Routing Optimization for High Speed Photon State-Channel Architecture

Xiaowei Ding[1*], Litai Ren[2], Zizhou Sang[3], Zijie Zhang[4], Yifan Du[4], and Peter Yan[5*]

[1] School of Information Management, Nanjing University, 163 Xianlin Road, 210023, China
[2] School of Chemistry, Nanjing University, 163 Xianlin Road, 210023, China
[3] School of Engineering Management, Nanjing University, 163 Xianlin Road, 210023, China
[4] School of Mathematics, Nanjing University, 163 Xianlin Road, 210023, China
[5] SmartMesh Foundation Pte. Ltd. and MeshBox Foundation Pte. Ltd.
dingxiaowei@nju.edu.cn, peter@smartmesh.io

Abstract. Compared with mainstream payment systems such as Visa, the biggest obstacles to blockchain based technologies such as Bitcoin and Ethereum becoming mainstream means of payment in human daily life lie in their low transaction rate and slow response time. A potentially promising solution is the state channel architecture. State channels are more general than payment channels, which provide off-chain transaction settlement without much need for expensive on-chain operations. We investigate the routing optimization problem for Photon, a state channel network for Spectrum, which is similar to Raiden being a state channel network on top of Ethereum. Yet Photon possesses interesting characteristics that Raiden lacks. Extensive simulations show our proposed algorithm can effectively achieve high success rate and throughput for Photon with low deposit lockup.

Keywords: Blockchain, Bitcoin, Ethereum, Lightning Network, Raiden Network, Spectrum, Photon Network, Payment Channel Network, State Channel Network

1 Introduction

Blockchain networks, such as Bitcoin and Ethereum, are notorious for having low transaction rate and slow response time. Bitcoin can process no more than 7 transactions per second [1] and it takes tens of minutes to confirm a transaction. Even if the parameters of blockchain (such as block size and block interval) are optimized, it is still difficult to exceed 100 transactions per second [2]. In contrast, mature payment systems such as Visa can process thousands of transactions per second with only a few seconds' latency.

Many contributions tackle the above-mentioned challenges from various angles, such as modifying consensus mechanisms, sharding, sidechains, and state(or payment) channels. While these methods may potentially expand the overall transaction capability, only the state channel networks, such as the Lightning Network and Raiden Network, also simultaneously improve on the cost and latency fronts [3-13]. The state

[*] Both Xiaowei Ding and Peter Yan are corresponding authors.

channel network technology aims to establish bilateral "channels" between nodes of the blockchain network, and the main blockchain is then only used to setup channels, close channels and handle abnormal situations. Between two nodes with a channel established, two-way payments can be performed at high speed, and the bilateral accounts are only updated off-chain, within the channel. Thus, state channels improve transaction throughput because most transactions do not need to be written onto the blockchain. State channels can be further interconnected to form networks, which can be used to transfer funds between nodes with no direct connections.

In July 2018, it was reported that Lightning Network launched an experiment involving more than 100 merchants [14]. High-quality routing in a state channel network is technically challenging, especially when facing dynamically changing unstructured network topologies. The challenge is even more prominent when the scale of the network reaches the level of everyday human transactions [15]. The path-finding optimization problem in state channel networks bears certain similarities to that in traditional communication networks. However, since blockchains transfer value, the challenges are not only technical, but also of profound economic and financial connotations.

We study the routing optimization problem for the Photon Network. Photon is the same as Raiden with the following differences:

- Photon routing will be optimized with Transactive Real-time AI Negotiator (TRAIN), which can make the Transfer-Matrix to be more smooth and predictable, and therefore improves the performance of Photon.
- We have MeshBoxes [17] acting as Photon Infrastructure Nodes which can create channels with deposits. Thus, for Photon, we have control in terms of deposits, channels and network topology. For others, they don't have such control since they use public nodes on the Internet which are controlled by normal users.

Against this background, as the first step in a series of research undertakings, we analyze the routing performance vs. deposit requirements for Photon. Our research contributes to the understanding of the tradeoffs in various design choices for the Photon State Channel Architecture, and hopefully it’s also helpful to the community at large.

The remainder of the paper is structured as follows. In Section 2 we introduce the background. In Section 3 we overview related works. In Section 4 we present our approach. In Section 5 we present evaluation. We conclude with Section 6.

2 Background

The state channel allows a peer-to-peer direct payment highway to be established between the two parties in a transaction. When establishing a payment channel, both parties are required to submit certain deposits. Both parties maintain the private bilateral ledger without having to write all transactions to the blockchain. At the same time, the channel guarantees that the parties will not have bad behavior, or the misbehavior will be punished. At the end, if off-chain transactions are no longer needed, or if one party proposes to close the state channel, or if the available balance is exhausted, the channel is closed and settled and the final state is recorded back onto the blockchain. If arbitration is needed for misbehavior, the blockchain will also be resorted to. One of the advantages of the state channel is that it can avoid on-chain transactions, and the throughput is limited only by the bandwidth between the two parties. Another advantage is that no miner service is required, so transaction costs can be saved.

In a state channel network, the path-finding algorithm must guarantee atomicity. If for an intermediate node, no outgoing channel adjacent to it has sufficient balance to service the fund transfer, then the path-finding failure at this node will cause the entire path to fail automatically. The path-finding algorithm can then explore other paths. The balance or capacity of the channel on a path is either updated simultaneously or not updated at all. There can be no partial updates because partial updates can result in loss of funds. The Lightning Network uses Hash Time-Lock Contract (HTLC) to solve this problem. In a state channel network, the path-finding algorithm must achieve the highest possible throughput and success rate using as little capital locked in the network as possible. In addition, it is also necessary to provide certain economic incentives to users, especially service providers and intermediate nodes that provide routing services. Unlike traditional network routing algorithms, state channel network routing algorithms focus on finding a path with sufficient available balance to support the funds to be transferred, rather than the traditional shortest path. In addition, routing algorithm is faced with constantly changing available balances on channels. Last but not the least, security and privacy protection are extremely important.

Spectrum is a Proof-of-Capability based public chain. In Spectrum, nodes that contribute more resources to the system will get more capability scores, giving them more opportunities to do the bookkeeping on the blockchain [16]. Important members of the Spectrum network are the Meshbox (indoor) and MeshBox++ (outdoor) wifi routers [17]. When a MeshBox (++) owner contributes bandwidth and disk space to the system, the node's capability score also increases.

3 Related Work

Research on state(payment) channel networks focuses on multiple dimensions such as performance, security and privacy.

Flare [6] is a beacon-based hybrid source-based routing scheme in Lightning Network, which borrowed ideas from the routing of mobile ad hoc networks. Due to the design decisions such as the adoption of beacons, this routing algorithm is probabilistic, that is, there is no guarantee that a path will be found between the sender and the receiver. This is acceptable for Lightning Network because when difficulties in path-finding arise, the network architecture allows a new channel to be opened to increase the connectivity of the network and also allows for funds to be transferred thru the main blockchain. Such a retry or fallback mechanism can also be integrated into the Lightning Network clients. The authors estimate that Flare can support millions of users.

In [19], the authors argued that single path routing schemes have a number of problems, with the most severe problem being the low utilization of available capabilities. The authors modelled payment channel networks as flow networks. They leveraged flow network algorithms to effectively utilize available capacities by aggregating multiple paths. They proposed an extended push-relabel algorithm and demonstrated that their algorithm could meet the demands when single-path based approaches fail.

SpeedyMurmurs [13] is an embedding-based or distance-based routing algorithm. It provides formal privacy guarantee in fully distributed environment and reduces the overhead of routing a transaction by more than a factor of two.

SilentWhispers [21] is a landmark-based routing algorithm. It achieves a number of privacy properties (sender, receiver, link, and value privacy). The use of highly linked Ripple-like gateway nodes is vital to make the system efficient, robust, and scalable.

Spider [18] is a packet-switched payment channel network: it breaks down the payment amount into transmission units and transmits them over multiple paths, using congestion control and in-network scheduling mechanisms. Spider features imbalance-aware routing. The authors derived decentralized algorithms for solving optimization problems taking into account the rate-imbalance constraints. In performance evaluation, for the same amount of funds locked in the network, Spider completes 10-45% more transactions than SpeedyMurmurs and SilentWhispers. In an ISP-like topology, Spider performs 5-15% more transactions than the traditional max-flow algorithm.

Revive [20] focuses on rebalancing. When the available balance of a channel is exhausted, such as in the case of a large number of asymmetric transactions, it is necessary to close the channel first, and then re-open and refund the channel. This is a very expensive operation. Revive proposed to refund the depleting channels thru rebalancing.

There are other related contributions. Due to limitation in space, our list of related works might be incomplete, for which we sincerely apologize.

4 Our Approach

Currently existing solutions assume that the network is defined by users in terms of nodes, channels, and deposits. For these existing solutions, they are given the network topology (connectivity) and deposits, and they try to route as best as they can. Usually deposits will be small, the amount which can be transferred is limited, and failure rates can be high, because users must put their own personal deposit onto channels, and other users will use up this deposit. Unless users are significantly incentivized, there is little reason they will lock up their own funds into channels which are used by others.

These is a major difference in our scenario, which is a niche. Since we have MeshBoxes, and invest significant deposits which can be several orders of magnitude more than in the above-mentioned user defined channel situation into the Infrastructure Channels (between MeshBoxes), we have control over the Infrastructure Nodes, Channels and the network topology, and deposits that are put onto the Infrastructure Channels. This should allow us to achieve a better solution than for the arbitrary networks.

In our architecture, user(initiator) nodes only make a channel to a core node, and a core node must make a channel to each user which can be a target. For networks with “super nodes”, such super nodes are similar to our core nodes. However, there is still a big difference between a few "super nodes" being instantiated, and our entire core networks, which is a regular topology of Meshbox Core Nodes with high deposit channels.

Our goal is to find the best rebalancing routing algorithm, but specifically for our Infrastructure network and pre-defined topology, and high-deposits. These are unique assumptions we have because we have control over Meshboxes. We should be able to reach much higher Transaction Per Second (TPS) and much higher success rate than others, given a certain distribution for the amount to be transferred (e.g. Uniform[1,100]) and given various symmetric and asymmetric Transfer Matrices(TM). In Fig. 1, on the left, we show the Photon Architecture, on the right, we show how funds transfer from User node a1 to User node h3. In this example, each link between a pair of nodes represents two uni-directional State-Channels: e.g. a channel from A to B (with Available Balance 100), and channel from B to A (with Available Balance 200). Fig. 2 shows that a channel's low available balance might be the cause of "failure".

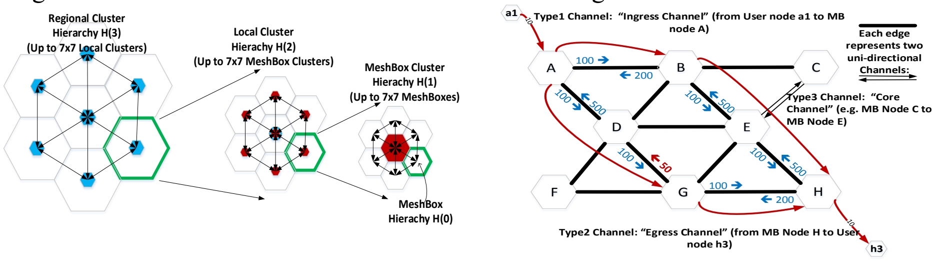
Regional Cluster Hierarchy H(3) (Up to 7x7 Local Clusters)
Local Cluster Hierachy H(2) (Up to 7x7 MeshBox Clusters)
MeshBox Cluster Hierachy H(1) (Up to 7x7 MeshBoxes)
MeshBox Hierachy H(0)
a1
10
Type1 Channel: "Ingress Channel" (from User node a1 to MB node A)
Each edge represents two uni-directional Channels:
A
B
C
D
E
F
G
H
100
200
500
50
Type3 Channel: "Core Channel" (e.g. MB Node C to MB Node E)
Type2 Channel: "Egress Channel" (from MB Node H to User node h3)
h3

Fig. 1. Left: three levels of "fractal" hierarchies for the Photon Networks. **Right:** funds transfer thru Core Channels (Nodes), or Infrastructure Channels (Nodes).

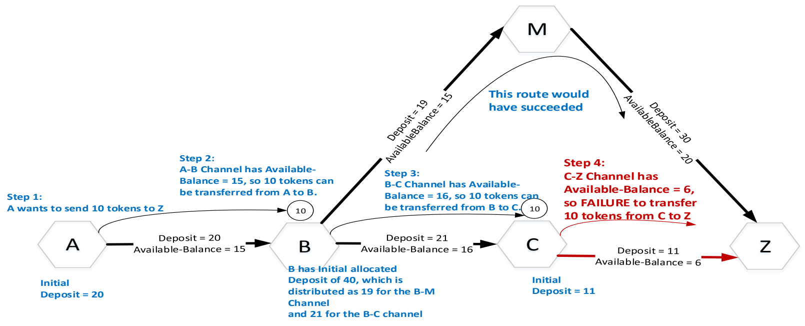
M
This route would have succeeded
Deposit = 19
AvailableBalance = 15
Deposit = 30
AvailableBalance = 20
Step 1:
A wants to send 10 tokens to Z
Step 2:
A-B Channel has Available-Balance = 15, so 10 tokens can be transferred from A to B.
Step 3:
B-C Channel has Available-Balance = 16, so 10 tokens can be transferred from B to C.
Step 4:
C-Z Channel has Available-Balance = 6, so FAILURE to transfer 10 tokens from C to Z
10
A
B
C
Z
Deposit = 20
Available-Balance = 15
Deposit = 21
Available-Balance = 16
Deposit = 11
Available-Balance = 6
Initial Deposit = 20
B has Initial allocated Deposit of 40, which is distributed as 19 for the B-M Channel and 21 for the B-C channel
Initial Deposit = 11

Fig. 2. The notion of "channel failure" due to insufficient available balance on channel.

4.1 Rebalancing Algorithms

Let us define:

$D_{i,j}$: (initial) deposit on channel $C_{i,j}$.

$A_{i,j}$: available balance on channel $C_{i,j}$.

$Y_{i,j}$: locked funds on channel $C_{i,j}$ during transaction.

$X_{i,j}$: total amount of successfully transmitted funds on channel $C_{i,j}$.

$D^{\notin}_{i,j} = A_{i,j} - A_{j,i}$: difference between forward channel and reverse channel.

We consider three types of rebalancing algorithms, Algorithm 1 being the foundational.

Algorithm 1 Conjugate Rebalancing

```
Input: Sender,Receiver,Payment
Output: Updated state matrix $A, Y, X$
1: Parameter: n: number of top shortest routes used, default n=2.
2: i:=Sender; isSuccess:=true;
3: while i not equal to Receiver do
4:     Call Dijkstra on (i,Receiver) and return top n shortest routes
5:     for j:=1 to n do
6:         Let k be i's outgoing edge i→k on the j-th shortest route
7:         if $A_{i,k}$ > Payment then
8:             $Y_{i,k} = Y_{i,k}$ + Payment
9:             $A_{i,k} = D_{i,k}$ -$X_{i,k}$ + $X_{k,i}$ - $Y_{i,k}$
10:            Declear "Channel Success"; i:=k;
11:            break
12:        end if
13:    end for
14:    Declear "Path Failure"
15:    # Reset A along its path, releasing the locked funds, all the way back to this transaction's Sender.
16:    for $i_k$ along the path, e.g.$i_0$(=Sender),$i_1$,$i_2$,...,$i_n$(=i) do
17:        $Y_{i_{k-1},i_k} = Y_{i_{k-1},i_k}$ - Payment
18:        $A_{i_{k-1},i_k} = D_{i_{k-1},i_k}$ - $X_{i_{k-1},i_k}$ + $X_{i_k,i_{k-1}}$ - $Y_{i_{k-1},i_k}$
19:    end for
20:    isSuccess:=false
21:    break
22: end while
23: Declear "Path Success"
24: if isSuccess then
25:    # Make updates, releasing the locked funds, along the whole path.
26:    for $i_k$ along the path, e.g.$i_0$(=Sender),$i_1$,$i_2$,...,$i_n$(=Receiver) do
27:        $X_{i_{k-1},i_k} = X_{i_{k-1},i_k}$ + Payment
28:        $Y_{i_{k-1},i_k} = Y_{i_{k-1},i_k}$ - Payment
29:        $A_{i_{k-1},i_k} = D_{i_{k-1},i_k}$ - $X_{i_{k-1},i_k}$ + $X_{i_k,i_{k-1}}$ - $Y_{i_{k-1},i_k}$
30:        $A_{i_k,i_{k-1}} = D_{i_k,i_{k-1}}$ - $X_{i_k,i_{k-1}}$ + $X_{i_{k-1},i_k}$ - $Y_{i_k,i_{k-1}}$
31:    end for
32: end if
```

The heuristics behind Algorithm 2 is that for an edge, if the available balances of the forward and reverse channels differ too much, then we artificially create a series of triangle-looped transactions to rebalance the channels.

Algorithm 2 Circle Rebalancing

```
1: Parameter: $\theta_1$ : threshold1, determines which level of $D^f_{i,j}$ will lead into circle rebalancing, default $\theta_1$=0
2:            $\theta_2$ : threshold2, determines one time's circle rebalancing's least adjusted A, default $\theta_2$=50
3: Insert the following pseudo-code to between line 10 and 11 in Algorithm 1:
4: ...
5: if $D^f_{i,k} > \theta_1$ then
6:     Delete $C_{i,k}$ from map, then call Dijkstra on (k,i) and return [k,q,i]
7:     $F_s$ := sum($A_{k,q}$,$A_{q,i}$,$A_{i,k}$) # $F_s$ : sum of A of forward channels within rebalancing circle
8:     $R_s$ := sum($A_{q,k}$,$A_{i,q}$,$A_{k,i}$) # $R_s$ : sum of A of reverse channels within rebalancing circle
9:     $\Delta := \frac{F_s+R_s}{6}$
10:    if $\Delta > \theta_2$ then
11:        $A_{k,q}$-=$\Delta$ , $A_{q,i}$-=$\Delta$ , $A_{i,k}$-=$\Delta$
12:        $A_{q,k}$+=$\Delta$ , $A_{i,q}$+=$\Delta$ , $A_{k,i}$+=$\Delta$
13:    end if
14: end if
15: ...
```

The heuristics behind Algorithm 3 is that at a node n, we would like to find the next node x, such that we can maximize $(A_{n,x} - A_{x,\mathrm{n}})$.

Algorithm 3 Heuristic Routing Rebalancing

```
1: Parameter: $N_A$:the number of candidate paths (from which to choose the best path with max $D^f_{i,k}$),
2:            default $N_A$=2
3:            $T_L$: limit on the max number of hops for a path, default $T_L$=10
4: Replace line 4 in Algorithm 1 with the following pseudo-code:
5: ...
6: for i=1 to n do
7:     if current hop count < $T_L$ then
8:         for j = 1 to $N_A$ do
9:             Call Dijkstra on (Sender,Receiver) and return the shortest path and the next node k
10:            if if at least one shortest path is returned then
11:                if $D^f_{i,k} > D^f_{max}$ then
12:                    Take this shortest path as the top i-th path
13:                    $D^f_{max} := D^f_{i,k}$
14:                end if
15:            else
16:                Temporarily mask $C_{i,k}$ away from the map
17:            end if
18:        end for
19:    end if
20: end for
21: ...
```

4.2 Utility Optimization

Let us define:

$\mathcal{T}$: Transaction Initiation Rate (TIR), the number of initiated transactions per second in the system.

$\tilde{\mathcal{T}}$: Transaction per Second(TPS), the number of successful transactions per second in the system.

P: Success Rate(SR), the number of successful transactions divided by total number of transactions in the system.

$\mathcal{T}^f := \mathcal{T} - \tilde{\mathcal{T}}$: failed TPS in the Photon system.

$D_S := \sum_{i \neq j} D_{i,j}$: sum of deposits.

Our utility function then can be defined as:

$$U\,(D, \mathcal{T}) = \frac{w_{Ptps} \cdot \tilde{\mathcal{T}} - w_{Pftps} \cdot \mathcal{T}^f}{D_S} = \frac{\mathcal{T}}{D_S} \cdot \left[-w_{Pftps} + \mathrm{P} \cdot \left(w_{Ptps} + w_{Pftps}\right)\right], \tag{1}$$

where the w's represent the weights on the Quality of Experience(QoE) of successful transactions and failed transactions, respectively. The w's are usually set to be constants.

With increasing number of failures per second, a human-being's QoE degrades, therefore, w_{Pftps} could be a function of $\mathcal{T}^f$. In the simplest case, w_{Pftps} could be set to be a linear function of $\mathcal{T}^f$: $w_{Pftps} = a + b \cdot \mathcal{T}^f = a + b \cdot \mathcal{T} \cdot (1 - P)$.

Then the utility optimization problem can be formulated as:

$$U(D, \mathcal{T}) = \frac{\mathcal{T}}{D_S} \cdot \left\{P \cdot w_{Ptps} + (P - 1) \cdot [a + b \cdot \mathcal{T} \cdot (1 - P)]\right\}, \tag{2}$$

$$\min U(D, \mathcal{T})\,, s.t. \begin{cases} \mathcal{T} > 0 \\ D_S = \sum_{i \neq j} D_{i,j} \\ \forall i, j, D_{i,j} = D(> 0) \end{cases}. \tag{3}$$

4.3 Simulation Setup

To understand the properties of the Photon system, we focused mainly on the 7-node setup. We adopted the Poisson process to model the Transaction Initiations in the system, with the parameter to the Poisson process being the TIR. Each element of the 3-tuple (Sender, Receiver, Payment) was drawn from uniform distribution. Given the uniform distribution(symmetrical Transfer Matrix), in the long run(steady state), assuming the rebalancing algorithms do well, it should be most effective for all the D's to be the same, as that is the long run equilibrium. Therefore we fix the elements of the D matrix to be one single value, which is the long run equilibrium value. In the follow-up studies, we will study the case of asymmetrical Transfer Matrix and non-uniform (Sender, Receiver, Payment) distributions.

In our study, we set $n = 2$ for Algorithm 1, $\theta_1 = 0, \theta_2 = 50$ for Algorithm 2, $N_A = 2, T_L = 10$ for Algorithm 3. And we experimented with various combinations of (w_{Ptps}, a, b) for the utility function in the optimization problem and set $w_{Ptps} = 1, a = 10, b = 10$ for the plots. For each hop, we set the delay to be 0.002 seconds.

5 Evaluation and Discussion

In Fig.3, the left panel shows the Success Rate vs. D given $\mathcal{T} = 20000$. It is seen that Algorithm 1 can achieve the highest Success Rate for relative low Deposit value. At TIR=20000, even for Deposits that are below 500, the system can still maintain above 90% Success Rate. The right panel shows that we can still reach >90% Success Rate at 25000 TIR. It is expected that Algorithm 1 achieves the highest performance in comparison. For symmetrical Transfer Matrix and uniformly distributed transactions, in the long run equilibrium, this fact is in line with our intuition. This might not hold when we are faced with asymmetrical Transfer Matrix with non-uniformly distributed transactions, which we will study further in the future. In Fig. 4 and Fig. 5, we show the Utility Function(notice the negated vertical axis). It is seen that the lower the deposit, the lower the Utility. The higher the TIR, the lower the Utility. It is intuitive to conjecture that the very low TIR will also lower the Utility value and therefore, there might be an optimal TIR point which maximizes the Utility Function. This will be one of our future works.

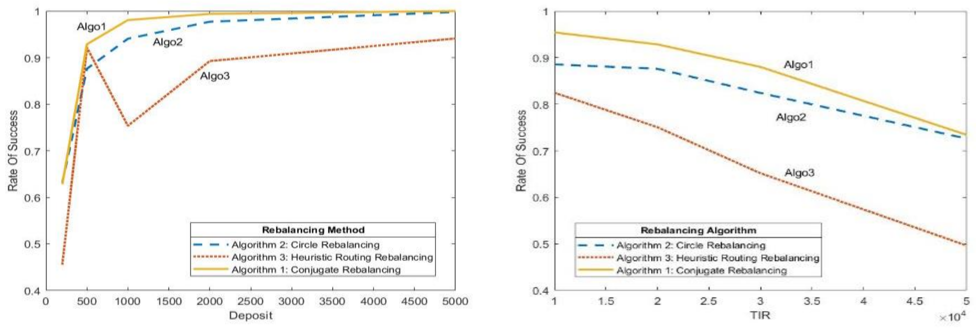

Fig. 3. The Rate of Successes. **Left:** Fix $\mathcal{T} = 20000$, $D \in [200{,}5000]$. **Right:** Fix $D = 500$, $\mathcal{T} \in [10000, 50000]$.

6 Conclusion

In this work, we investigate the routing optimization problem for Photon. Extensive simulations show that our proposed algorithm can effectively achieve high success rate and throughput of fund-transferring for Photon systems with low deposit lockup.

In the future, we will further deepen our understanding of the system and improve our algorithms. For instance, we will study the case of asymmetrical Transfer Matrix and non-uniform (Sender, Receiver, Payment) distributions. Moreover, to model a more realistic system, Deep Learning can be used with the following considerations: several parameters will change as a function of time, and could be correlated with calendar days (day of the month, day of the week, holidays), time-of-day, local or regional news events, stock market, weather, etc. With enough (big) data, it may be possible to learn how such parameters will vary in advance and configure the parameters (e.g. deposits) proactively in order to maximize the utility function.

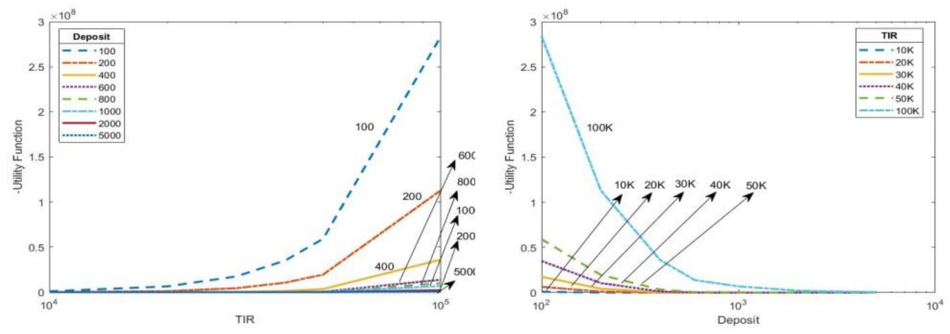

Fig. 4. Utility Function. **Left:** Fix D =100,200,400, 600, 800,1000,2000,5000, $\mathcal{T}$ ∈[10000,100000]. **Right:** Fix $\mathcal{T}$ =10000,20000,30000, 40000,50000,100000， $D \in [0,5000]$.

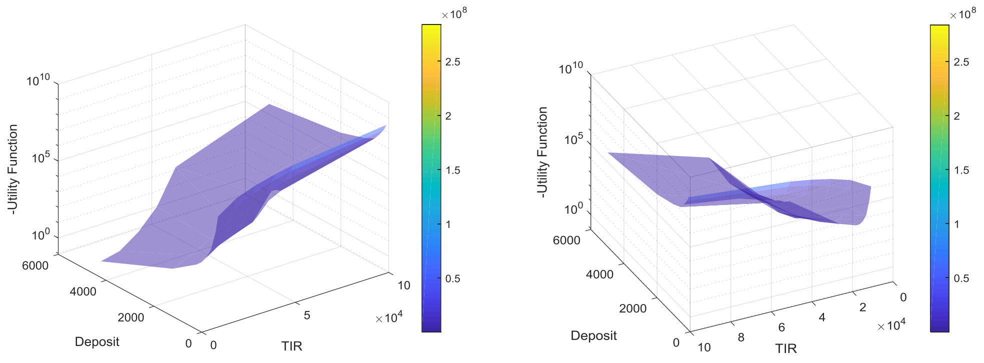

Fig. 5. Utility Function(3D plots, two different views), $D \in [100,5000]$， $\mathcal{T} \in [10000,100000]$.

7 Acknowledgements

We greatly appreciate the help from SmartMesh Foundation and MeshBox Foundation.

References

1. Croman, K., Decker, C., Eyal, I., Gencer, A. E., Juels, A., Kosba, A., ... & Song, D. (2016, February). On scaling decentralized blockchains. In International Conference on Financial Cryptography and Data Security (pp. 106-125). Springer, Berlin, Heidelberg.
2. Gervais, A., Karame, G. O., Wüst, K., Glykantzis, V., Ritzdorf, H., & Capkun, S. (2016, October). On the security and performance of proof of work blockchains. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (pp. 3-16). ACM.
3. Poon, J., & Dryja, T. (2016). The bitcoin lightning network: Scalable off-chain instant payments. https://lightning. network/lightning-network-paper. pdf.

4. Miller, A., Bentov, I., Kumaresan, R., & McCorry, P. (2017). Sprites: Payment channels that go faster than lightning. *CoRR abs/1702.05812*.
5. Raiden network. http://raiden.network/.
6. Prihodko, P., Zhigulin, S., Sahno, M., Ostrovskiy, A., & Osuntokun, O. (2016). Flare: An approach to routing in lightning network. *White Paper*.
7. Decker, C., & Wattenhofer, R. (2015, August). A fast and scalable payment network with bitcoin duplex micropayment channels. In *Symposium on Self-Stabilizing Systems* (pp. 3-18). Springer, Cham.
8. McCorry, P., Möser, M., Shahandasti, S. F., & Hao, F. (2016, July). Towards bitcoin payment networks. In *Australasian Conference on Information Security and Privacy* (pp. 57-76). Springer, Cham.
9. Malavolta, G., Moreno-Sanchez, P., Kate, A., Maffei, M., & Ravi, S. (2017, October). Concurrency and privacy with payment-channel networks. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (pp. 455-471). ACM.
10. Heilman, E., Alshenibr, L., Baldimtsi, F., Scafuro, A., & Goldberg, S. (2017). TumbleBit: An untrusted Bitcoin-compatible anonymous payment hub. In *Network and Distributed System Security Symposium*.
11. Lind, J., Eyal, I., Pietzuch, P., & Sirer, E. G. (2016). Teechan: Payment channels using trusted execution environments. *arXiv preprint arXiv:1612.07766*.
12. Green, M., & Miers, I. (2017, October). Bolt: Anonymous payment channels for decentralized currencies. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (pp. 473-489). ACM.
13. Roos, S., Moreno-Sanchez, P., Kate, A., & Goldberg, I. (2017). Settling Payments Fast and Private: Efficient Decentralized Routing for Path-Based Transactions. *arXiv preprint arXiv:1709.05748*.
14. 100 merchants can now trial Bitcoin's lightning network risk free. https://finance.yahoo.com/news/100-merchants-now-trial-bitcoins-120126391.html. Last accessed: 2018/10/07.
15. Engelmann, F., Kopp, H., Kargl, F., Glaser, F., & Weinhardt, C. (2017, December). Towards an economic analysis of routing in payment channel networks. In *Proceedings of the 1st Workshop on Scalable and Resilient Infrastructures for Distributed Ledgers* (p. 2). ACM.
16. Spectrum Foundation (2018, July). Parallel Internet Value Transfer Protocol and DAPP Platform. *White Paper*.
17. MeshBox Foundation(2018, Jan). MeshBox will create a new distributed routing/storage device standard. *White Paper*.
18. Sivaraman, V., Venkatakrishnan, S. B., Alizadeh, M., Fanti, G., & Viswanath, P. (2018). Routing Cryptocurrency with the Spider Network. *arXiv preprint arXiv:1809.05088*.
19. Rohrer, E., Laß, J. F., & Tschorsch, F. (2017). Towards a Concurrent and Distributed Route Selection for Payment Channel Networks. In *Data Privacy Management, Cryptocurrencies and Blockchain Technology* (pp. 411-419). Springer, Cham.
20. Khalil, R., & Gervais, A. (2017, October). Revive: Rebalancing off-blockchain payment networks. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (pp. 439-453). ACM.
21. Malavolta, G., Moreno-Sanchez, P., Kate, A., & Maffei, M. (2016). SilentWhispers: Enforcing Security and Privacy in Decentralized Credit Networks. *IACR Cryptology ePrint Archive*, *2016*, 1054.