is currently working towards the PhD degree with the University of Hong Kong, China. He was working as a research assistant with Human-Cloud Systems Laboratory. His current research interests include machine learning and distributed computing.



Rong Zhao received the BEng degree in communication engineering from the Southern University of Science and Technology, China, in 2018. He is currently working towards the PhD degree in computer and information engineering with the Chinese University of Hong Kong, Shenzhen, China. He is working as a research assistant with Human-Cloud Systems Laboratory. His current research interests include blockchain, game theory, and crowdsourcing.

Zehua Wang (Member, IEEE) received the bachelor’s degree from Wuhan University, the master’s degree from Memorial University, and the PhD degree from the University of British Columbia (UBC), Vancouver, in 2016 and was a postdoctoral research fellow with the Wireless Networks and Mobile Systems (WiNMoS) Laboratory directed by Prof. Victor C. M. Leung from Feb. 2017 to Aug. 2018. He is now an adjunct professor with the Department of Electrical and Computer Engineering, UBC, Vancouver and the CTO at Intellium Technology Inc., BC, Canada. He was a recipient of the Four Year doctoral fellowship with UBC from 2012 to 2016 and the graduate support initiative awards at UBC in 2014 and 2015. He received the Chinese Government Award for Outstanding Self-Financed Students Abroad in 2015. He is an editor of the *Wireless Networks* and served as the guest editors for a few special issues in top journals including the *IEEE Access* and *ACM/ Springer Mobile Networks & Applications*.

Wei Cai (Senior Member, IEEE) received the BEng degree in software engineering from Xiamen University, China, in 2008, the MS degree in electrical engineering and computer science from Seoul National University, Korea, in 2011, and the PhD degree in electrical and computer engineering from The University of British Columbia (UBC), Vancouver, Canada, in 2016. From 2016 to 2018, he was a postdoctoral research fellow with UBC. He is currently an assistant professor of computer engineering with the School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen. He is serving as the director of CUHK(SZ)-White Matrix Joint Metaverse Laboratory. He has co-authored more than 90 journal and conference papers in the areas of interactive multimedia and distributed/ decentralized systems. His recent research interests are mainly in the topic of human-centered computing for metaverse, including blockchain, digital games, Web3, and computational art. He is now serving as a TPC member for top conferences including ACM MM, MMSys, NOSSDAV, GameSys, associate editor for *IEEE Transactions on Cloud Computing*, and guest editor for many leading journals including *ACM Transactions on Multimedia Computing, Communications, and Applications*, *IEEE Multimedia*, *IEEE Transactions on Network Science and Engineering*, *IEEE Transactions on Computational Social Systems*, etc. He was a recipient of 6 Best Paper Awards. He is a member of the ACM.