SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

January 2023, Vol. 66 112105:1–112105:18
https://doi.org/10.1007/s11432-021-3362-9

# Transaction transmission model for blockchain channels based on non-cooperative games

Peiyun ZHANG[1*], Chenxi LI[2], Mengchu ZHOU[3*], Wenjun HUANG[1], Abdullah ABUSORRAH[4] & Omaimah O. BAMASAG[5]

[1]*Engineering Research Center of Digital Forensics of Ministry of Education, School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing* 210044*, China;*
[2]*School of Computer and Information, Anhui Normal University Wuhu, Anhui* 241003*, China;*
[3]*Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark* NJ 07102*, USA;*
[4]*Center of Research Excellence in Renewable Energy and Power Systems, Department of Electrical and Computer Engineering, Faculty of Engineering, and K.A. CARE Energy Research and Innovation Center, King Abdulaziz University, Jeddah* 21589*, Saudi Arabia;*
[5]*Department of Computer Science, Faculty of Computing and Information Technology, Center of Excellence in Smart Environment Research, King Abdulaziz University, Jeddah* 21589*, Saudi Arabia*

Received 16 August 2021/Revised 28 September 2021/Accepted 12 November 2021/Published online 16 December 2022

**Abstract** In blockchain networks, transactions can be transmitted through channels. The existing transmission methods depend on their routing information. If a node randomly chooses a channel to transmit a transaction, the transmission may be aborted due to insufficient funds (also called balance) or a low transmission rate. To increase the success rate and reduce transmission delay across all transactions, this work proposes a transaction transmission model for blockchain channels based on non-cooperative game theory. Channel balance, channel states, and transmission probability are fully considered. This work then presents an optimized channel transaction transmission algorithm. First, channel balances are analyzed and suitable channels are selected if their balance is sufficient. Second, a Nash equilibrium point is found by using an iterative sub-gradient method and its related channels are then used to transmit transactions. The proposed method is compared with two state-of-the-art approaches: SilentWhispers and SpeedyMurmurs. Experimental results show that the proposed method improves transmission success rate, reduces transmission delay, and effectively decreases transmission overhead in comparison with its two competitive peers.

**Keywords** Blockchain channel, transaction transmission, channel balance, channel state, channel transmission probability, non-cooperative game

**Citation** Zhang P Y, Li C X, Zhou M C, et al. Transaction transmission model for blockchain channels based on non-cooperative games. Sci China Inf Sci, 2023, 66(1): 112105, https://doi.org/10.1007/s11432-021-3362-9

## 1 Introduction

### 1.1 Background

Blockchains without permission, such as Bitcoin [1] and Ethereum [2], allow any participant to join or leave at any time and offer various applications [3–6]. These replace trusted third parties with a peer-to-peer network in which peers may not trust each other. However, the current scalability of blockchains is limited [7,8]. For example, a Bitcoin system takes an average of 10 minutes to confirm new transactions [9]. Channel technology is one of the primary methods to improve blockchain scalability [10]. Through peer-to-peer channels, transactions can be processed off-chain instead of on-chain [11]. A channel network is composed of multiple channels and nodes, where each channel may have some available funds, i.e., channel balance. Channel balances are deposited in a multi-signature address managed by both nodes. After a transaction is transmitted through a channel, the channel balance decreases with it. If channel balance is insufficient, it cannot transmit transactions.

* Corresponding author (email: zpy@nuist.edu.cn, zhou@njit.edu)

 

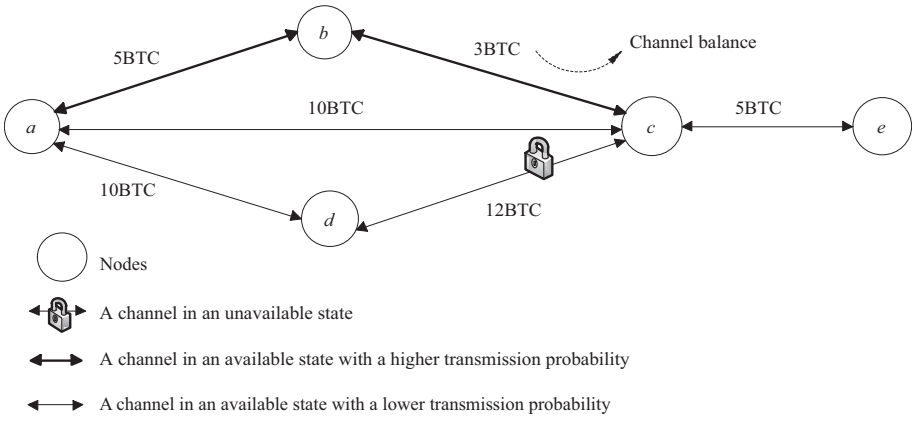


**Figure 1** Application scenario of a blockchain channel.

### 1.2 Problems and motivations

Some related concepts are given as follows.

(1) Insufficient channel balance. A node may choose any available channel to transmit a transaction. The chosen channel is required to have enough fund to finish such transmission. Hence, if it does not have enough fund, it cannot succeed in doing so, resulting in an insufficient channel balance issue.

(2) Unavailable state. A channel cannot transmit a transaction since it is occupied or down.

(3) Channel transmission probability. It is the probability for a channel to transmit at least one transaction in a time slot successfully.

Channel transmission performance is affected by insufficient balances, unavailable state, and channel transmission probability. An application scenario is shown in Figure 1. From Figure 1, we see that there are two channels with higher transmission probabilities (one is between nodes $a$ and $b$, and the other is between nodes $b$ and $c$). A channel between nodes $c$ and $d$ is in an unavailable state, and other channels (such as those between nodes $a$ and $c$, between $a$ and $d$, and between $c$ and $e$) are in an available state (i.e., a channel can transmit a transaction) but with a lower transmission probability. There may be three issues: insufficient channel balance, unavailable state, and low channel transmission probability.

(1) Insufficient channel balance. There is a transaction requiring node $b$ to transmit 6 BTC to node $c$. Channel balance between nodes $b$ and $c$ is insufficient to transmit the transaction. Thus, it should be transmitted through an intermediate node (e.g., node $a$). Otherwise, the transmission may be aborted. As channel balance may be insufficient due to a decrease in its transaction transmission, channels may fail to transmit a transaction and result in a longer waiting time. This may cause a low transmission success rate and a prolonged transmission delay. Channels in [12, 13] face the problem of insufficient channel balance and failure to complete transaction transmission.

(2) Unavailable state. There is a transaction requiring node $d$ to transmit 6 BTC to node $c$. However, the channel between nodes $d$ and $c$ is in an unavailable state. The transaction may be transmitted through other channels in an available state through an intermediate node (e.g., node $a$). Otherwise, transaction transmission may be aborted. This may cause a long transmission delay and a low transmission success rate. For channels in [14,15], the worst-case is usually caused by failing to reasonably consider the impact of the unavailable state of a channel on transaction transmission.

(3) Low channel transmission probability. We suppose that there is a transaction requiring node $a$ to transmit 6 BTC to node $c$. The transaction can be transmitted through other channels with high transmission probabilities, such as intermediate node $b$ in Figure 1. Nodes may select channels with high transmission probabilities. This may cause the balances of channels with high transmission probabilities to be continuously consumed and eventually lead to the failure of transaction transmission due to insufficient channel balances. In addition, multiple transactions may wait for these channels to transmit, which leads to a long delay. Both can hurt a transaction's transmission success rate and cause delay. Channels in [16, 17] fail to address the problem caused by low channel transmission probability.

In summary, most existing methods fail to improve transaction transmission performance under insufficient balances, unavailable states, and low transmission probability. They may affect transaction transmission performance in a channel and make transaction transmission aborted. In the end, they lead to a low transmission success rate and delayed channel transactions. Driven by the above problems, the

motivations of this work are to increase transmission success rate and decrease delay.

### 1.3 Contributions

To solve the above-mentioned problems, a transaction transmission model is proposed for blockchain channels based on the non-cooperative game theory. This work intends to make the following contributions:

(1) It proposes a transaction transmission model to optimize transmission performance in channels by considering channel balances, channel states, and transmission probabilities. The model is based on non-cooperative games to find each channel's Nash equilibrium point where a channel has its ideal transmission performance.

(2) Using the model, this work designs an optimized channel transaction transmission (OCTT) algorithm to increase transmission success rate and reduce transmission delay of transactions through channels. The remainder of the paper is organized as follows. Section 2 gives related work. Section 3 proposes a transaction transmission model. Section 4 proposes its solution algorithms. Section 5 presents experimental results. Section 6 draws conclusion and suggests future work.

## 2 Related work

Existing channels are divided into two main categories: payment and state channels [18].

(1) Payment channel. This type of channel allows nodes to establish a direct point-to-point payment. Nodes can maintain and update their ledgers privately. There are some representatives. Spilman [19] introduced a channel payment protocol based on a bitcoin system that realizes the instant confirmation of transactions. During a channel payment process, funds for transaction receivers always increase and receivers only publish their latest channel states on-chain. However, the protocol supports one-way payment only but not reverse one. Duplex micro-payment channel [12] extends Spilman's one-way payment to a "two-way" approach to realizing high scalability for digital payments in a Bitcoin system. The time lock function of bitcoin transactions makes transaction transmission through duplex micropayment channels safe. However, the use of time lock does not allow channel balances to be used until the time lock threshold is reached, even if transaction transmission is complete. Hence, using such a channel causes a long waiting time for a transaction to transmit. Lightning channel [13] is the first one to form a payment network (called a lightning network) with payment channels and improves the transaction throughput of a blockchain while realizing two-way payments. It can transfer payments between two nodes through a channel with a two-way payment. Because nodes adopt the shortest link algorithm to select a path to transmit transactions, this method fails to consider insufficient channel balance issues. Raiden channel [20] can implement complex intelligent contracts on Ethereum and supports one-way payments. Yet some smart contracts may block certain tokens of counterparties in Raiden, which leads to its poor scalability.

(2) State channel. This type of channel results from the optimization of a payment channel by supporting arbitrarily complex smart contracts and allowing participants to safely modify the locked part of blockchain states. There are some typical examples. Miller et al. [14] proposed the concept of Sprites channels to improve the efficiency of a lightning network. The worst case with this method is usually due to the failure to reasonably consider the impact of the channel being unavailable, i.e., the channel cannot transmit transactions or the insufficient balance prevents the selected channel from doing so. To solve the problem of insufficient channel balances, Khalil et al. [21] introduced a revive channel. Based on linear programming, their approach implements a persistent mechanism to adjust its balance among intermediate nodes in a blockchain network. However, its solution complexity is too high for some large-size cases. Such channels fail to handle problems caused by low transmission probability. Perun channel creates a virtual platform without interacting with intermediate transmission nodes for each payment. Dziembowski et al. [17] extended it and introduced a fully virtual state channel with an arbitrary length. This further reduces the delay and costs of transaction transmissions through such channels. However, this virtual channel method fails to consider the impact of channel transmission probability.

(3) Other channels. A counterfactual channel [22] is a general state channel that encapsulates functions and states in instantiated contracts. This allows users to deploy new functions to existing channels without contacting blockchains. Only users from a defined set can transmit transactions to each other. However, for a dynamic blockchain channel, other users (those who are not in the set) cannot transmit transactions

through such channels, which limits its applications. Other new blockchain channel technologies have been proposed for special tokens, which are bitcoin substitutions designed to improve the weak anonymity of blockchains, e.g., using zero-knowledge proof [23]. Zerocash achieves anonymity, which is the first completely anonymous ledger-based token [24]. However, it has the problem of poor scalability. Zhang et al. [15] improved Zerocash by supporting a multi-signature mechanism. The improved Zerocash is then used to build a small payment system called Z-Channel, which improves the scalability significantly and reduces the confirmation time of Zerocash payments. However, users who use Z-Channel must store hundreds of megabytes of public reference strings. This approach also needs a trusted third party to generate these strings, which requires high storage space and a trusted third party.

The above analysis indicates that channel technology is a feasible solution to improve the scalability of blockchains. All prior channels face two challenging issues: (a) how to increase transmission success rate when there are channels with insufficient balances; and (b) how to decrease transaction transmission delay when there are channels in unavailable states. We design a transaction transmission model to solve the above problems.

## 3 Proposed model

A uniform random blockchain channel network is given by: $G = \langle V, E \rangle$, where $V$ and $E$ represent a set of nodes and channels, respectively, with $l = |V|$ and $K = |E|$. Notations in this paper are given in Table A1 in Appendix A.

### 3.1 Related concepts

In a blockchain channel network, transactions are completely independent and broadcast to the network. Source and target nodes select channels to establish transmission links (a link in brief, which includes one or more channels) to transmit a transaction. Transaction transmission is a random event that occurs independently in disjoint time slots and only happens once over a sufficiently small slot. Hence, transaction arrivals follow a Poisson process [25].

Let $T_i^t$ be the total time to transmit transactions through channel $i \in \{1, 2, \ldots, m\}$ in a random time slot $t$, where $m$ is the number of channels with sufficient balances that make a link from a source node to a target one. Let $T$ be the time interval between the end of the current transaction transmission and the start of the next through channel $i$. Let $\alpha_i^t$ be channel transmission probability of channel $i$ to transmit at least one transaction when $T \leqslant T_i^t$. That is

$$\alpha_i^t = P\{T \leqslant T_i^t\} = 1 - \mathrm{e}^{-R_i T_i^t}, \tag{1}$$

where $R_i$ is the transmission rate of channel $i$. Note that each channel's transmission time is assumed to follow an exponential distribution [26]. Let $P_i^t$ be the probability that a node in a blockchain network can receive a transaction through channel $i$ during time slot $t$:

$$P_i^t = C_m^1 \alpha_i^t (1 - \alpha_i^t)^{m-1}. \tag{2}$$

Let $S_i^t$ be the probability of a node successfully transmitting a transaction through channel $i$ during time slot $t$:

$$S_i^t = \alpha_i^t P_i^t. \tag{3}$$

Let $\pi_1$ be the probability of all nodes successfully transmitting transactions during time slot $t$:

$$\pi_1 = \prod_{i=1}^{m} S_i^t, \tag{4}$$

where $i \in \{1, 2, \ldots, m\}$. Channel states can be either available or unavailable.

Available state. This indicates that a channel can transmit a transaction.

Unavailable state. This indicates that a channel cannot transmit a transaction because it is busy or broken down. Many reasons cause channel transmission failures, such as closure by one party, insufficient channel balance, and damage due to attacks from malicious internal nodes.

For a blockchain channel network, let $T_s$ be the time slot for channel $i$ to transmit a transaction in an available state:

$$T_s = \pi_1 T_d / m, \tag{5}$$

where $T_d$ is the time slot to transmit a transaction through a channel in a blockchain network. Based on the two types of states, $T_i^t$ is calculated as

$$T_i^t = \pi_1 T_s + \pi_0 T_f, \tag{6}$$

where $\pi_0 = 1 - \pi_1$ is the probability that channel $i$ is in an unavailable state during time slot $t$, and $T_f$ is the time slot of channel $i$ in an unavailable state.

(1) Transmission time. Let $X_i^t$ be the average number of transactions that are successfully transmitted from a node through channel $i$ during a unit time slot

$$X_i^t = 1/S_i^t, \tag{7}$$

where $S_i^t$ is given in Eq. (3). Let $\phi_i^t$ be the transmission time of transactions through channel $i$ during time slot $t$. Thus, we have

$$\phi_i^t = X_i^t T_i^t = \frac{\pi_1 T_s + \pi_0 T_f}{\alpha_i^t C_m^1 \alpha_i^t (1-\alpha_i^t)^{m-1}}. \tag{8}$$

(2) Deviation of channel transmission probability. Users may blindly select channels with high transmission probabilities instead of the ideal one, which is obtained by games. It is noted that an ideal transmission probability is not equal to the highest transmission probability. We use an indicator denoted as $\rho_i^t$ to measure the deviation of channel transmission probability between the current transmission probability and the ideal one of channel $i$ during time slot $t$:

$$\rho_i^t = k_i |\alpha_i^t - \gamma_i^t|, \tag{9}$$

where $k_i \geqslant 0$ is a correlation coefficient of channel transmission probability of channel $i$. And $\gamma_i^t$ is a Nash equilibrium solution, i.e., the ideal transmission probability of channel $i$. Thus, a blockchain channel system can optimize the transaction transmission performance of channels to achieve an ideal transmission for a blockchain system. To increase transmission success rate, nodes should select a channel with an ideal transmission probability. A smaller $\rho_i^t$ gives a higher transmission success rate. If $\rho_i^t$ is too large, the transmission probability of channel $i$ may deviate from the ideal case, which may lead to the failure of transaction transmission.

Transaction transmission depends on the transmission probability of a channel as well those of all its neighbors. In a channel network, there may be many transactions waiting for the same channel with high transmission probability, which leads to long transmission delays. Hence, we use a non-cooperative game to avoid this problem and decrease the transmission time to achieve an ideal transmission performance.

### 3.2 Transaction transmission model of blockchain channels

A transaction transmission model is proposed based on non-cooperative game theory with four basic elements: participants, strategy space, profit set, and Nash equilibrium points. Non-cooperative game theory [27] describes individual learning and adaptation strategies [28, 29] and can provide Nash equilibrium solutions.

(1) Participants. Channels with sufficient balances are taken as participants in a non-cooperative game to optimize transmission performance for transactions by adjusting their strategies. Let $I$ be the set of participants

$$I = \{1, 2, 3, \ldots, m\}. \tag{10}$$

Participants in a game are driven by their profits and transmit transactions based on game results.

(2) Strategy space. The strategy space of each participant is a set of transmission probabilities for channel transactions during time slot $t$. Let $\chi$ be the strategy space:

$$\chi = \langle \alpha_1^t, \alpha_2^t, \ldots, \alpha_i^t, \ldots, \alpha_m^t \rangle, \tag{11}$$

where $i \in I$ and $\alpha_i^t$ is the transmission probability of channel $i$. Note that the transmitting probability for every transaction is determined by respective channels. That is, for transactions transmitted through

channel $i$, transaction transmitting probability is $\alpha_i^t$. Let $\alpha_i^t = \langle \alpha_1^t, \ldots, \alpha_i^t, \alpha_{i+1}^t, \ldots, \alpha_m^t \rangle$ be the strategy space of the channels except for channel $i$.

(3) Profit set. We propose a channel utility function to obtain a profit set. Let $U_i^t$ be the normalized utility function of channel $i$ during time slot $t$:

$$U_i^t = \omega_1 \frac{\phi_i^t - \min \phi_i^t}{\max \phi_i^t - \min \phi_i^t} + \omega_2 \frac{\rho_i^t - \min \rho_i^t}{\max \rho_i^t - \min \rho_i^t}, \tag{12}$$

where $\omega_1$ and $\omega_1$ are the weighting factors satisfying $\omega_1 + \omega_2 = 1$. The values of $\omega_1$ and $\omega_1$ depend on transmission time or transmission probability of channel $i$. The profit set of non-cooperative games during time slot $t$ is denoted as

$$U = \{U_1^t, U_2^t, \ldots, U_i^t, \ldots, U_m^t\}, \tag{13}$$

where $i \in I$. The normalized utility function of channel $i$ can be expressed as $U_i^t(\alpha_i^t, \alpha_{-i}^t)$, where $(\alpha_i^t, \alpha_{-i}^t)$ is a strategic combination.

(4) Nash equilibrium points. For non-cooperative games, a Nash equilibrium is a stable state, i.e., no participant can obtain better benefits by unilaterally changing its strategies by deviating from a Nash equilibrium [30, 31].

The model is constructed based on the above four basic elements. For the normalized utility function, a smaller transmission time and a smaller deviation of transmission probability of a channel, a better transmission performance for channel transactions. Let $\Theta$ be the proposed transaction transmission model for blockchain channels (the proposed model in brief):

$$\Theta : \min\{U_i^t\} = \min \left\{ \omega_1 \frac{\phi_i^t - \min \phi_i^t}{\max \phi_i^t - \min \phi_i^t} + \omega_2 \frac{\rho_i^t - \min \rho_i^t}{\max \rho_i^t - \min \rho_i^t} \right\}, \tag{14}$$

$$0 \leqslant \gamma_i^t \leqslant \alpha_i^t \leqslant 1, \tag{15}$$

$$\sum_{i=1}^{m} \alpha_i^t \leqslant 1, \tag{16}$$

where (14)–(16) indicate that the transmission performance of a channel is affected by its transmission time and transmission probability. It is noted that $\gamma_i^t$ is the Nash equilibrium solution of channel $i$ and $\gamma_i^t = \arg\min_{\alpha_i^t \in (0,1]} \{U_i^t\}$. We use $\gamma_{-i}^t$ to denote the Nash equilibrium solution including $m-1$ channels but not channel $i$, and $(\gamma_i^t, \gamma_{-i}^t)$ to represent the Nash equilibrium point of the non-cooperative games. A theoretical description of a solution's optimality analysis can be found in [32].

## 4 Optimized channel transaction transmission (OCTT) algorithm

Let $C$ be the channel attribute structure of channel $i$

$$C = \langle \text{sID}, \text{tID}, \text{balance}, u_i^t \rangle, \tag{17}$$

where sID and tID are the IDs of the source and target nodes, respectively, balance is the current balance of channel $i$, and $u_i^t$ is the channel steady-state transmission rate, which indicates the stability of a transaction being successfully transmitted through channel $i$

$$u_i^t = \alpha_i^t T_s / T_i^t, \tag{18}$$

where $\alpha_i^t$ is the transmission probability of channel $i$, $T_s$ is the time slot of channel $i$ in an available state, and $T_i^t$ is as given in Eq. (6).

A link may contain multiple channels whose transmission rates could be different. To make a transaction transmit within a single-hop distance in a link, we analyze channel steady-state transmission rates for each link. The steady-state transmission rate of each channel varies with time slots using an iterative gradient method. Let $u_i^{t+1}$ be the iterative update for the steady-state transmission rate of channel $i$ during time slot $t+1$

$$u_i^{t+1} = \alpha_i^{t+1} T_s / T_i^{t+1}. \tag{19}$$

We propose an optimized channel transaction transmission (OCTT) algorithm based on the proposed model. Its flow chart is shown in Figure 2. OCTT calls two primary sub-algorithms: the channel

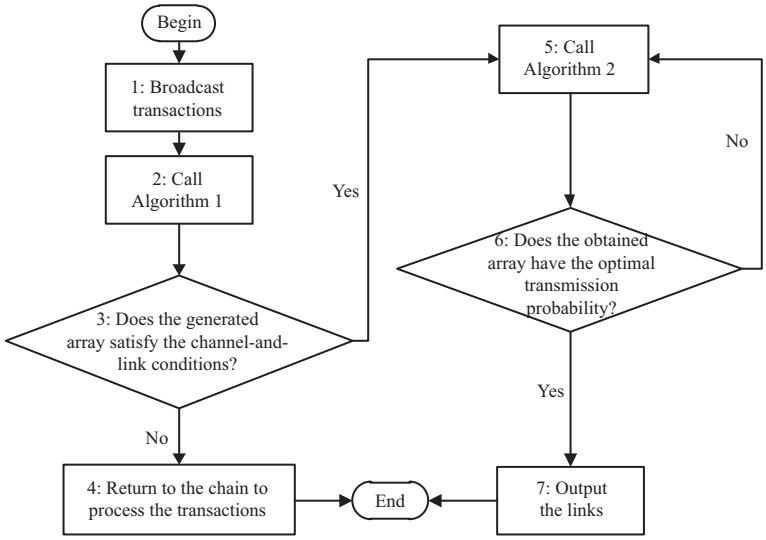


**Figure 2** Flow chart of OCTT algorithm.

analysis algorithm (Algorithm 1) and the iterative sub-gradient transmission algorithm (Algorithm 2). Channel-and-link conditions need to be satisfied in Algorithm 1.

(1) Channel condition. The selected channels should satisfy two conditions. (a) The balance in a channel is sufficient, which indicates that it is greater than or equal to the transaction amount. (b) The source and target nodes of each link are consistent with the transaction initiator and recipient.

(2) Link condition. Each link has a unique maximum steady-state transmission rate to reduce the effects of different steady-state transmission rates for the channels through the same link. Otherwise, a transaction cannot be transmitted within a single-hop distance in a link.

There are three critical issues in a blockchain channel network: insufficient channel balance, unavailable state, and low transmission probability. To solve the first issue, Algorithm 1 is designed to analyze channel balances and channel steady-state transmission rates. A link array is then obtained following Algorithm 1 by selecting channels that satisfy the channel-and-link conditions. To solve the next two issues, Algorithm 2 is invoked using the results of Algorithm 1, and the links with ideal transmission probabilities are obtained to transmit transactions.

The detailed explanations about Figure 2 are as follows.

Step 1. Source nodes initiate transactions and broadcast them in a blockchain channel network.

Steps 2–4. Based on the transactions, OCTT calls Algorithm 1 to analyze channel balances and channel steady-state transmission rates (Step 2). If an array satisfying the channel-and-link conditions is generated (Step 3), OCTT calls Algorithm 2 (Step 5) and proceeds to Step 6. Otherwise, it shows that there is no channel satisfying the channel-and-link conditions. The transactions are then returned on-chain to process (Step 4) and OCTT ends.

Steps 5, 6. The array obtained in Algorithm 1 is used in Algorithm 2 to guide whether each channel reaches the Nash equilibrium point $(\gamma_i^t, \gamma_{-i}^t)$ based on non-cooperative games. If not, Step 5 is repeated to call Algorithm 2.

Step 7. OCTT outputs the link with an ideal transmission probability and ends.

### 4.1 Channel analysis algorithm

We design the channel analysis algorithm to obtain a link array satisfying the channel-and-link conditions, as shown in Algorithm 1.

Some remarks about Algorithm 1 are given as follows.

(1) Algorithm 1 aims to obtain a link array satisfying the channel-and-link conditions during time slot $t$. For a transaction amount $Q$, there is a unique array of channels, which varies during time slot $t$.

(2) If the balance for each channel in a blockchain network $G$ is greater than or equal to $Q$ (Line 8), the next-hop channel is found based on the source and target nodes corresponding to a given transaction (Lines 9–27).

(3) If the source node is in the current channel corresponding to the transaction, Algorithm 1 first calculates the steady-state transmission rate of the current channel (Lines 12, 13). It then obtains a maximum channel steady-state transmission rate through a comparison with the other channels in the same link. Finally, it assigns the maximum to the corresponding channel (Lines 14–16). The assignment aims to make all channels in the same link transmit each transaction within a single-hop distance. Each channel only uses the steady-state transmission rate between its single-hop neighboring channels to transmit transactions.

(4) Algorithm 1 adds the condition-satisfied channel to $\iota$ (Line 17) and accumulates the number of channels (Line 18). The target node of the channel is used as the source node of the next-hop to search for the next-hop channel (Line 19) until the target node of the queried channel is consistent with the target node corresponding to the transaction. That is, if a link satisfying the channel-and-link conditions is found, Algorithm 1 exits the current loop. The algorithm runs again (Line 9) according to the ID of the source node for the transaction (Line 24).

(5) The numbers of rows (Line 27) and columns (Line 22) of the array are obtained. When time slot $t$ changes, the steady-state transmission rate of channel $i$ during time slot $t+1$ is updated from (19) (Line 31). There is one loop in Algorithm 1. Thus, its time complexity is $O(m)$.

**Algorithm 1** Channel analysis

**Input:** $s$, $e$, $Q$, $G$. /*$s$ and $e$ are the IDs of the source and target nodes, respectively, $Q$ is the transaction amount, and $G$ is the network topology diagram*/

**Output:** $\mathcal{L}$, maxrow, maxcolumn, $m$. /*$\mathcal{L}$ is the link array satisfying the channel-and-link conditions. maxrow and maxcolumn are the numbers of rows and columns in $\mathcal{L}$, respectively. $m$ is the number of channels with sufficient balances that link a source node and a target one*/

1: **Declare** $\mathcal{L}$, maxrow, maxcolumn, maxcsstr, $m$, count, $z$, $k$; /*maxcsstr is a temporary variable that stores the maximum
2: steady-state transmission rate of a link, count is a temporary variable that stores the number of channels with sufficient balances,
3: and $z$ is a temporary variable that stores the ID of the source node of the next-hop channel*/
4: **Declare** $C$; /*The channel attribute structure of channel $i$*/
5: $\mathcal{L} \leftarrow$ null; maxrow $\leftarrow 0$; maxcolumn $\leftarrow 0$; $m \leftarrow 0$; count $\leftarrow 0$; $z \leftarrow 0$; $k \leftarrow 0$;
6: $C \leftarrow \langle \text{sID}, \text{tID}, \text{balance}, u_i^t \rangle$;
7: **For**(each channel $C$ in $G$) /*During each time slot $t$*/
8: **If**($C$.balance $\geqslant Q$)
9: **If**($C$.sID == $s$)
10: $j \leftarrow 0$;
11: **If**($C$.sID == $r$ && $C$.tID! = $e$)
12: $i$ = count; /*The $i$-th channel in $\mathcal{L}$*/
13: $C.u_i^t \leftarrow \frac{\alpha_i^t T_s}{T_i^t}$; /*Calculate channel steady-state transmission rate $u_i^t$ via (18) of channel $i$*/
14: **If**(maxcsstr $\leqslant u_i^t$) /*Unified maximum channel steady-state transmission rate of the current link of $\mathcal{L}$*/
15: $C.u_i^t \leftarrow$ maxcsstr;
16: **EndIf**
17: $\mathcal{L}[k][j++] \leftarrow C$;
18: count++;
19: $z \leftarrow C$.tID; /*Store the ID of the source node to the next-hop channel*/
20: **EndIf**
21: **If**(maxcolumn $\leqslant j$)
22: maxcolumn $\leftarrow j+1$; /*Column number of $\mathcal{L}$*/
23: **EndIf**
24: $z \leftarrow s$; /*Store the ID of the source node to $z$*/
25: $k++$; /*Traverse the next transmission link*/
26: **EndIf**
27: maxrow $\leftarrow k+1$; /*Row number of $\mathcal{L}$*/
28: **EndIf**
29: **EndFor**
30: $m \leftarrow$ count;
31: $u_i^{t+1} \leftarrow \frac{\alpha_i^{t+1} T_s}{T_i^{t+1}}$; /*Update channel steady-state transmission rate via (19) during time slot $(t+1)$*/
32: **Return** $\mathcal{L}$, maxrow, maxcolumn, $m$.

### 4.2 Iterative sub-gradient transmission algorithm

Algorithm 1 provides the maximum steady-state transmission rate of each link and a link array and solves the critical issue of an insufficient channel balance. To solve other critical issues of unavailable state and low transmission probability, we design Algorithm 2 based on Algorithm 1 to allow the transmission probability of each channel to reach a global optimum (i.e., an ideal transmission probability and the Pareto Nash equilibrium point of the proposed model). Algorithm 2 uses an iterative sub-gradient

**Algorithm 2** Iterative sub-gradient transmission

**Input:** $\mathcal{L}$, $t$. /*$\mathcal{L}$ is from Algorithm 1 and $t$ is a random time slot*/
**Output:** $\mathcal{L}$, $r$. /*$r$ is the number of iterations*/
1: **Declare** $r$, $\mu_i^r$, $Ri$, $\varepsilon$, $\phi$; /*$\mu_i^r$ is the Lagrange multiplier of channel $i$ in the $r$-th iteration, $R_i$ is the transmission probability
2: of channel $i$, $\varepsilon$ is the upper bound of step $\sigma_\mu^t$, and $\phi$ is the threshold*/
3: $C \leftarrow \langle \text{sID}, \text{tID}, \text{balance}, \text{csstr} \rangle$;
4: $r \leftarrow 0$; $i \leftarrow 0$;
5: **Do**
6: $r \leftarrow r + 1$;
7: **For**(each iteration $r$)
8: $\alpha_i^t \leftarrow 1 - \exp\{-R_i T_i^t\}$; /*Based on $R_i$, the transmission probability $\alpha_i^t$ for channel $i$ is computed via (1)*/
9: $\sigma_\mu^t \leftarrow \varepsilon/t$; /*Calculate the step size $\sigma_\mu^t$ via $\varepsilon$ and $t$*/
10: $r \leftarrow r + 1$;
11: $\mu_i^{r+1} \leftarrow [\mu_i^r + \sigma_\mu^t(\alpha_i^t - \alpha_i^*)]^-$; /*Update the Lagrange multiplier $\mu_i^{r+1}$ via (19) during time slot $t$*/
12: **EndFor**
13: **Until** $|\mu_i^{r+1} - \mu_i^r|/\mu_i^{r+1} > \phi$;
14: **For**(each channel $i$)
15: **Do**
16: Send $\alpha_i^t$ and the multiplier $\mu_i^r$ to channel $i$;
17: Calculate $L_d(\alpha_i^t, \mu_i^r)$ via $\alpha_i^t$ and $\mu_i^r$;
18: Calculate $L_d(\alpha_i^t, \mu_i^{r+1})$ via $\alpha_i^t$ and $\mu_i^{r+1}$;
19: **Until** $L_d(\alpha_i^t, \mu_i^{r+1}) \leqslant L_d(\alpha_i^t, \mu_i^r)$;
20: $\alpha_i^* \leftarrow \text{argmin}_{0 \leqslant \gamma_i^t \leqslant \alpha_i^t \leqslant 1}\{L_d(\alpha_i^t, \mu_i^r)\}$;
21: **EndFor**
22: **Return** $\mathcal{L}$, $r - 1$. /*$\mathcal{L}$ has the ideal transmission probability and $r - 1$ is the number of iterations*/

transmission based on the Lagrange equation. Firstly, the Lagrange equation of the model is defined as

$$L(\alpha_i^t, \mu_i^r, \nu_i) = U_i^t + \mu_i^r(\alpha_i^t - \gamma_i^t) + \nu_i(\alpha_i^t - 1), \tag{20}$$

where $\mu_i^r$ and $\nu_i^r$ are Lagrange multipliers related to the constraints in (15) and (16), respectively, and $\mu_i^r$ is the Lagrange multiplier of the $r$-th iteration of channel $i$. We have $\mu_i^r \geqslant 0$ and $\nu_i^r \geqslant 0$. For a given $\mu_i^r$, the Lagrange dual-function $L_d(\alpha_i^t, \mu_i^r)$ is the maximized $L(\alpha_i^t, \mu_i^r, \nu_i)$ about $\alpha_i^t$ and $\mu_i^t$, where $(\alpha_i^t, \mu_i^r)$ is a dual variable. Then, the Lagrange dual-function is defined

$$\min\{L_d(\alpha_i^t, \mu_i^r)\} = \max\{L(\alpha_i^t, \mu_i^r, \nu_i)\}. \tag{21}$$

To obtain the solution of (21), we use the iterative sub-gradient method to update the Lagrange multiplier and find the final invariant Lagrange multiplier $\mu_i^t$. Given a set of non-negative $\mu_i^t$, the ideal solution for the transmission probability of the dual-function function is

$$\underset{0 \leqslant \gamma_i^t \leqslant \alpha_i^t \leqslant 1}{\arg\min}\ \{L_d(\alpha_i^t, \mu_i^r)\}. \tag{22}$$

For a set of Lagrange multipliers, the gradient definition of the dual-function $L_d(\alpha_i^t, \mu_i^r)$ is

$$\frac{\partial L_d(\alpha_i^t, \mu_i^r)}{\partial \mu_i^r} = \alpha_i^t - \alpha_i^*. \tag{23}$$

According to the iterative sub-gradient method, the Lagrange multiplier is updated using the gradient as

$$\mu_i^{r+1} = [\mu_i^r + \sigma_\mu^t(\alpha_i^t - \alpha_i^*)], \tag{24}$$

where $[\cdot]^- = \min(\cdot)$, $\theta_\nu^t$ is the step size, $\varepsilon$ is the upper bound of $\theta_\nu^t$ with $\varepsilon \geqslant 0$, and $\theta_\nu^t = \varepsilon/t$. The $\theta_\nu^t$ causes the Lagrange multiplier to converge to a constant value of $\mu_i^r$ after $r$ iterations. Then, the ideal solution is obtained based on (22).

Then, Algorithm 2 can iteratively find a Lagrange multiplier during time slot $t$ that converges to a constant after iterations. Finally, the link with ideal transmission probability $\alpha_i^*$ is obtained based on the Lagrange multiplier. Some remarks regarding Algorithm 2 are described as follows.

(1) The input of $\iota$ for Algorithm 2 is from Algorithm 1. Algorithm 2 has two loops, both of which are double nested. The first one (Lines 5–13) uses the iterative sub-gradient method to find changeless in the Lagrange multiplier for each channel. After $r$ iterations, if the Lagrange multiplier does not change once it converges to $\mu_i^r$, then $\mu_i^r$ is the Lagrange multiplier of channel $i$. Therefore, the ideal solution of $\Theta$ is obtained.

(2) In the first nested loop (Lines 7–12), step size $\theta_{\mu}^{t}$ controls the convergence speed of Algorithm 2. An increased $t$ can improve the convergence speed; however, a too large $\theta_{\mu}^{t}$ may cause fluctuations in convergence rates. As Algorithm 2 is convergent, any initial multiplier $\mu_i^r$ is valid. The upper bound of $\theta_{\mu}^{t}$ is set to $\varepsilon$ so that Algorithm 2 can obtain a better convergence.

(3) The second nested loop (Lines 16–19) uses the obtained Lagrange multiplier in the first nested loop to find the minimum of the Lagrange dual-function $L_d(\alpha_i^t, \mu_i^r)$. Once reaching the minimum, the value of the corresponding independent variable is the ideal solution, i.e., $\alpha_i^*$ of $\Theta$ (Line 20), which is the optimization goal of the proposed model.

(4) Finally, according to $\alpha_i^*$, Algorithm 2 outputs $\iota$ and $r-1$ (Line 22). Algorithm 2 has a nested loop to traverse all channel elements in $\iota$. Thus, its time complexity is $O(n^2)$. Based on the analysis of Algorithms 1 and 2, the time complexity of OCTT is $O(m^2)+O(n^2)$. The specific deployment process of the proposed model is given as follows.

(1) A blockchain channel network initiates a transaction $T$, generates a certain number of messages for $T$, and broadcasts the messages to the network. The proposed model calls OCTT. OCTT analyzes the channel balances and channel steady-state transmission rates. Thus, an array is generated to satisfy the channel-and-link conditions.

(2) Based on the array, the model guides each channel to reach a Nash equilibrium point through non-cooperative games. Then, OCTT outputs one or more links with an ideal transmission probability.

(3) Nodes choose channels from the network to transmit $T$ according to the output of OCTT. A Nash equilibrium point is obtained through non-cooperative games, which means that a channel has an ideal transmission performance at the point. All algorithms are deployed through smart contracts and executed automatically when conditions are satisfied.

## 5 Experiments

### 5.1 Experimental settings

(1) Experimental environment and tools. Omnet++ 5.0 [33] and OverSim module [34] are used to simulate a blockchain channel network. The former is used to build the network [35], and the latter is an open-source module based on Omnet++, which is used to simulate a P2P network with hundreds of thousands of nodes. The simulator eliminates encryption overhead and facilitates statistical analyses when simulating large-scale blockchain channel networks. Experimental environments include a CPU (Intel Core i5, 4.2 GHz), Memory (16 GB), and Windows 10.

(2) Dataset and experimental parameters. Because there is no available transaction tracking dataset from a real payment network, similarly to [36, 37], we use the dataset obtained from Ripple [38] with the network topology and transaction information from January 2013 to November 2016. As a credit network, Ripple uses blockchain technology to enable real-time settlement. Ripple is an on-chain payment network, where transaction traces record the real transaction needs of real users. Hence, it can be used to generate transaction requests in the following experiments. The original dataset has 93000 nodes and 330000 edges. We remove some nodes without any neighbors and some edges without funds. Thus, the dataset includes 5929 nodes and 35233 edges. Each edge indicates a channel with some balances. The used parameters are shown in Table 1.

Transaction count varies from 1000 to 45000. The average transaction size is set to approximately 300 bytes based on a Bitcoin system. Similar to [39, 40], all experiments are repeated five times and average values are used.

(3) Compared methods. We compare the proposed method with two state-of-the-art approaches applied to blockchain channel networks: SilentWhispers [41] and SpeedyMurmurs [38], which are widely used as benchmark methods in related studies, e.g., [37, 42].

(1) SilentWhispers. This approach uses landmark-centered routing to discover multiple links by randomly choosing landmarks. To make a quick private settlement and payment, it performs multi-party calculations to determine channel balances sent through each link. The landmark-centered routing method is an approximation that only calculates a subset of all possible links between senders and receivers. The nodes corresponding to the links in this subset are called landmarks.

(2) SpeedyMurmurs. As the most advanced routing algorithm in blockchain payment channels, this approach extends VOUTE [43] to deliver transactions in a P2P network with limited routing by randomly

**Table 1** Experimental parameters

| Parameter | Description | Value |
|---|---|---|
| $L$ | Node count | [1000, 4000] |
| $K$ | Channel count | [2000, 14000] |
| $N_c$ | Transaction count | 1000, 2000, 3000, ..., and 45000 |
| $V$ | Transaction size | 300 bytes |
| $W$ | Network bandwidth | 20 Mbps |
| $R_i$ | Transmission rate | 815, 820, 825, and 830 Tx/s |
| $T_d$ | Time slot to transmit a transaction through a channel | 1000 μs |
| $T_f$ | Time slot of channel $i$ in an unavailable state | 22 μs |
| $\mu_i^t$ | Lagrange multiplier of channel $i$ in the $r$-th iteration | 0.05 |
| $E$ | Upper bound of step $\sigma_\mu^t$ | 0.01 |
| $t$ | Random time slot | 20 μs |
| $\Phi$ | Threshold in Algorithm 2 | 1E−13 |
| $\omega_1, \omega_2$ | The weighting factors in (12) | 0.5 |

selecting landmarks.

Both methods are based on landmark routing, which cannot include all transaction transmission links in a network into a spanning tree. There may be no feasible transaction transmission link in the spanning tree based on landmarks, and the failure of link search may also lead to the decline of transaction transmission success rate. The defect of a tree structure itself may also lead to link lookup failure and ultimately transaction transmission failure. Hence, both transmission success rates are lower than that of the proposed method, to be shown later.

### 5.2 Performance metrics

(a) Transaction receiving rate

$$\dot{R} = \sum_{i=1}^{m} (S_i^t / T_i^t)/m, \tag{25}$$

where $S_i^t$ is the transmission probability that a transaction is successfully transmitted through channel $i$ during time slot $t$ (see Eq. (3)). The $T_i^t$ is given in (6) and $m$ is the number of channels with sufficient balances that make a link between a source node and a target one.

(b) Transmission success rate

$$S = \sum_{i=1}^{m} N_i^s / \sum_{i=1}^{m} N_i^g, \tag{26}$$

where $N_i^s$ is the number of transactions successfully transmitted through channel $i$ and received by a target node, and $N_i^g$ is the total number of transactions broadcasted from a source node and transmitted through channel $i$.

(c) The number of channels required to transmit a transaction

$$A = \sum_{k=1}^{H} |L_k|/H, \tag{27}$$

where $|L_k|$ is the length of the $k$-th transmission link (i.e., channel count in link $k$), and $H$ is the number of transmission links. As transactions can be transmitted through a link with one or more channels, transmission delay is related to the average number of channels required to transmit a transaction. The greater the average number of channels required, the longer the transmission delay.

(d) Transaction count during time slot $[t, t+1]$:

$$N_M = N_{t+1} - N_t, \tag{28}$$

where $N_{t+1}$ and $N_t$ are the total numbers of transactions before time slot $(t+1)$ and after time slot $t$, respectively. Transaction count affects transmission overhead. Hence, more transactions may generate a greater transmission overhead.

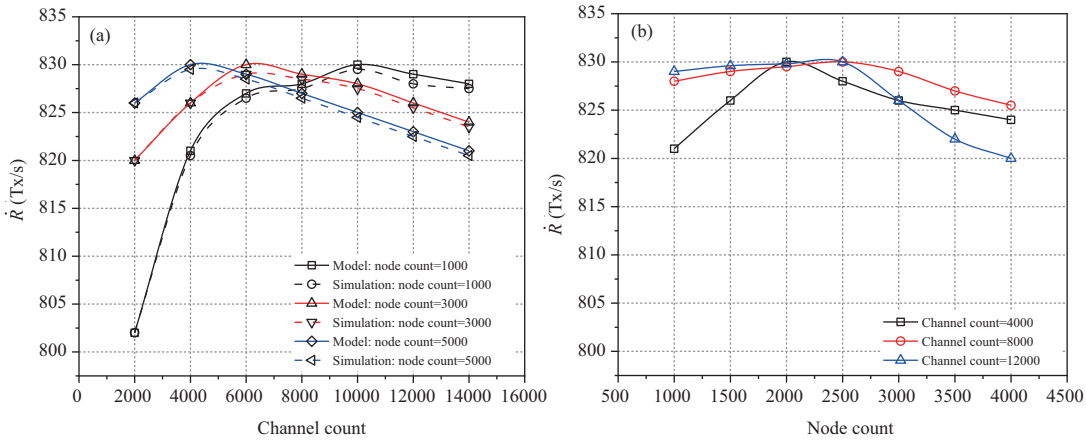


**Figure 3** (Color online) (a) Channel count vs. transaction receiving rate; (b) node count vs. transaction receiving rate.

### 5.3 Experimental results and analysis

(1) Transaction receiving rate. Transaction receiving rate primarily depends on transmission time and the probability of a node to successfully transmit a transaction. Both of them are affected by the number of adjacent channels and nodes.

(a) Channel count vs. transaction receiving rate. In Figure 3(a), node counts in three groups are 1000, 3000, and 5000, respectively. Channel count varies from 2000 to 16000. The channel counts are small in the initial stage. They have sufficient balances to successfully transmit transactions. Therefore, the transaction receiving rate quickly increases to a maximum then slowly decreases with channel counts. 830 Tx/s is the maximum transaction receiving rate corresponding to the different numbers of channels and nodes. The transaction receiving rate then starts to decrease after this value for the following reasons. For a given channel count, a larger node count gives a lower channel network density and larger network sparsity. The network is linked with both nodes and channels. It is noted that the channel network density refers to the channel scale over a blockchain channel network. The greater (smaller) the transaction receiving rate, the greater the density (sparsity). On the one hand, many transactions may have conflicts due to competition with the same channel to transmit transactions. On the other hand, channel balances may be constantly consumed. Lower balances indicate fewer transactions can be successfully transmitted. We compare the experimental results with the calculations based on the proposed model in terms of the transaction receiving rate. We can see that the calculated and experimental results are well fit, which shows that the proposed model conforms to real data and verifies its correctness.

(b) Node count vs. transaction receiving rate. The following experiments are conducted to analyze the transaction receiving rates by varying the number of adjacent nodes, as shown in Figure 3(b). Channel counts are set to 4000, 8000, and 12000, respectively. And the node count varies from 1000 to 4000. All channels initially use the same transmission rate. For each given channel count, the transaction receiving rate first increases to the maximum of 830 Tx/s and then decreases with node count. Initially, when node counts are small, more channels give a higher channel network density. Therefore, more transactions are successfully transmitted. As the node count increases, more channels give a higher transaction receiving rate. However, when the node count continues to increase, if the channel count does not change, the blockchain channel network may become sparser. Then, more transactions may be transmitted by existing channels simultaneously. On the one hand, channel balances are consumed with the number of transmitted transactions and may be insufficient. Hence, the network may cause transaction transmission failure and a low transmission success rate. On the other hand, many transactions need to be transmitted through the same channels, which may constantly lead to longer waiting time and may drop the transaction receiving rate.

(2) Evolution process of the transmission rate ($R_i$). The evolution process of the transmission rate ($R_i$ in Eq. (1)) is shown in Figure 4. From Figure 3, we see that the maximum transaction receiving rate is 830 Tx/s before decreasing with the number of channels and nodes. Therefore, the upper bound of the transmission rates is set to 830 Tx/s. The node count is set to 3000. Four thousand channels are randomly selected and then divided into four groups. The transmission rates of each group are set to 815, 820, 825, and 830 Tx/s, respectively. From Figure 4, we can see that the proposed method can

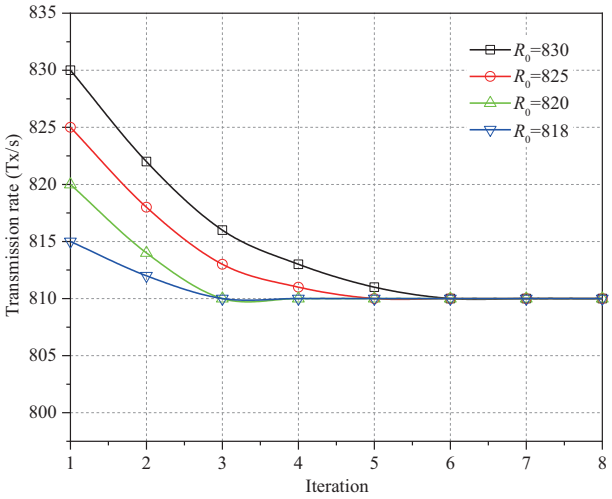


**Figure 4** (Color online) Evolution of the transmission rate with different initial values.

successfully guide the four groups of experiments to converge to 810 Tx/s, i.e., the ideal transmission rate after games.

(3) Transmission success rates.

(a) Channel count vs. transmission success rate. The results of the channel count vs. the transmission success rate are shown in Figure 5(a) when the node count is set to 3000 and the channel count varies from 2000 to 14000. The transmission success rate of the proposed method increases with channel count and tends to be stable when channel count is greater than 10000. The transmission success rate from SilentWhispers increases slowly and then tends to be stable, while the transmission success rate of SpeedyMurmurs decreases slowly. For a given node count, when the channel count increases, the channel network density may increase, which causes an increased link count following the proposed method. The size of spanning trees constructed in SilentWhispers also grows, which increases the number of feasible links between nodes. Therefore, the transmission success rates of the proposed method and SilentWhispers increase with the channel count. As SpeedyMurmurs lacks the consideration of changing channel counts, its transmission success rate decreases slightly with the channel count.

(b) Node count vs. transmission success rate. The results of the node count vs. the transmission success rate are shown in Figure 5(b), where the number of channels is 6000, and the node count varies from 1000 to 4000. We see that the transmission success rates of the proposed method and SilentWhispers increase with node count. However, the transmission success rate of SpeedyMurmurs decreases with the node count but is still smaller than that of the proposed method when the node count is above 1500.

The proposed method considers the blockchain channel network as a whole and stores all links. It also considers balances, channel states, and channel transmission probability to improve the transmission success rates for channel transactions. The spanning-tree structure used in SilentWhispers and SpeedyMurmurs has some inherent disadvantages as it only reflects the connection states of a local network over a certain time slot and cannot adapt to network changes. Therefore, their transmission success rates are lower than that of the proposed method.

(c) Transaction count vs. transmission success rate. The results of the transaction count vs. the transmission success rate are shown in Figure 5(c), where the numbers of channels and nodes are set to 6000 and 3000, respectively. The trend of the transmission success rate is observed by varying the transaction count. When the transaction count varies from 1000 to 45000, the proposed method has a higher transmission success rate than either the SilentWhispers or SpeedyMurmurs, which rely on landmark-centered routing. However, the proposed method analyzes the transmission performance of channel transactions from three aspects: channel balances, channel states, and transmission probability. It selects suitable channels with sufficient channel balances. The ideal transmission probability is obtained to transmit a transaction based on non-cooperative games, which improves the transmission success rate. As non-cooperative games have better adaptability to dynamically changing networks, the proposed method has better adaptability to dynamic transaction counts. Thus, its transmission success rate is higher than its two peers. As the channel utility function is the smallest when using the ideal transmission probability, the proposed method allows a channel network more time to transmit more transactions. Therefore, the

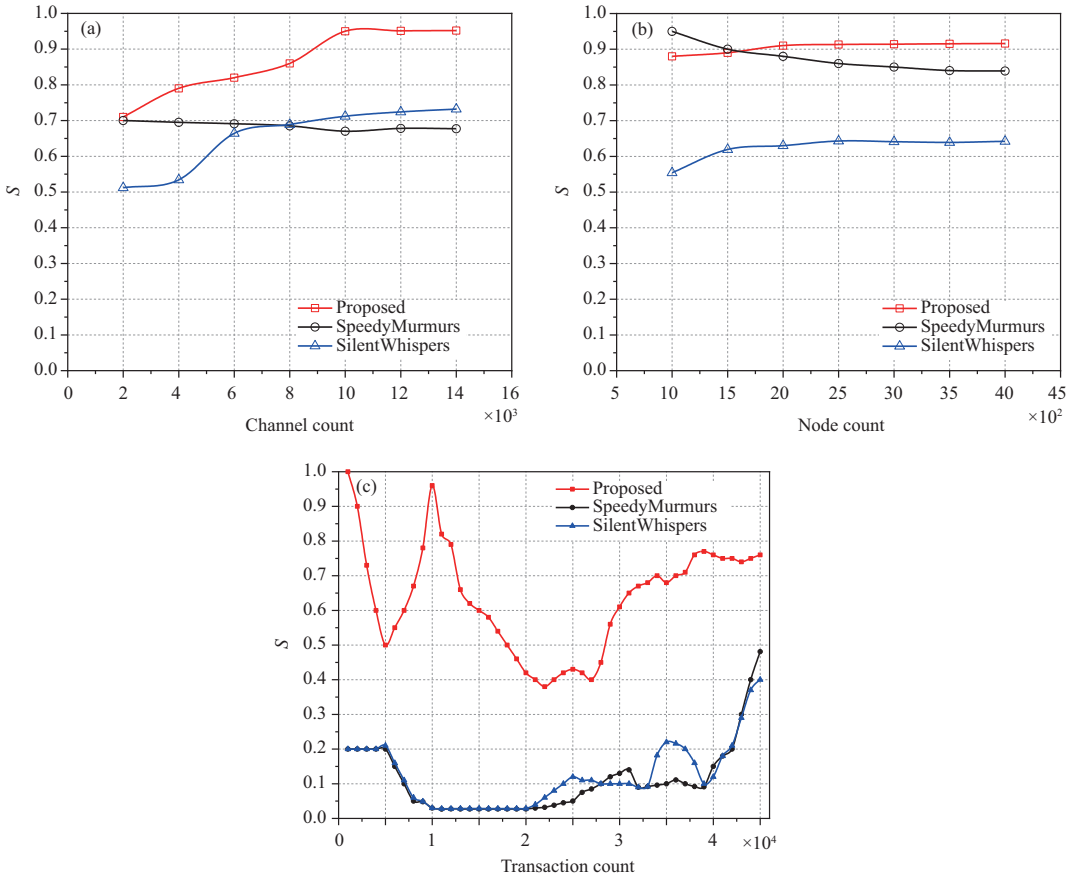


**Figure 5** (Color online) (a) Channel count vs. transmission success rate; (b) node count vs. transmission success rate; (c) transaction count vs. transmission success rate.

transmission success rate can increase to a certain extent. Because all transmission links are not included in a spanning tree using landmark-centered routing following SilentWhispers and SpeedyMurmurs, a feasible transmission link may not be found in the tree. Hence, the failure of a link search leads to a decreased transmission success rate. As mentioned before, the defect of a tree structure itself may also produce a failed link search, which may result in the failure of transaction transmission.

(4) Transmission delay.

(a) Channel count vs. transmission delay. The results of the channel count vs. the transmission delay are shown in Figure 6(a), where the node count is set to 3000, and the channel count varies from 2000 to 14000. For all three methods, the average number of channels required to transmit a transaction increases with channel count. The proposed method has the least required channels, and its associated transmission delay increases slightly with the channel count but is smaller than that of its peers. The main reason is that the proposed method selects channels based on non-cooperative games and makes each channel have the ideal transmission performance. Thus, the channel count has little influence on the transmission delay in the proposed method. The transmission delay of SilentWhispers increases sharply because it relies on a landmark-centered routing while all links include landmarks. The spanning tree in SilentWhispers reflects the local structure of a network. If a short link between the sender and receiver of a transaction is not a part of the spanning tree, a longer link may be used to substitute the short link based on the tree. This may eventually lead to the need for additional channels. SpeedyMurmurs faces the same problem of meaningless link growth and periodically maintaining a spanning tree.

(b) Node count vs. transmission delay. The results of node count vs. transmission delay are shown in Figure 6(b), where the number of channels is 6000, and the node count varies from 1000 to 4000. As the node count increases, the sparsity of the network increases. Hence, the channel count required to transmit a transaction increases slightly with the proposed method. However, the channel count is still smaller than those of its peers. A greater average number of channels required gives a longer transmission delay, which increases slightly with the node count for the proposed method but is still less than those of

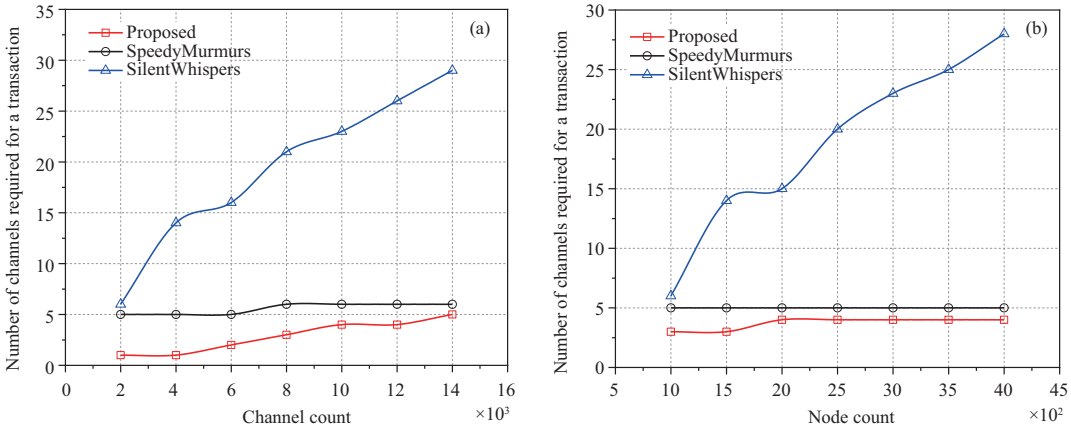


**Figure 6** (Color online) (a) Channel count vs. transmission delay; (b) node count vs. transmission delay.

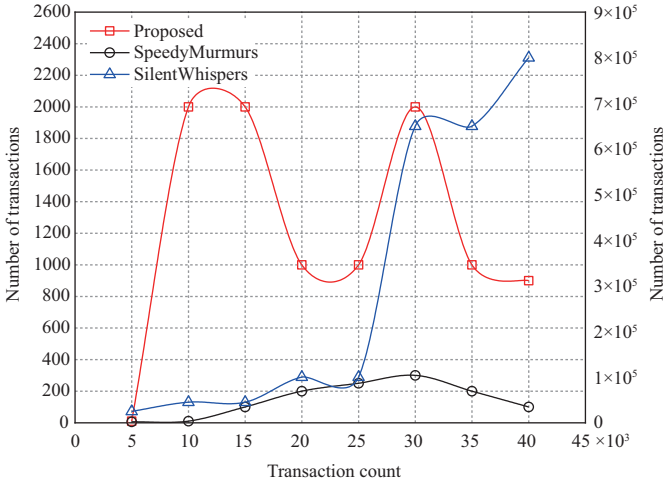


**Figure 7** (Color online) Transaction count vs. transmission overhead.

its two peers. The transmission delay of SilentWhispers increases sharply due to the previously discussed issues.

(5) Transmission overhead. Transaction count affects transmission overhead. A greater average number of transactions may generate more transmission overhead, as shown in Figure 7. The number of channels and nodes are set to 6000 and 3000, respectively, and the transaction count varies from 5000 to 40000. The number of transactions with the proposed method is between those of its peers and is approximately 10 times that of SpeedyMurmurs but nearly 400 times less than that of SilentWhispers. The proposed method aims to improve the transmission success rate and reduce transmission delay when obtaining links. SpeedyMurmurs has the ideal transmission overhead because it uses a periodically maintained spanning tree to quickly find links, which reduces the high overhead caused by finding links in its spanning tree. Although the proposed method is not as fast as SpeedyMurmurs, it is simpler than SilentWhispers, which uses periodic breadth-first traversal to generate trees with more complexity. Therefore, SilentWhispers has the highest transmission overhead.

Based on the above experimental results, as the number of channels and nodes increases, the proposed OCTT reaches the maximum transaction receiving rate. Non-cooperative games allow different transmission rates to converge to the ideal transmission rate, at which the Nash equilibrium solution of OCTT can be used to obtain each channel's Nash equilibrium point. The transmission success rate of OCTT continuously increases with the number of channels and nodes and outperforms both SilentWhispers and SpeedyMurmurs. The transmission delay of OCTT slightly increases with the number of channels and nodes and is shorter than its peers. OCTT has lower transmission overhead than SilentWhispers if transaction count is over 30000. SpeedyMurmurs has the lowest transmission overhead. We summarize the results in Table 2.

**Table 2** Comparison among three methods

| Metric | Method | | |
|---|---|---|---|
| | SpeedyMurmurs | SilentWhispers | OCTT |
| Transmission success rate | Middle | Low | High |
| Transmission delay | Middle | High | Low |
| Transmission overhead ($N_c < 30000$) | Low | Middle | High |
| Transmission overhead ($N_c \geqslant 30000$) | Low | High | Middle |

## 6 Conclusions

In a blockchain channel network, we propose a transaction transmission model based on non-cooperative game theory to optimize transaction transmission performance (transmission success rate and delay) for blockchain channels. The proposed method considers channel balances, channel states, and transmission probability to improve the transmission success rate and decrease delay. An optimized channel transaction transmission algorithm is proposed based on the model, which can guide each channel to find Nash equilibrium points to achieve ideal transmission performance when transmitting transactions. For transactions, ideal transmission links with ideal transmission probabilities can be obtained from equilibrium points. The performance of the proposed method is validated through experiments, which show that the proposed method is superior to two state-of-the-art approaches. Because the length of different transaction paths tends to be different, the storage structure of a two-dimensional array may lead to the problem of excessive storage space requirements. Our future research aims to reduce the proposed method in terms of transmission overhead by using recently presented algorithms, e.g., [44–46]. In addition, the personalized transaction should be studied to analyze its impact on the blockchain channel stability and transmission performance, so as to further improve the transmission transaction success rate and reduce delay.

**Acknowledgements** This work was supported by National Natural Science Foundation of China (Grant No. 61872006), Scientific Research Activities Foundation of Academic and Technical Leaders and Reserve Candidates in Anhui Province (Grant No. 2020H233), Top-notch Discipline (specialty) Talents Foundation in Colleges and Universities of Anhui Province (Grant No. gxbj2020057), Startup Foundation for New Talents of NUIST, and in part by Deanship of Scientific Research (DSR) at King Abdulaziz University (Grant No. RG-30-135-42).

**References**

1 Xiao R Y, Ren W, Zhu T Q, et al. A mixing scheme using a decentralized signature protocol for privacy protection in bitcoin blockchain. IEEE Trans Depend Secure Comput, 2021, 18: 1793–1803
2 Fu X, Wang H M, Shi P C. A survey of Blockchain consensus algorithms: mechanism, design and applications. Sci China Inf Sci, 2021, 64: 121101
3 Mamta, Gupta B B, Li K C, et al. Blockchain-assisted secure fine-grained searchable encryption for a cloud-based healthcare cyber-physical system. IEEE CAA J Autom Sin, 2021, 8: 1877–1890
4 Huang X M, Ye D D, Yu R, et al. Securing parked vehicle assisted fog computing with blockchain and optimal smart contract design. IEEE CAA J Autom Sin, 2020, 7: 426–441
5 Shen M, Liu H S, Zhu L H, et al. Blockchain-assisted secure device authentication for cross-domain industrial IoT. IEEE J Sel Areas Commun, 2020, 38: 942–954
6 Li Y X, Cao B, Peng M G, et al. Direct acyclic graph-based ledger for Internet of Things: performance and security analysis. IEEE ACM Trans Netw, 2020, 28: 1643–1656
7 Zhang P Y, Zhou M C. Security and trust in blockchains: architecture, key technologies, and open issues. IEEE Trans Comput Soc Syst, 2020, 7: 790–801
8 Misra S, Mukherjee A, Roy A, et al. Blockchain at the edge: performance of resource-constrained IoT networks. IEEE Trans Parallel Distrib Syst, 2021, 32: 174–183
9 Tschorsch F, Scheuermann B. Bitcoin and beyond: a technical survey on decentralized digital currencies. IEEE Commun Surv Tut, 2016, 18: 2084–2123
10 Malavolta G, Moreno-Sanchez P, Kate A, et al. Concurrency and privacy with payment-channel networks. In: Proceedings of 2017 ACM SIGSAC Conference on Computer and Communications Security, ACM, 2017. 455-471
11 Xu Y, Ren J, Wang G J, et al. A blockchain-based nonrepudiation network computing service scheme for industrial IoT. IEEE Trans Ind Inf, 2019, 15: 3632–3641
12 Zhang D, Le J Q, Mu N K, et al. An anonymous off-blockchain micropayments scheme for cryptocurrencies in the real world. IEEE Trans Syst Man Cybern Syst, 2020, 50: 32–42
13 Varma S M, Maguluri S T. Throughput optimal routing in blockchain-based payment systems. IEEE Trans Control Netw Syst, 2021, 8: 1859–1868
14 Miller A, Bentov I, Kumaresan R, et al. Sprites and state channels: payment networks that go faster than lightning. In: Proceedings of International Conference on Financial Cryptography and Data Security, 2019. 508–526
15 Zhang Y, Long Y, Liu Z, et al. Z-channel: scalable and efficient scheme in zerocash. In: Proceedings of Computers Security and Communication Networks, 2019. 112-131
16 Dziembowski S, Eckey L, Faust S, et al. Perun: virtual payment hubs over cryptocurrencies. In: Proceedings of IEEE Symposium on Security and Privacy (SP), San Francisco, 2019. 106–123

17 Dziembowski S, Faust S, Hostakova K. Foundations of state channel networks. In: Proceedings of ACM SIGSAC Conference on Computer and Communications Security, 2018. 949–966
18 Dziembowski S, Faust S, Hostáková K. General state channel networks. In: Proceedings of ACM SIGSAC Conference on Computer and Communications Security, Toronto, 2018. 949–966
19 Spilman J. Re: anti DoS for tx replacement. 2020. https://www.mail-archive.com/bitcoin-development@lists.sourceforge.net/msg02028.html
20 Raiden network. 2020. http://raiden.network/
21 Khalil R, Gervais A. Revive: rebalancing off-blockchain payment networks. In: Proceedings of ACM SIGSAC Conference on Computer and Communications Security, Dallas, 2017. 439–453
22 Coleman J, Horne L, Li X J. Counterfactual: generalized state channels. 2020. https://l4.ventures/papers/statechannels.pdf
23 Gabay D, Akkaya K, Cebe M. Privacy-preserving authentication scheme for connected electric vehicles using blockchain and zero knowledge proofs. IEEE Trans Veh Technol, 2020, 69: 5760–5772
24 Sasson E, Chiesa A, Garman C, et al. Zerocash: decentralized anonymous payments from bitcoin. In: Proceedings of IEEE Symposium on Security and Privacy, San Jose, 2014. 459–474
25 Misic J, Misic V B, Chang X, et al. Modeling of bitcoin's blockchain delivery network. IEEE Trans Netw Sci Eng, 2020, 7: 1368–1381
26 Larsson P, Rasmussen L K, Skoglund M. Throughput analysis of hybrid-ARQ-A matrix exponential distribution approach. IEEE Trans Commun, 2016, 64: 416–428
27 Zhang P Y, Li C X, Zhou M C. Game-theoretic modeling and stability analysis of blockchain channels. In: Proceedings of IEEE International Conference on Systems, Man, and Cybernetics, Toronto, 2020. 1–6
28 Yuan H, Fu H Y, Liu J, et al. Non-cooperative game theory based rate adaptation for dynamic video streaming over HTTP. IEEE Trans Mobile Comput, 2018, 17: 2334–2348
29 Liu W, Gu W, Wang J H, et al. Game theoretic non-cooperative distributed coordination control for multi-microgrids. IEEE Trans Smart Grid, 2018, 9: 6986–6997
30 Sun Z M, Liu Y H, Wang J, et al. Non-cooperative game of throughput and hash length for adaptive Merkle tree in mobile wireless networks. IEEE Trans Veh Technol, 2019, 68: 4625–4650
31 Arcak M, Martins N M L C. Dissipativity tools for convergence to nash equilibria in population games. IEEE Trans Control Netw Syst, 2021, 8: 39–50
32 Zhang P Y, Li C X, Zhou M C, et al. A transaction transmission model for blockchain channels. In: Proceedings of IEEE International Conference on Systems, Man, and Cybernetics (SMC), Melbourne, 2021. 1–6
33 Shahsavari Y, Zhang K W, Talhi C. A theoretical model for block propagation analysis in bitcoin network. IEEE Trans Eng Manage, 2020. doi: 10.1109/TEM.2020.2989170
34 Baumgart I, Heep B, Krause S. OverSim: a flexible overlay network simulation framework. In: Proceedings of IEEE Global Internet Symposium, Atlanta, 2007. 79–84
35 Sommer C, German R, Dressler F. Bidirectionally coupled network and road traffic simulation for improved IVC analysis. IEEE Trans Mobile Comput, 2011, 10: 3–15
36 Wang P, Xu H, Jin X, et al. Flash: efficient dynamic routing for offchain networks. In: Proceedings of the 15th International Conference on Emerging Networking Experiments and Technologies (CoNEXT), Orlando, 2019. 370–381
37 Lin S Y, Zhang J J, Wu W G. FSTR: funds skewness aware transaction routing for payment channel networks. In: Proceedings of the 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Valencia, 2020. 464–475
38 Roos S, Moreno-Sanchez P, Kate A, et al. Settling payments fast and private: efficient decentralized routing for path-based transactions. In: Proceedings of Network and Distributed System Symposium (NDSS), San Diego, 2018. 1–15
39 Arami A, van Asseldonk E, van der Kooij H, et al. A clustering-based approach to identify joint impedance during walking. IEEE Trans Neural Syst Rehabil Eng, 2020, 28: 1808–1816
40 Zhong S W, Chen S H, Chang C I, et al. Fusion of spectral-spatial classifiers for hyperspectral image classification. IEEE Trans Geosci Remote Sens, 2021, 59: 5008–5027
41 Malavolta G, Moreno-Sanchez P, Kate A, et al. SilentWhispers: enforcing security and privacy in decentralized credit networks. In: Proceedings of Network and Distributed System Security Symposium, San Diego, 2017. 1–15
42 Mazumdar S, Ruj S, Singh R G, et al. Hushrelay: a privacy-preserving, efficient, and scalable routing algorithm for off-chain payments. In: Proceedings of IEEE International Conference on Blockchain and Cryptocurrency (ICBC), Toronto, 2020. 1–5
43 Khamis J, Schmid S, Rottenstreich O. Demand matrix optimization for offchain payments in blockchain. In: Proceedings of IEEE International Conference on Blockchain and Cryptocurrency (ICBC), Sydney, 2021. 1–9
44 Fu Y P, Zhou M C, Guo X W, et al. Scheduling dual-objective stochastic hybrid flow shop with deteriorating jobs via bi-population evolutionary algorithm. IEEE Trans Syst Man Cybern Syst, 2020, 50: 5037–5048
45 Hai X S, Wang Z L, Feng Q, et al. A novel adaptive pigeon-inspired optimization algorithm based on evolutionary game theory. Sci China Inf Sci, 2021, 64: 139203
46 Wang Y R, Gao S C, Zhou M C, et al. A multi-layered gravitational search algorithm for function optimization and real-world problems. IEEE/CAA J Autom Sin, 2021, 8: 94–109

## Appendix A Table A1

**Table A1** Notations

| Name | Description |
|---|---|
| BTC | Unit of bitcoin |
| OCTT | Optimized channel transaction transmission |
| $A$ | The number of channels required to transmit a transaction |
| $C$ | Channel attribute structure:$C = \langle \text{sID}, \text{tID}, \text{balance}, u_i^t \rangle$, where sID and tID are the IDs of the source and target nodes, balance is the current balance of channel $i$, and $u_i^t$ is the channel steady-state transmission rate |

| | |
|---|---|
| $G$ | $G = \langle V, E \rangle$, where $V$ and $E$ represent a set of nodes and set of channels, respectively $l = \|V\|$ and $K = \|E\|$ |
| $H$ | The number of transmission links |
| $I$ | $I = \{1, 2, \ldots, m\}$, which is the set of participants in non-cooperative games |
| $\mathcal{L}$ | An array used to store links |
| $\|L_k\|$ | The length of the $k$-th link to transmit a transaction (i.e., the number of channels in the link) |
| $N_E$ | Number of messages in a blockchain channel at the end of transmitting a transaction |
| $N_i^s$ | The number of messages successfully transmitted via channel $i$ and received by a target node |
| $N_i^g$ | The total number of messages broadcasted from a source node and transmitted via channel $i$ |
| $N_M$ | Transaction count during time slot $[t, t+1]$ |
| $N_t$ | The total number of transactions after time slot $t$ |
| $N_{t+1}$ | The total number of transactions before time slot $(t+1)$ |
| $P_f^t$ | The probability that the channel is in an unavailable state when channel $i$ transmits a message during period $t$ |
| $P_i^t$ | The probability that a node in the network can receive a transaction via channel $i$ during time slot $t$ |
| $Q$ | Transaction amount |
| $\dot{R}$ | Transaction reception rate |
| $R_i$ | Transmission rate of channel $i$ |
| $S$ | Transmission success rate |
| $S_i^t$ | Probability of a node successfully transmitting a transaction via channel $i$ during time slot $t$ |
| $T$ | The time interval between the end of the current transaction's transmission and the start of the next transaction's transmission via channel $i$ |
| $T_d$ | The time slot to transmit a transaction via a channel in a blockchain network |
| $T_f$ | The time slot of channel $i$ in an unavailable state |
| $T_i^t$ | The total time to transmit transactions via channel $i \in \{1, 2, \ldots, m\}$ in a random time slot $t$ |
| $T_s$ | Duration of channel $i$ in an available state |
| $U_i^t$ | Normalized utility function of channel $i$ during time slot $t$ |
| $X_i^t$ | Average number of transactions successfully transmitted from a node via channel $i$ during a unit time slot |
| $\Phi_i^t$ | Transmission time of transactions via channel $i$ during time slot $t$ |
| $i$ | Blockchain channel $i$ and $i \in I$ |
| $k_i$ | The correlation coefficient of node $i$'s channel transmission probability |
| $m$ | The number of channels with sufficient balances that make a link between a source node and a target one |
| $t$ | A random period |
| $u_i^t$ | The channel steady-state transmission rate |
| $u_i^{t+1}$ | The iterative update of channel steady-state transmission rate of channel $i$ during time slot $t+1$ |
| $\alpha_i^t$ | The transmission probability of channel $i$ when $T \leqslant T_i^t$ |
| $\alpha_{-i}^t$ | The strategy space of channels except for channel $i$ and $\alpha_{-i}^t = \{\alpha_1^t, \ldots, \alpha_{i-1}^t, \alpha_{i+1}^t, \ldots, \alpha_m^t\}$ |
| $\gamma_i^t$ | Nash equilibrium solution of channel $i$ |
| $\gamma_{-i}^t$ | Nash equilibrium solution including $m-1$ channels except for channel $i$ |
| $(\gamma_i^t, \gamma_{-i}^t)$ | Nash equilibrium point of non-cooperative games |
| $\alpha_i^*$ | Optimal solution of model $\Theta$ |
| $\mu_i^r$ | The Lagrange multiplier of $r$-th iteration of channel $i$ |
| $\nu_i$ | The Lagrange multiplier of channel $i$ related to constraints |
| $\sigma_\mu^t$ | Iterate the step size of sub-gradient method and $\sigma_\mu^t = \varepsilon/t$ |
| $\rho_i^t$ | The indicator of the deviation of nodes' channel transmission probability between the current and ideal transmission probability of channel $i$ during time slot $t$ |
| $\varepsilon$ | Upper bound of step size $\rho_\mu^t$ and $\varepsilon > 0$ |
| $\phi$ | Threshold in Algorithm 2 |
| $r$ | Iteration times |
| $\pi_0, \pi_1$ | Probability that channel $i$ is in an unavailable state and all nodes successfully transmitting transactions during time slot $t$, respectively |
| $\chi$ | Strategy space of non-cooperative games |
| $\mathcal{U}$ | Profit set of non-cooperative games |
| $\Theta$ | Transmission model |
| $\omega_1, \omega_2$ | The weighting factors in Eq. (12) |