Improving Transaction Success Rate in Cryptocurrency Payment Channel Networks

Suat Mercan[a], Enes Erdin[b], Kemal Akkaya[a]

[a]Department of Electrical and Computer Engineering, Florida International University, Miami, FL 33174
[b]Department of Computer Science, University of Central Arkansas, Conway, AR 72035

Abstract

Blockchain-based cryptocurrencies has received a substantial interest in the last decade as Blockchain can ensure trust among users without relying on third parties. However, cryptocurrency adoption for micro-payments has been limited due to slow conrmation of transactions and unforeseeable high fees, especially in the case of Bitcoin. To this end, creating o-chain payment channels between users is proposed which enables instant and nearly free transactions without writing to blockchain. O-chain channel idea is then extended to establish pay-ment channel networks to scale the idea to allow payment routing among many users. However, due to the way these channels are designed, both sides of a channel have a xed one-way capacity for making transactions. Consequently, if one side consumes the whole one-way capacity, the channel becomes non-transitive in that particular direction, which causes failures of payments that would like to pass through. Eventually, the network becomes partitioned with unevenly distributed funds. In this paper, we propose the adoption of three specic techniques that aim to increase the overall success rate of payments and address channel imbalance problem to keep the payment channel network sustainable in the long run. First, we show the eectiveness of balance-aware routing that better utilizes available funds in the channels. Second, we propose an ecient method for selection of the gateway (i.e., connection point) for a user by considering the gateway's inbound and outbound capacity. It exploits the fact that end-users can connect the network through multiple gateways any of which can be used to initiate the payment. Finally, we propose proportional payment splitting method to further increase success rate especially for large transactions. We implemented the three approaches for assessing their eec-tiveness. Compared to existing approaches such as maximum ow or greedy, the proposed approaches can achieve much higher success rates with channels balanced better.

Keywords: Bitcoin, Cryptocurrency, Blockchain, Payment Channel Network,

---

Email address: smercan@fiu.edu (Suat Mercan)

A poster version of this paper appeared in the proceedings of IEEE International Conference on Blockchain and Cryptocurrency (ICBC), Toronto, Canada, May 2020

Routing, Lightning Network

1. Introduction

Cryptocurrencies which ensure secure transactions using cryptographic techniques are one of the most inuential technologies in the last decade [1, 2, 3]. The intriguing idea behind cryptocurrencies is storing the ownership of the entities in an append-only, tamper-proof database which is known as Blockchain. Consensus mechanism (i.e., Proof of Work (PoW)) used in Blockchains eliminates the necessity of a central authority to approve and keep the records. This enables making transactions in a trustless environment. The system is resilient against node failures and malicious acts thanks to its highly distributed structure. Therefore, Blockchain solves the single point of failure problem and removes dependency on the trusted third parties for transactions. Although Blockchain is being proposed to be used in various applications [4], Bitcoin cryptocurrency is the rst practical and widely accepted use case. There are also other cryptocurrencies that depend on Blockchain but their market cap is much less [5].

Even though the concept of a virtual currency is a brilliant idea with a promising architecture, it still suers from lack of wide adoption [6, 7] due to its impracticality in day-to-day micro-payments stemming from two main problems: 1) high transaction fees and; 2) long block conrmation times. For instance, in Bitcoin it takes 10 minutes to approve a block of transactions. Furthermore, to prevent double spending, as a rule of thumb, the merchants wait for approval of 6 consecutive blocks. Additionally, the size of a block is limited to 4 MB which not only hinders the possibility of an approval of a payment on a congested day but also limits the total number of transactions in a unit time. The theoretical maximum in Bitcoin is calculated to be 7 transactions per second [8] which is far lower than what Visa or MasterCard can process [9]. The energy spent by the miners is another factor in the valuation of the transaction fees [10].

There have been some recent attempts to address these issues. Payment channel network (also known as o-chain transaction networks) concept [11, 12] is among the proposed solutions. The idea is to utilize smart-contracts and avoid writing every transaction on the Blockchain. Instead, the transactions are recorded o-chain until the accounts are reconciled. Specically, once a channel is created between two peers, many transactions can be performed in both directions as long as there is enough funds. When many nodes come together, the o-chain transaction channels turn into a network of payment channels. Instead of opening a direct channel, a peer makes use of an already established channel to forward money over existing nodes by paying a transaction fee as long as a path exists from the payer to the payee. This also helps in reducing channel opening costs for a user since existing channels can be utilized. Lightning Network (LN) is a perfect example of this concept that has reached to almost 10K users in 2 years [11] for Bitcoin transactions. This payment network consists of nodes which charge transaction fees to other users for passing their data. Basically, a user connects to a gateway node, which is further connected to routing nodes that act as the network backbone. Ultimately, LN achieves almost real-time transactions with negligible fees compared to Bitcoin fees and transaction validation times.

Nevertheless, payment channel networks (PCNs), including LN, come with their own challenges due to the way the channel capacities are consumed. Basically, if there is not enough available capacity on a channel, the transactions can not be sent. More specically, the channels are bi-directional and when they are created, each peer's (say A and B) one-way capacity is set independently. As a result, a channel's capacity in one direction (i.e., from A to B) can be totally consumed while the capacity in the opposite direction (i.e., from B to A) holds all of the funds. This means, while one of the peers, B, can make transactions in one direction, the other peer, A, cannot make any transactions through the channel. To be able to send transactions again, A needs to receive payments from B so that it can increase its channel capacity. Due to this feature, insucient funds in one-way channels drastically drops the chance of payment transfers in a PCN. Although the amount of funds deposited in channels during their establishment is a signicant factor that determines the success of network, the way they are consumed as the payments arrive in timeline is also impor-tant for long term sustainability. As an example, one of the recent studies [11] indicates that chance of sending a $5 payment successfully in LN, which is the most prominent implementation of PCN concept, is around 50% that makes it practically useless for end-users. Therefore, there is a need to address this issue to increase the success rate of transactions in PCNs.

In this paper, we propose the adoption of three specic techniques to improve the route selection in PCNs for a balanced PCN topology.

First, we advocate balance-aware routing that sets the weights based on imbalance rate between opposite directions on the same channel. We propose a common channel weight policy to be adapted by all nodes instead of letting users individually decide their weights. Fundamentally, the motivation is to encourage nodes to use high-balanced channels and avoid low-balanced ones for payments dynamically. This will help to utilize the channels in a manner that will keep available balances in the channels in all directions.

Second, another important feature of PCNs is that the end-users are connected to multiple points (gateways) in the network which can be used to initiate a payment. This feature can be exploited to re-balance the exhausted channels especially in case of skewness in the payment ow. Therefore, we propose a smart gateway selection named gateway inbound/outbound ratio which calculates the ratio of total inbound capacity to outbound capacity of each connected gateway and chooses the minimum among these. The insight that leads us to this method is that the bottleneck in PCNs is the gateways where the payments are initiated from or destined to the users. Routing nodes that constitute the backbone of the payment network are naturally being balanced since they transmit high number of transactions in both ways. However, this does not apply to gateway node and thus their selection needs to be controlled.

The third technique we propose relates to giant payments. Specically, to further increase the success rate for higher amounts, we focus on split payments, where a payment can be divided into pieces and sent independently. We propose a proportional split method that corroborates our second method by assigning payments to gateways based on their inbound outbound ratios. The user assigns the biggest portion to the gateway that needs most to balance the incoming balance to the outgoing balance.

We implemented and tested the eectiveness of the proposed approaches under various payment scenarios. For weight selection, we compared with uniform setting policy while for gateway selection, we compared our approach to other naive approaches such as greedy and maxow. Finally, for split payment, we compared to Equal share which distributes the total amount among the gateways. In all cases, the experiments results showed that our method improves the success rate signicantly.

This paper is organized as follows: Section 2 summarizes the related work in the literature while Section 3 provides some background explaining the concepts used in payment networks and our assumptions. Section 4 presents the problem denition and our approach. In Section V, we assess the performance of the proposed mechanism. Finally, Section VI concludes the paper.

2. Related Work

There exist works that attempts to realize, implement and improve PCNs by proposing various methods [20, 24, 25]. LN [11] designed specically for Bitcoin and Raiden [12] for Ethereum are two prominent examples in practice. LN was proposed in 2016 and deployed in late 2017. It is currently the most active and known platform with more than 10,000 nodes and 30,000 channels [21]. LN utilizes source-routing for transferring payments [17]. The node rst estimates a path with available channel capacity using its local-view global topology, then initiates the transaction. If the transaction fails, the recent path is excluded and try to nd another path. LN is still being improved, but it suers from problems such as reliability, scalability and eective routing. The topology of the LN converges to hub-and-spoke model [13] due to existence of high-degree nodes. The capacity in the channels has also not increased in proportion to its network size [14] which is another factor that inhibits its success in payment transmissions especially for higher amounts.

Apart from these practical implementations, there has been a lot of other research works which studied dierent aspects of payment routing problem in PCNs that may relate to our work in this paper. We summarize these eorts below:

Sivaraman et al. [18] attempted to apply packet-switching routing techniques to PCNs by splitting payments into micro-payments. The idea is creating smaller packets similar to maximum transmission unit (MTU) in traditional computer networks. However, creating tiny units may drastically increase the total fee as there exists a base fee for each payment regardless of the amount. The authors also employ congestion control and adopt best-eort model to improve payment throughput by specically choosing the paths that re-balance the channels. The payments are queued at spider routers and they are transferred when the fund is available. Thus, the payment transfer may not be instant since it may get stuck or delayed at some routers after it is sent from the source.

A distributed routing algorithm is implemented in Flash [19] to better handle constantly changing balances. Flash is using modied max-ow algorithm. Smaller amounts called mice and bigger amounts called elephant payments are treated in dierent ways. Small payments are sent randomly over pre-computed paths while for large payments, the nodes are probed to nd a channel with available funds. The reason behind this is that reducing the overhead of the probing messages is used to gure out available capacity. Since max-ow is an accepted approach used to implement a PCN [24, 19], we use it as benchmark in our evaluations to measure the performance.

Another work, called Revive [22], assumes that a node has multiple connections and the skewed payments make some of the links depleted. It tries to nd cycles in the network and a user sends a payment to itself to re-balance the depleted channel through others. The ecacy of the system depends on the existence of such a loop that each one in the loop would benet from a payment ow in the same direction. If everyone benets from this cycle, the process might be free, otherwise there might be some transaction fee. However, the proposal has not been evaluated to see how much it can benet in real case scenarios. Nonetheless, this method can be used as complimentary to any other technique including ours.

Flare [23] is using landmark routing where only some nodes store routing tables for the complete network. The rest only knows how to reach to one of those landmark nodes. A user transmits the payment to the gateway node which handles the rest. However, Flare is using static routing approach and does not consider dynamic channel capacity. Thus, it cannot adapt to varying conditions.

In our work, we rst classify the nodes based on their roles (end-user, gate-way, routing) in the network which aects payment trac passing through it. We explore on three specic techniques; balance-aware weight calculation, smart gateway selection strategy and proportional payment splitting which has a great impact on the overall success of payment channel networks. While most of the existing works focus on routing mechanisms, we contrarily highlight the importance of gateway nodes that connects end-users to the network. The routing nodes are in the inner region of the network and they are receiving high number of transactions in both ways. This makes them re-balanced quickly, and alternative paths can be found within the network. However, gateway nodes are more vulnerable to balance problem since they are at the edge and they have less trac. This can cause payment failures for the nodes connected through this gateway. Thus, our work focus on gateway nodes and keep their channels open. Moreover, we designed a splitting scheme of large payments in a way that it supports the previous idea by dividing the amounts proportional to gateway's inbound/outbound ratio. So, our work distinguish from existing work by putting a focus on gateway nodes to initiate the payment instead of the routing protocol.

3. Preliminaries and Assumptions

3.1. Blockchain and Bitcoin

Blockchain is similar to a distributed database where the building data struc-ture is called a \block". For Bitcoin, a block is simply comprised of transactions (data), timestamp, nonce, the hash of the block and the hash of the previous block[2] as shown in Fig. 1. The hash of transaction are inserted into a Merkle tree which enables users to easily verify whether a transaction is in the block or not. In cryptocurrency-based networks, the nodes come to a consensus for the approval of a block by proving that they have enough interest in the network. For instance, in Bitcoin, PoW mechanism is utilized. In order for a block to be accepted as valid, the hash of the block should be smaller than a number which is decided by considering the total accumulated computational power in the entire network. By changing the nonce value in the block, the miner aims to nd a suitable hash result. Soon after a valid block is found, it gets distributed in the network. After the other nodes validate that block, the next block calculation starts. In blockchain, it is possible to create smart contracts which enables participant to dene rules which will be enforced by the network. The joining parties will interact under the dened rules. It provides mechanisms to embed governance rules in veriable way that can be audited by the consensus algorithm.

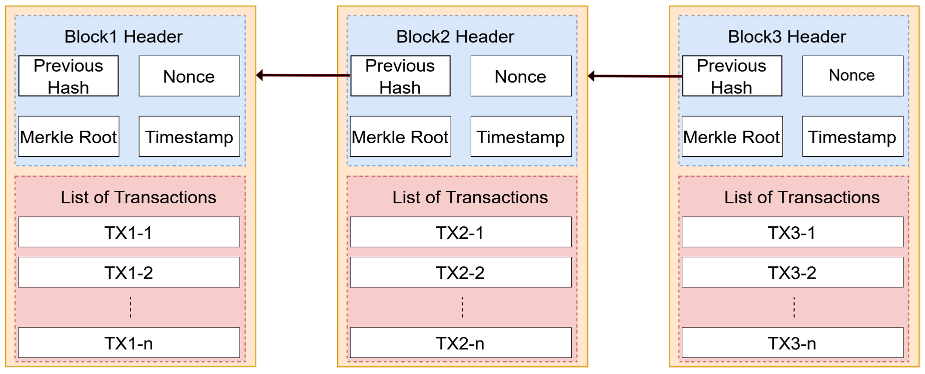


Figure 1: Blockchain Structure.

3.2. O-Chain Payment Channels

For Bitcoin, the average time spent for the approval of a block is around 10 minutes. This duration casts suspicion on the usability and practicality of the Bitcoin. More precisely, using Bitcoin for day-to-day spending becomes almost impossible. The reason for that is a payee waits at least 6 blocks to count a transaction to be valid. So, for example, if one buys a cup of coee and uses Bitcoin to pay, s/he has to wait at least one hour for the payment to get cleared. Moreover, during the congested times, s/he has to either pay a lot of transaction fees, possibly greater than the price of the coee or has to wait much more than the anticipated time. Apparently, this is an undesirable case not only for a customer but also for a shop owner.

To solve that problem, developers came up with a concept called o-chain payment channel that leverages the smart contract mechanism in blockchain. In this concept, two users, say A and B, come to a mutual agreement on establishing a business. Then they sign a contract by transferring collateral to a shared 2-of-2 multi-signature address and initiate the channel by publishing it on the blockchain. This contract type is called Hash Time Locked Contracts (HTLC). When the users agree on any amount of payment, they prepare a new HTLC, exchange the new contract, and update the state of the channel. To initiate a payment from a payer, a challenge, namely, a pre-image is sent to the recipient. If the recipient can reply successfully to the challenge, the contract becomes valid, and the ownership of the money gets transferred.

O-chain mechanism brings a huge advantage since the peers do not need to publish every transaction on the blockchain. That is, the payments are theoretically instantaneous. Moreover, as there is no need for frequent on-chain transactions, the transactions will be protected from uctuating, unexpectedly high on-chain transaction fees. In fact, a transaction fee can be 0 (zero) if the peers agree so. Thus, for a well-dened channel between honest peers, there will be two on-chain transactions: one to establish the channel and one to close (nalize) the channel.

An important feature of such a channel is that the direction of the payments matter. Specically, two ows from opposite directions on the same link negate each other's capacity consumption. This is shown in Fig. 2. Initially, a channel between two parties A and B is established. Both A and B put 100 unit of currency which in turn makes the channel capacity 200 units. After A makes 2 transactions each of which is 50 units, the directional capacity from A to B will be zero. Hence, A can not transfer any more unless B transfers back some money B sends 70 unit back and after they close the channel they get their corresponding shares from the multi-signature address.

3.3. Payment Channel Networks

O-chain payment channels can be extended to a PCN idea. As shown in Fig. 3, assume that A and B have a channel, and B and C have a channel too. If somehow, A wants to trade with C only, what s/he has to do is hash-lock a certain amount of money and forward it to C through B. As C already knows

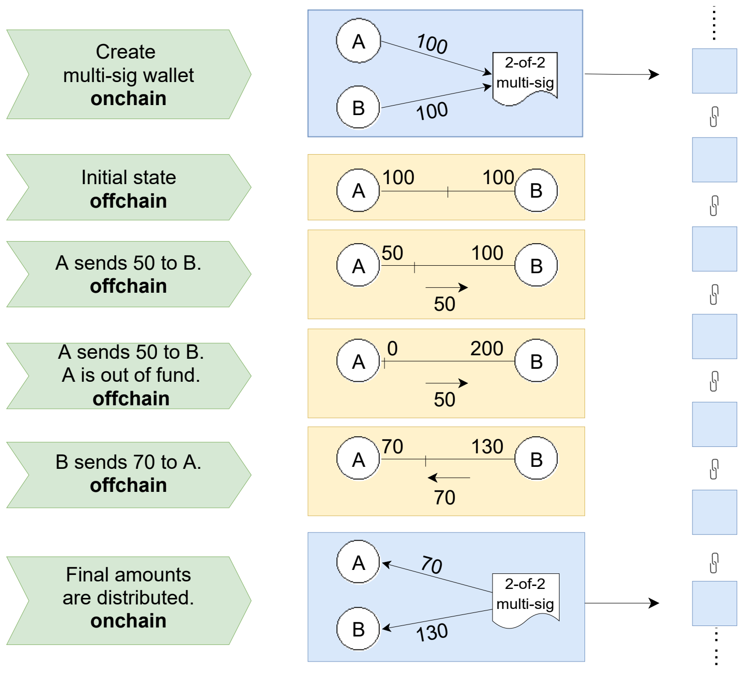


Figure 2: Illustration of a payment channel.

245 the answer to the challenge, C will get her/his money from B by disclosing the answer. The brilliance of the HTLC appears here. As C discloses the answer to the challenge, B learns the answer. Now, B will reply to the challenge successfully and get her/his share from A. In this way, one can reach everyone in a network through multi-hop payments forming a PCN.

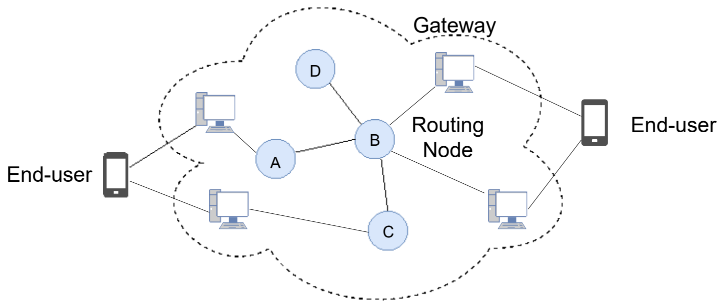


Figure 3: A sample payment channel network.

250 A PCN illustrated in Fig. 3 consists of three types of nodes in terms of functionality; end-user, gateway, and routing nodes:

End-user: An end-user usually makes payment to purchase an item and rarely receives payment such as refund. S/He connects to a gateway to access the rest of the PCN. S/He needs to maintain certain amount of funds in his/her channel with the gateway to continue sending payments. S/He should open the channel with the best gateway that can connect him to others continuously and cheaply. The best choice would be connecting to a gateway that s/he will usually have means for making direct transactions.

Gateway nodes: This type of nodes are usually the stores that expect to receive payments in cryptocurrency. They also relay the payments of their end-user's payments to other nodes. Gateway nodes should connect to routing nodes with good connectivity. They may not aim to earn transfer fees. Their primary purpose is to sell their products in cryptocurrency.

Routing nodes: These nodes typically act as backbone routers (similar to BGP routers on Internet) to connect gateways which are nodes A, B, C and D in Fig. 3. They regard having a node in LN as an investment opportunity. They try to have high number of connections and maintain the channels well balanced. This makes them a hub point in the network and lure the transactions so that they increase the return of their investments.

3.4. Assumptions

In this paper, we consider that a PCN consists of nodes connected with o-chain payment channels. The channel capacity represents the amount of money deposited in a 2-of-2 multi-signature address. Although any node can send to and receive payment from any other node, we want to distinguish nodes as end-users, gateways and routing nodes since we want to simulate the case that people want to use cryptocurrency for shopping and micropayments where the cash ow is mostly from a customer to a gateway (a store in real life). After a person establishes a payment channel with a gateway, s/he can make a payment to anybody in the network.

We build up our work on a presumed existing routing protocol and message gossiping mechanism which is used by the nodes to advertise the weights to the rest of the network and carry a specied payment from a source to destination. Each node is assumed to know the complete topology to calculate the path, and the updates about the channels are propagated to other nodes in the network as we are not focusing on the eciency and overhead of the routing protocol.

We would like to note that our approach is designed to be used in any PCN. While we use LN as an example to explain some concepts and problems, the proposed approach is not specically designed to work solely within LN. We consider only one type of (any type) cryptocurrency as we are not addressing the cooperation among dierent types.

4. Proposed Approach

4.1. Problem Motivation and Overview

In a PCN, an end-user transfers a payment to a store by initiating the payment via a gateway. This payment rst goes through the gateway, then routing nodes and arrives at the destination unless there is a direct channel between the end-user and the gateway. During this process, various problems might occur causing the payment to fail or leading to ineciency in transmission. Multi-hop transmission may not be completed because of insucient funds in the channels. This is not acceptable in many applications where payments need to be done real-time and the service should be available at all times. We discuss some challenges below that might cause a payment to fail and hinder the overall success of the network:

Problem: Highly Directional Payments: If nodes constantly transmit payments in one direction on the same channel, the balance of the channel will be depleted in that direction. The funds in channels may shift to one particular side of the channel if the payments are highly directional. It will create a weak connection or disconnection in the network which might cause: 1) a group of nodes to be disconnected from the rest of the network until the balance of the channel is increased; 2) a node to travel longer paths; and 3) two payments arriving a node simultaneously not get transmitted due to lack of available funds. Balance-aware routing and multiple connections to gateways can be exploited to address this issue.

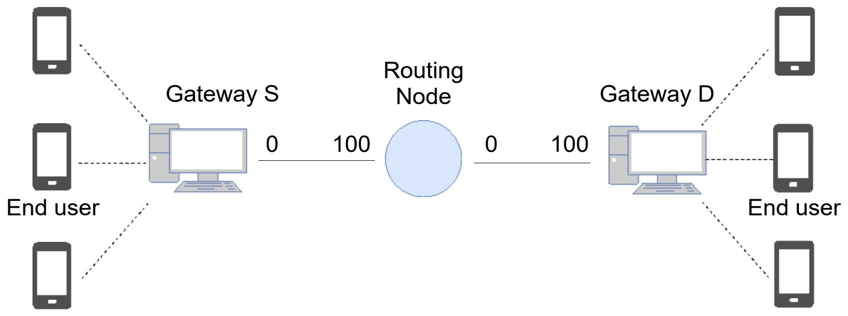


Figure 4: Depleted channels cause disconnection.

Problem: Over-used Gateways: The second problem is about the gateways which are used by the users as rst hop to send the payments. If a gateway has used up its outgoing capacity, then it can not initiate any payment and function as a transit node unless it receives some payment. Suppose that a store is having a busy day and receiving many payments. The channel between the store (gateway) and a routing node will be depleted and the store will not be able to receive payments anymore. This is depicted as an example in Fig. 4. In this gure, the end-users connected to Gateway S can not make payments through this node as its outgoing channel capacity is 0. They have to wait for Gateway S to receive some payments destined to it. Similarly, if a gateway consumed all of its incoming capacity, then it can not receive any payment and function as a transit node as well until it is used to send some payment. In Fig. 4, Gateway D can not receive any payment since the incoming channel capacity is 0. It has to wait for any end-user to send payments originating from D. While routing nodes may not suer dramatically from this type of problem because of high number of transactions and well-balanced channels, the gateway nodes are particularly prone to channel balance exhaustion. A proper gateway selection will be useful to remedy this problem.

Problem: Giant Payments: The problem with atomic payments is that there must be a wide enough single channel to carry a payment from source to destination. In such a system, large payments will cause major shift in channel capacities which might reduce the success of the transactions in the overall system. The results of the experiment shown in Fig. 5 performed by [26] indicates that success rate decreases dramatically with increasing transfer amount. LN recently has introduced the capability of splitting payments while it does not dene how to do it. We argue that a smart splitting methodology help to better build a scalable payment network.

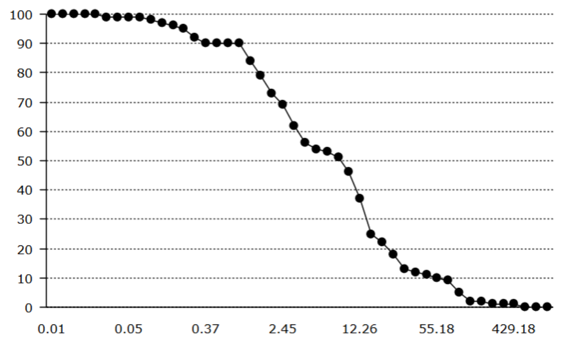


Figure 5: Success rate vs amount of the payment [26].

Problem: Lack of balance knowledge: LN currently only reveals the total balance of each channel and hides the directional balances. It also restricts the frequent fee updates across the network. The users are supposed to nd an available path according to their local routing tables which must be updated after unsuccessful payment attempts. Even though the developers might have their own reasons for this design choice, it certainly reduces the overall success of the transactions in the network. Additionally, node owners are not subject to any rule when they are setting the fees. Users prefer low-cost fee without paying attention to depleting balances in channels. They calculate routes based on the optimum fee which are set by node owners. The user optimal ow may be dierent than the system's optimal ow. A common weight policy might be helpful to create a more sustainable network.

In the rest of this section, we propose three specic techniques to deal with the problems mentioned above. We detail each of these ideas separately below.

4.2. Balance-aware Routing

One of the most important factors for the success and sustainability of the PCNs lies in keeping the channels balanced. Therefore, we propose using a mandated weight calculation method which must be adopted by all the nodes in the PCN when nding their payment routes. As well-known, shortest path algorithms such as Dijkstra, utilize weights as link costs. Basically, each directed edge in the network is assigned a weight inversely proportional to its current channel capacity. The route calculations for each node will then be based on this newly assigned weight. Specically, the weight of each channel is computed using the following equation by each node:

$$W(u;v) = \frac{B(v;u)}{B(u;v) + B(v;u)} \tag{1}$$

where W represents the directional weight of a channel between two edge nodes u and v, $B(u;v)$ is the current balance of the channel from u to v. This method adjusts the weight according to the balance on each side of the channel. Note that the links between nodes are directional and thus there will be a dierent weight from v to u.Consequently, this new weight will strongly encourage users to use channels with available balance while helping them to avoid routing over low-balanced channels.

Advertising the updated weight across the network requires an eective information dissemination method and sending an update message after each payment transmission may bring additional overhead. Our model assumes a gossip protocol such that nodes establish information channels, in addition to payment channels, with other nodes to expedite the message ow. Each node advertises its current weight to its neighbors periodically. In order to diminish the number of message exchange, the nodes are grouped into clusters, and the leaders of these cluster, called landmark, make a second layer similar to intra-AS and inter-AS used in Internet to eectively disseminate message. Moreover, instead of sending an update message after each payment, threshold values are set to decide a message generation for nodes. For instance, after each %25 change in the directional capacity, a fee update is initiated. This reduces the number of messages drastically while providing a dynamic route information.

In Fig. 6, it is the user's responsibility to keep the channel between himself and gateway balanced so that he can make transaction which is assumed to be existing. The algorithm is focusing on the connections among core nodes of the network (gateway and routing nodes). The end-user devices are connected to one or more nodes in the PCN. In case there is only one available connected gateway, the user has to send payment through this specic one using shortest path.

Having multiple connections enables a user to initiate the payment from various vantage points. Specically, although we may not control the destinations of the payments, we can inuence the initial gateway connection points for users. The route to the destination is calculated from all these points and the shortest one is picked which is expected to contribute to the network stability because of the weight calculation method

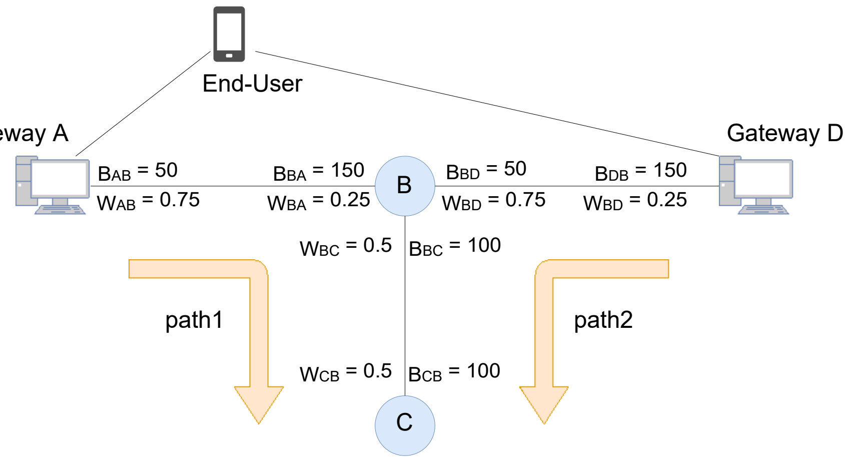


Figure 6: Route selection of an end-user.

For instance, the user shown in Fig. 6 has two options to initiate the payment to C: 1) path1 (cost = 1:25) which starts from A and 2) path2 (cost = 0:75) which starts from D. Per our approach, the user will choose path2 after the route calculation. This choice will help node A to preserve its limited outgoing capacity (C_{AB}) and keep the channel between A and B balanced in both directions. It will also increase D's incoming capacity, C_{BD} (due to payment sent to B) and more equally balance the channel between B and D.

Overall, the proposed route calculation is given in Algorithm 1. The weight for each link is calculated rst using Equation 1. Then based on the link weight, each node computes the shortest path to destination from its available connections using Dijkstra's shortest path algorithm. From amongst these, the minimum cost path is selected to initiate the payment. The computation-demanding part of the algorithm is shortest path calculation which should be repeated for each connected gateway. The number of connected gateways does not increase with the network size, but mostly limited to a few (i.e., constant cost) for the end-users because of the nancial cost. So, the complexity of the proposed algorithm is $O(E \log V)$ where E is the number of edges and V is the number of vertices in the network.

4.3. Smart Gateway Selection

As mentioned, a unique trait of payment channel networks is that a payment can be initiated from various points. Channels can be re-balanced by a smart selection of gateway with the help of cooperative end-user node implementation.

Algorithm 1 Route Calculation

```
1:  Input: C=Store connection list, G=Connected directed graph
2:  for every edge, e(u; v), in G do
3:      weight(u; v)=bal(v; u)/[bal(u; v)+bal(v; u)]
4:  end for // weight calculations are done
5:  min = Integer.Max
6:  for every connection, s in C do // Calculate shortest path from each point
7:      Path=ShortestPath(G, from=s, to=d)
8:      if Path less than min then
9:          min = Path
10:     end if
11: end for
12: Output: min
```

Gateway nodes that connect end-users to network are a critical factor for the success of the overall system. Insucient funds in gateway channel will have greater impact than the routing nodes. Routing nodes that constitute the backbone of the PCN are naturally being balanced since they transmit high number of transactions in both ways. However, gateway nodes need to be taken care of explicitly.

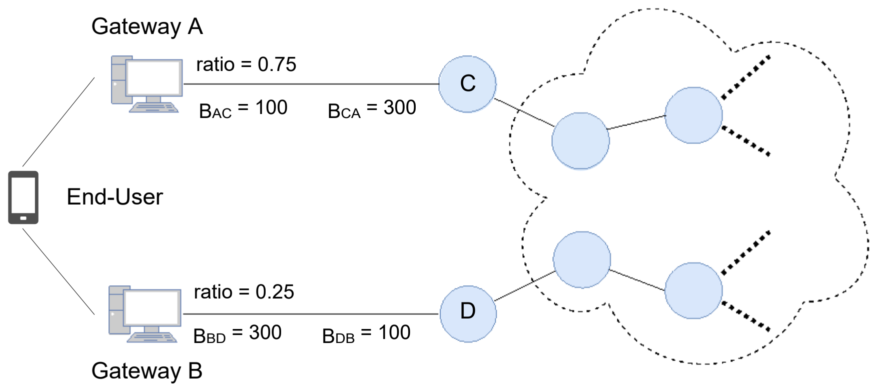


Figure 7: Smart gateway selection.

Thus, we propose a gateway selection method that utilizes multiple connections eectively between the end-user and the gateway to create a more stable network. This method focuses on the channel balances of the gateway channels and strive to keep them balanced both inward and outward. The rationale is balancing incoming and outgoing payments amounts for the gateways so that they are not blocked from sending or receiving future payments. Therefore, we refer to this as Gateway Inbound/Outbound Ratio approach. A gateway receiv-ing a high volume of payments should be preferred as the source, and a node which has less outgoing capacity should be avoided to originate payments.

For instance, the user shown in Fig. 7 has two options to initiate the payment; GatewayA and GatewayB. The user will choose second option by looking at the ratios (GatewayA ratio = 0:75 and GatewayB ratio = 0:25) calculated using total inbound capacity to outbound capacity of each connected gateway. After we decide the gateway to initiate the payment, the rest of the path is calculated using shortest path algorithm. To clarify the distinction from the previous section, it is worth to emphasize that this method considers the fund distribution on the channel between gateway and routing node to select the initial gateway. After deciding the gateway based on this criterion, it utilizes the shortest path from there to reach the destination.

Algorithm 2 Inbound/Outbound Ratio

```
1: Input: C=Gateway list, G=Connected directed graph
2: for each directed edge, (u; v), in G do
3:     weight(u; v)=bal(v; u)/[bal(u; v)+bal(v; u)]
4: end for // weight calculations are done
5: min = Integer.Max
6: for every gateway connection, s in C do // Find the gateway with lowest
   inbound/outbound ratio
7:     total_in = total_out = 0
8:     for every channel, (x; y), of gateway s do
9:         total_in = total_in + capacity(y; x)
10:        total_out = total_out + capacity(x; y)
11:    end for
12:    ratio = total_in / [total_in + total_out]
13:    if ratio less than min then
14:        min = ratio
15:    end if
16: end for
17: Path=ShortestPath(G, from=min, to=d)
18: Output: Path
```

The pseudo-code presented in Alg. 2 calculates the total inbound and outbound capacity for each connected gateway, and then it nds the one (gateway) with the lowest ratio. Basically, the inbound/outbound ratio is used as the metric when choosing the gateway to initiate the payment. Since we are using single payment method, the links which do not have enough capacity to transmit the amount is not considered in path computation. If any of the gateways does not have the required capacity, that is ruled out in the path calculation either. When the gateway is decided, the transaction process starts. Sending the payment through gateway that has lower inbound/outbound ratio will shift the balance to the other side, thus will make it balanced. The shortest path computation in the algorithm is again computation intensive part as the previous portion does weight assignment for each link and ratio calculation for each gateway.

4.4. Split Payments

The last challenge that we focus on is splitting payments. The ability to split a payment might allow big amounts to be transferred by using multiple gateways if a single path is not enough to carry the totality of a transaction. That should increase the success rate and help manage the balances over channels better.

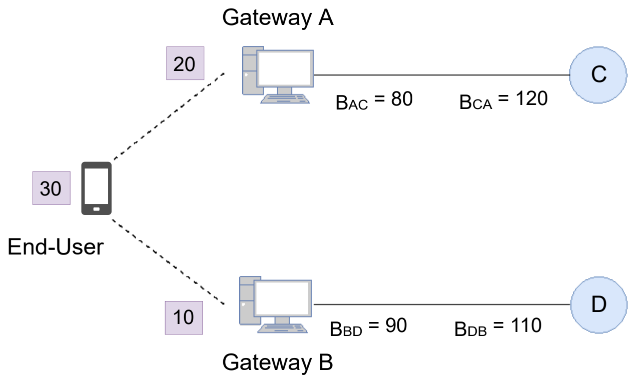


Figure 8: Proportional Split.

A payment can be split in many ways, such as dividing into very small units (i.e., micropayments) or equal bigger chunks. These chunks can be sent through multiple gateways. These methods might incur additional transaction costs depending on the fee policy. For instance, if there is a base fee for each transaction, micropayment model will be infeasible. We consider a proportional splitting method as detailed below:

Proportional Split: In this split method, we consider a split criterion based on the gateways' inbound/outband ratios instead of the number of gateways. We propose this method to further improve the Inbound/Outbound Ratio method proposed in the previous subsection. The total amount is divided proportional to inbound/outbound ratio of the gateways if possible. Specically, the gateway with less outbound capacity compared to inbound capacity transfers less amount of transactions. For instance, the user shown in Fig. 8 divides the amount 30 as 20 and 10 so that both channels will be equally balanced.

This method should help the gateways better to stay balanced especially for the large payments. Note that this method is also in line with our gateway selection approach in the previous subsection as they both consider the same criterion. Thus, we expect that this splitting will further improve the performance in terms of success rate.

The details of the splitting algorithm are as follows: The user rst check if there is enough total capacity from all connected gateways to the destination. If the calculated capacity is not enough, splitting also does not help to perform the transfer because basically there is no route for transferring this amount. If we nd out that a sucient capacity exists, then we divide the payment into multiple chunks based on the number of available connections of the end user. So, we implement maxow using Ford-Fulkerson ow algorithm [27].

Algorithm 3 Proportional Split

```
1: Input: C=Gateway list, G=Connected directed graph, A=Total amount to
   send, s=source, d=destination
2: Flow=MaxFlow(G, from=s, to=d)
3: if Flow less than A then
4:     break
5: end if
6: B = C // copy gateway values to temp list
7: unitpay = 0
8: S[n] = f0g // calculate amount for each gateway
9: while unitpay not greater than A do
10: B_i = findGateway(B) // Find the gateway with lowest in-
   bound/outbound ratio
11:     B_i:outbound
12:     B_i:inbound + +
13:     unitpay + +
14:     S[i] + +
15: end while
16: for every connection, s in C do // Calculate shortest path from each gateway
17:     Path=ShortestPath(G, from=S[i], to=d)
18: end for
```

Alg. 3 computes the amount to be sent through each gateway and shortest path from all these gateways to destination. The user starts checking with the available bandwidth. If the existing capacity is less than the amount we want to send, the payment will fail. In case the capacity is sucient, the next thing it does is calculating the amount to be assigned to each gateway. It starts with assigning a unit amount to the gateway with lowest ratio. This procedure is repeated until the whole amount is distributed to the gateways.

In this algorithm, while loop executes in linear time as the maxflow and shortestpath are the computation intensive parts. The running time of the shortestpath is $O(E \log V)$, while the complexity of the maxflow is $O(V E^2)$. Thus, the complexity of the split algorithm is determined by the maxow.

5. Performance Evaluation

This section presents the experiment setup and results for the proposed approaches.

5.1. Experiment Setup

We developed a PCN simulator in Java that allows us to run the experiments and measure the dened metrics. We list the necessary parameters in Table 1 to set when running the experiments.

Table 1: Experiment Parameters

Number of Nodes	100
Degree of a Node	3
Initial Channel Capacities	50 to 150
Payment Amount	5 to 85
Number of Payment	5K

Network conguration: The results are based on a random regular network with 100 nodes (gateway and routing nodes), each with degree 3. We want to have a at topology as the LN is criticized because of the existence of central nodes which undermines the decentralization idea. The channels between the nodes are assigned a random capacity uniformly distributed between 50 and 150 which is similar to regular channel capacity in LN [16]. All nodes (routing and gateway) in the network have similar capacity. In fact, any node in the network can be gateway and routing node. The categorization is based its role for a specic payment. It can just transmit a payment, then it is considered as routing node, or it can be the initiation (or destination) point for a payment then it is considered as a gateway.

Payment les: We build dierent payment sequences consisting of 5000 end-to-end transactions. Each node sends and receives 50 transactions on average. Each transaction amount is selected randomly during the experiment within a specied range. Source and destination are not necessarily the same which means that node A transmits to B but may receive from C. The imbalance rate refers to the dierence between the number of incoming and outgoing payments for a node.

Payment transfer: Each node calculates the path using specied method and the payment is sent through intermediate nodes by decrementing the amount from each channel used and incrementing in the opposite direction. In the rst set of experiment, we use only single payment while in the second part we use split payment.

Experiment run: Each payment le is run with various seeds which randomized the initial channel capacity and payment amount. Thus, the results are aggregate of randomized tests on the network using 100 dierent seed value.

5.2. Performance Metrics

We use the following metrics to assess the performance of the proposed approach:

- Success rate: This metric shows the average of the number of payments that could be sent successfully from a source to destination for the whole PCN.
- Fee: This is the total amount of fees that an end-user has to pay to node owners on the path to destination. Note that in our experiments, the

fees are calculated proportional to transfer amount at each hop without a base fee. Obviously, actual fee depends on market value and policy of the underlying payment network.

Network imbalance: This metric measures the capacity dierence between two sides of a channel. We use the average for the network to show the overall imbalance.

Capacity Distribution: It shows the distribution of channel capacity for the network which is similar to network imbalance.

Path Length: This is the number of hops that an actual payment has traveled when the payment is executed. This could be important if there is a base fee or any delay coming from intermediate nodes.

Network Diameter: This is the longest path between two arbitrary nodes in the network.

5.3. Experiment Results

In this section, we present the results collected from the experiments.

5.3.1. Balance-aware Routing Results

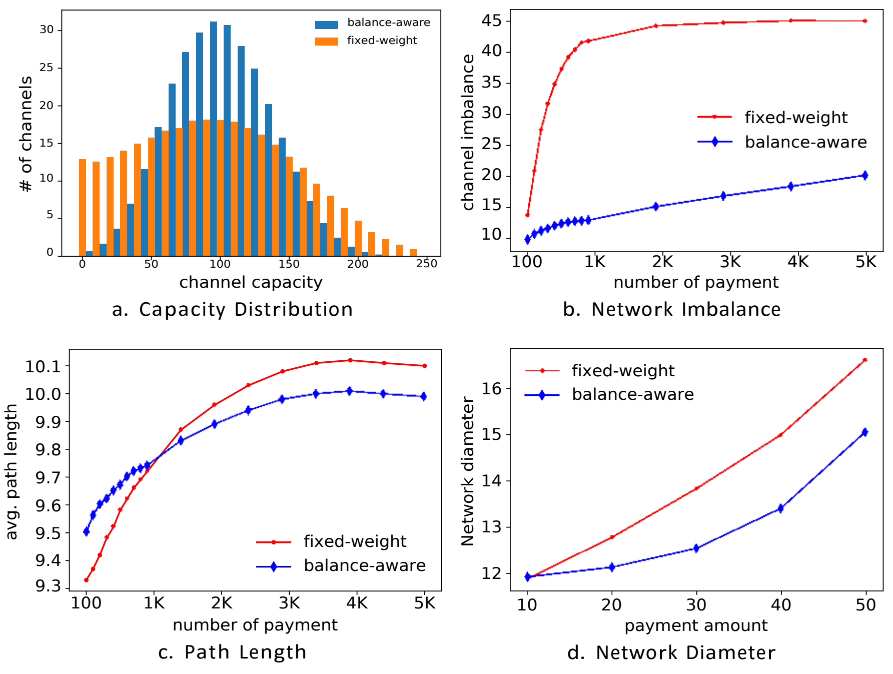


Figure 9: Experiment Results for Balance-aware Routing using Single Connection

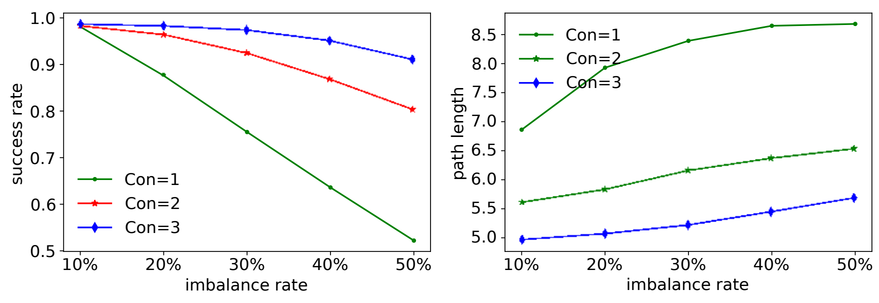
We rst evaluate the balance-aware routing where we applied common weight policy using a single connection from each customer to store. As a baseline for comparison, we used xed-weight policy (static) where the weight for each edge is set based on xed fee. It means any node owner on the network can decide his own fee, and this fee is not updated and disseminated dynamically throughout the network. Even the two sides of a channel may apply dierent rules when setting the fees. Therefore, there is not a cooperative eort to keep the channels balanced. The reason we chose such an approach to compare is that LN adopts a free market model and discourages the nodes to update the fees frequently.

We rst looked at the channel capacity distribution. To this end, we created 300 directional channels all of which were initially assigned a balance between 50 and 150 in a uniformly distributed manner. The payments are sent from source to destination according to the scenario in the payment le each of which contains 5000 payment. Fig. 9a shows the channel capacity distribution after all the payments are executed. A bar in the gure represents the number of channels whose capacity is between a given x and x + 10. As seen, when we apply random xed-weight policy, the distribution gets attened. A substantial part of the channels is low-balanced. A quarter of them drops below 50 which was the lowest value for a channel in the initial setup. 25 channels' capacity is less than 20. In case of our proposed balance-aware routing, the channel capacity distribution resembles a Gaussian distribution with a mean 100. 85 % of the channel capacities are still within the initial range. The number of channels whose balance is less than 20 is only 3. This suggests a more balanced network.

To measure the deviation more precisely and quantify it, we did another experiment to assess the network imbalance. We set all the channels to 100 so that we can measure the variance accurately. We preferred to use a timeline in the x-axis to see the change. After every hundred payment, network imbalance is measured. As shown in Fig. 9b, for the rst case, it increases dramatically, which continues to increase slowly. For our approach, the value is much less and the increase is steady. It ends up with an average distance of 18. It is obvious that the channel capacities are staying closer to mean.

We then investigated the impact on path length for payments and network diameter on the network topology. First, we checked the number of hops each payment in the payment le has to go through. For this experiment, we used a network with 1000 nodes to magnify the results. As seen in Fig. 9c, the quantitative dierence between the results of the two methods is not signicant. The trends are also similar. It increases with time, then becomes steady. This can be explained as follows: At the beginning, our approach uses longer paths because we force payments to travel over high capacity channels even though they are longer. Fixed-weight approach takes advantage of shorter paths at the beginning at the cost of balance exhaustion. This causes payments to go over longer paths at later stages.

Finally, Fig. 9d shows the diameter of the network for various payment amounts. Applying common weight creates a more compact network. Note that the path between two nodes might be dierent based on the amount. Higher amounts have to travel longer paths. Our approach provides better connectivity over the network.



a. Success rate b. Path Length

Figure 10: Experiment Results for multiple connections available

In the rst batch of the experiments, the payments among the nodes are well distributed which means that the number of payments a node sends and receives is same even though the amount is chosen randomly in a specic range. This enabled us to observe the impact of weight policy without getting into success rate discussions. In this subsection, we perform additional experiments for the cases where customers have multiple connection points to stores. We generated scenarios where the payments sent and received for a particular node is not equally distributed. Specically, when the payments are skewed, the number of incoming and outgoing payments will not be equal for a node, and thus unsuccessful payments occur because of channel depletion.

To create imbalance among payments sent and received, we varied the dier-ence among these from 10% to 50%. For instance, if the dierence is 10%, then the number of payments sent will be 10% more or less than the number of payments received. We tested single (C1), double (C2) and triple (C3) connections against varying imbalance rates (10 to 50).

As seen in Fig. 10a, the success rate for single connection (C1) drops to 50% while it is around 90% for triple connection (C3). Additional connections enable an end-user to use an alternative route in case one gateway is disconnected because of outgoing capacity erosion, and re-balance asymmetric channels. Fig. 10b shows the average path length of payments for each connection case. These results indicate that multi-connections also reduce the total number of hops that a payment has to go through.

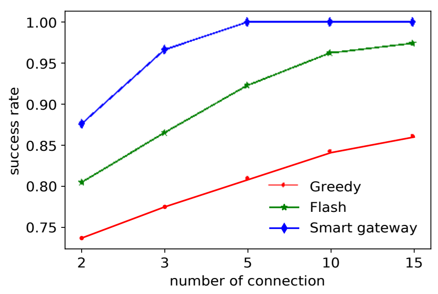
5.3.2. Smart Gateway Selection

In the previous section, we showed that multiple connections from users to various gateways signicantly improves the success rate. This part presents the evaluation for our gateway selection method inbound/outbound ratio by comparing to greedy which prefers the shortest path and max-ow which prioritizes available capacity from source to destination. Before proceeding to evaluations, we give short descriptions for these two approaches.

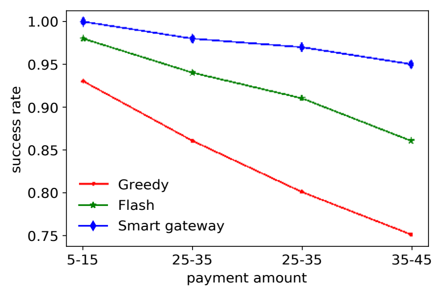
Greedy: This approach chooses the shortest path from available gateways. We refer to this approach as greedy method since it focuses on minimum fee and represents a selsh approach. In this approach, the shortest path is computed from connected gateways to the destination in terms of weight, and minimum path among those is selected. The user behaves selsh to minimize the transaction fee.It does not consider either available capacity of the path or the imbalance problem over channels [11, 28]. For instance, if there is one hop path with low capacity, that will be selected to minimize the fee even this might block the path for future payment in the same direction. This approach corresponds to the technique we explained in Section 4.2.

Maxow: Some works including Flash[19], CoinExpress[29] and SilentWhisper[24] adopted maximum ow as a solution by approaching the payment channel network as a commodity ow problem. So, we implement maxow approach representing the Flash that computes the maximum ow from source to destination which prioritizes the path capacity and leaving the fee as a secondary factor. In this method, instead of choosing the low-cost path in a selsh way, the maximum ow from each gateway to destination is calculated using Ford-Fulkerson ow algorithm [27]. Then, the approach uses the path with highest channel capacity. The idea is leaving available funds in the channels after making a transaction so that upcoming payments will have higher chances to nd a proper path. It calculates the maximum-ow value from each gateway, then initiates the payment from the gateway that has highest value.

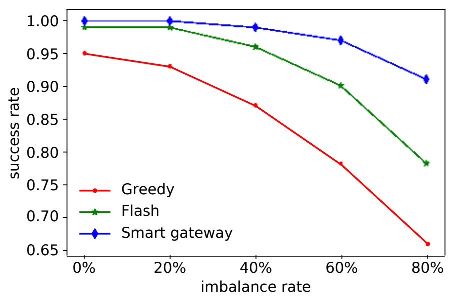
Fig. 11a shows the success rate with respect to the number of gateway connection, we observe that each method is performing better with higher number of connections to gateways as it increases the chances of having a route to any destination in the network. Having higher number of connections means holding more money in the channels for end-users. Based on the results, there is a diminishing return for excessive connections after three. The optimum number of connections may dier based on the size and connectivity of the network. The greedy approach performs worst in terms of overall performance of the network while it provides cheapest transaction cost for successful payments as will be seen in Fig. 11d. One reason for lower performance is that it consumes the channel funds selshly without taking higher balanced routes into consideration. That causes huge skewness in channels as seen in Fig. 11f. The max-ow method chooses a gateway with higher total capacity to destination to send the payment which yields better performance compared to the rst one since utilizing wider paths will leave space for possible upcoming payments that might use the common channels. However, it incurs higher fee per transaction as seen in Fig. 11d. Our proposed approach gives the best success rate among all. This is because, it focuses on keeping the gateways open for end-users. It is easier to nd a path in the inner part of the network if the payment can go through source and destination gateways. With the current setup, success rate with 5 connections reaches almost 100% with around 30% higher fee than the greedy method. It also keeps the channels equally balanced in both directions as seen in Fig. 11f which helps the network to stay stable. Given that the fees in payment channel networks are much lower compared to blockchain networks, a small in-



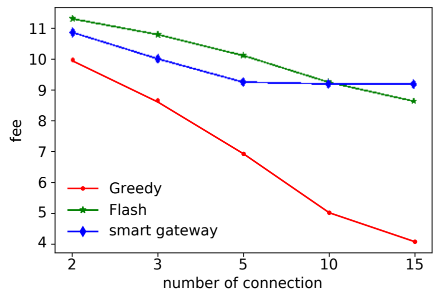
a. Success rate vs. connection count



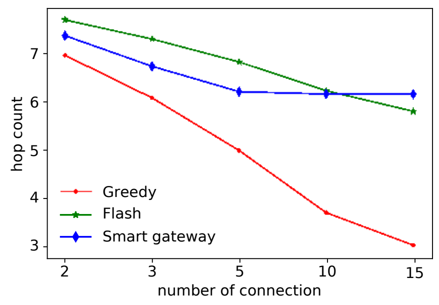
b. Impact of payment amount



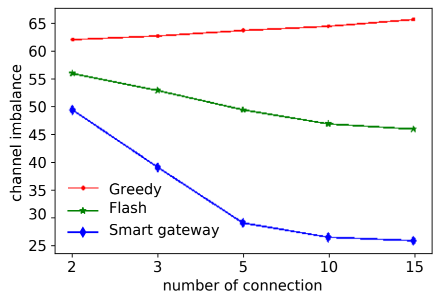
c. Success rate vs. imbalance rate



d. Relative Fee



e. Hop Count



f. Channel Imbalance

Figure 11: Experiment Results for Smart Gateway Selection

crease in the fees for almost guaranteeing all the transactions is a reasonable price.

We then look at the impact of payment amount on the success rate as shown in Fig. 11b. We observe that all the methods are suering when the amount is increasing because it becomes harder to nd a path. However, our approach still signicantly outperforms others. The gap even increases with the increased payment amounts.

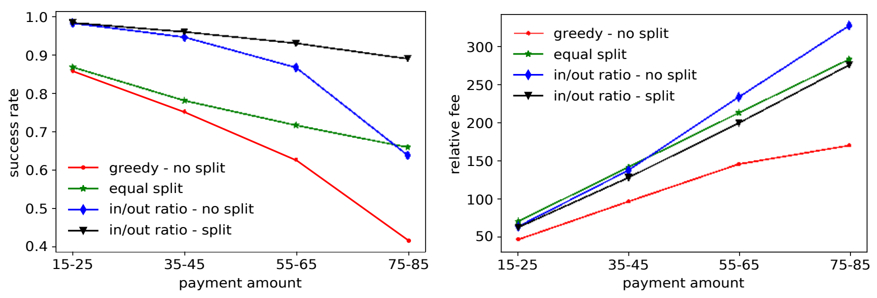
The imbalance rate also signicantly eects the success rate because dier-ence between number of incoming and outgoing payments for the nodes is high which consumes the funds in channels most in one way. Thus, the chance of nd-ing an available path decreases dramatically. Fig. 11c shows that our approach stands out to still maintain high success rate. One of the main challenges that we try to address with smart gateway selection by utilizing multi-connection is imbalance in sent and received payments. According to the results, focusing on gateways with in/out ratio is a valid strategy that increases the success rate.

Fig. 11d compares the relative transaction fees for the proposed methods. The numbers do not represent the actual fees, we use those number for the sake of comparison. Actual fee depends on market value and policy. Increasing the number of connections reduces the fee since there will be shorter paths. The fee in greedy method decreases linearly because it always chooses the cheapest route, and extra connections will provide better options in this respect. On the other hand, it stays steady for our approach after a certain point because we try to assign payments based on gateway channel balances. Note that based on the actual fee, the dierence between the methods may not be very signicant. Current fee in LN is as small as negligible. It is not easy to predict how the network and fees will shape in the future. There can be added some other factors in the fee calculation and path preference. Routing nodes with high connectivity and channel balance may apply higher fees as the network converges to highly centralized structure. Fig. 11e shows number of hops that a payment has to travel to arrive at the destination. The trends are similar to fee results since the fee is directly related to number of hops it has to travel. Average number of hops is also related to degree of the nodes which is set as three in our test network.

In Fig. 11f, we quantify the channel imbalance which basically shows the dierence between inbound and outbound capacities in the channels. The difference should be kept lower so that the payments can be transferred from both ways. The greedy method makes the network suer from imbalance problem and causes higher number of failures while our method method relieves it. Higher connectivity helps assigning payments to gateways who needs most to adjust the imbalance. The nodes just routing the payments from one neighbor to another will be balanced naturally in the long run because of the high number of transactions. But the nodes being origin or destination for the payments should be handled explicitly.

5.3.3. Results for Split Payments

In order to evaluate the performance of proportional split payment, we de-ne a benchmark Equal Share Split [30]. In this model, the payment request is divided among the gateways in equal shares if they all are capable of transferring that amount. We still check if there is enough capacity to carry the payment from source to the destination. If we nd out that a sucient capacity exists, then we divide the payment into multiple chunks based on the number of available connections of the end user. We do not consider any criteria of the gateways such as total capacity, fee or ratio. All the gateways will forward equal amounts if possible. However, if any of them does not have enough capacity, then others will have to carry more to compensate for the remaining amounts. The idea is utilizing the channels fairly by distributing the payments among the gateways as much as possible.



a. Success rate vs. payment amount b. Fee vs. payment amount

Figure 12: Experiment results for split payments.

Bigger payment size reduces the success rate dramatically. Split payment is the only solution for this problem. We present the success rate results of split payment solutions by comparing it with previous (without split) methods for varying payment amounts and connections. Specically, two split methods are implemented: equal split that distributes the payment among connected gateways by dividing into equal amounts to assign to gateways, and ratio proportional split that divides the payment based on the inbound/outbound ratio. Thus, there are four approaches we compare: 1) greedy approach without splitting (greedy - no split); 2) our approach without splitting (in/out ratio - no split); 3) equal split (equal split); and 4) ratio proportional split (in/out ratio proportional split). The computation time for the split method for the current setup takes less than a millisecond (0.2 ms) on a regular computer which will be higher for larger networks or resource constrained devices.

In Fig. 12a, we varied the payment amount from 20 to 80 to observe the success rate of each approach. As can be seen, for smaller amounts, splitting does not have much impact on the results. However, with increasing transaction amounts the dierence becomes obvious. For both split methods, the success rates are decreasing linearly while those of single ones are decreasing much faster. Our proposed ratio proportional splitting performs the best. It beats the no-split version of the same approach signicantly especially when the payment amount increases. In addition, the equal splitting is better than the greedy approach with no splitting in any case because it distributes the payments to all connected gateways.

When we look at the fees shown in Fig. 12b, the dierence is not signicant and the trends are similar. It should be noted that we do not apply any base fee. If there is a base fee in payment channel network design, then splitting might cost more in proportional to the number of splits. In that case, micro payments (splitting payment into tiny amounts) will not be advantageous.

6. Conclusion and Future Work

In this paper, we showed the eciency of balance-aware routing by proposing a common weight policy across the payment channel network which keeps the channels better balanced in both directions. We then presented a smart gateway selection method for end-users to send their payments in order to improve the success rate of these payments. It is especially useful under imbalanced payment scenarios. We observed that keeping the gateways open in terms of channel balance to send and receive payments is an important objective which can be achieved by considering inbound and outbound capacity balance. Proportional split method is introduced to further increase this success rate and to tackle giant payment amounts. We investigated the eectiveness of the proposed methods by comparing them with common approaches with extensive experiment analysis. The results indicated the eectiveness of our balance-aware, gateway selection and split methods in achieving high success rates.

We plan to extend our work in various ways. We want to investigate a threshold-based weight update scheme which can reduce the messaging overhead and contribute to the privacy of payments since the nodes will disseminate the new weight not after each payment transmission. As long as the balance stays within a range despite many payments in opposing directions, the weight will be same and not disseminated which can improve the overhead at the cost of degraded performance in terms of success rate. Integration of cryptocurrency and payment channel network with Internet of Things will enable automatic machine-to-machine payment and foster novel application in IoT ecosystem. Since such an ecosystem will be dependent on nancial micro-transactions among digital objects/devices, a reliable payment system without human intervention is desirable for a seamless experience. Thus, we want to steer our work towards this direction to create IoT-compatible and secure integration methods. There exists various blockchain designs each of which aims improving specic targets at the cost of others such as scalability vs. security while it is dicult to address all which is known as blockchain trilemma. So, it seems multiple cryptocurrencies will survive among many proposals which will require handling cross chain operations, some of which may be more vulnerable to the attacks. Transferring an asset from a low secured blockchain based coin to a highly secure one might put the latter one into danger. Such a conversion might be possible either through a central governance (which is actually not aligned with blockchain's decentralization idea) or one blockchain must follow the other's PoW. Although this is a challenging problem to be solved since each has dierent security levels, it might help to solve existing problems of single blockchain; however, it requires further investigation and ecient solutions for wide adoption of cryptocurrency.

References

[1] Narayanan, A., Bonneau, J., Felten, E., Miller, A. and Goldfeder, S., 2016. Bitcoin and cryptocurrency technologies: a comprehensive introduction. Princeton University Press.

[2] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. Technical Report, https://bitcoin.org/bitcoin.pdf, 2008.

[3] G. Wood. Ethereum: a secure decentralized transaction ledger, 2014.

[4] Crosby, M., Pattanayak, P., Verma, S. and Kalyanaraman, V., 2016. Blockchain technology: Beyond bitcoin. Applied Innovation, 2(6-10), p.71.

[5] Coin Market Cap https://coinmarketcap.com.

[6] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich. Algorand: Scaling byzantine agreements for cryptocurrencies. In Proc. ACM SOSP, 2017.

[7] Kyle Croman, Christian Decker, Ittay Eyal, Adem Efe Gencer, Ari Juels, Ahmed Kosba, Andrew Miller, Prateek Saxena, Elaine Shi, Emin Gun Sirer, Dawn Song, and Roger Wattenhofer. On scaling decentralized blockchains. In FC, 2016.

[8] Transaction Rate of Bitcoin. https://www.blockchain.com/en/charts/transactions-per-second, 2019.

[9] J. Poon and T. Dryja. The bitcoin lightning network: Scalable o-chain instant payments. Technical Report, https://lightning.network/lightningnetwork- paper.pdf, 2016.

[10] Bitcoin historical fee chart. https://bitinfocharts.com/comparison/bitcoin-median_transcation_fee.html.

[11] Y. Sompolinsky and A. Zohar, Secure High-Rate Transaction Processing in Bitcoin, pp. 507{527, Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

[12] The Raiden Network. https://raiden.network/, 2019.

[13] Seres, I.A., Gulya, L., Nagy, D.A. and Burcsi, P., 2020. Topological analysis of bitcoin's lightning network. In Mathematical Research for Blockchain Economy (pp. 1-12). Springer, Cham.

[14] Martinazzi, S., 2019. The evolution of Lightning Network's Topology during its rst year and the inuence over its core values. arXiv preprint arXiv:1902.07307.

[15] Ripple. https://ripple.com/, 2019.

[16] Ripple. https://1ml.com/statistics.

[17] Lightning Network Daemon. https://github.com/lightningnetwork/lnd.

[18] Sivaraman, V., Venkatakrishnan, S.B., Alizadeh, M., Fanti, G. and Viswanath, P., 2018. Routing cryptocurrency with the spider network. arXiv preprint arXiv:1809.05088.

[19] Wang, P., Xu, H., Jin, X. and Wang, T., 2019. Flash: ecient dynamic routing for ochain networks. arXiv preprint arXiv:1902.05260.

[20] Erdin, E., Cebe, M., Akkaya, K., Solak, S., Bulut, E. and Uluagac, S., 2020. A Bitcoin Payment Network with Reduced Transaction Fees and Conrmation Times. Computer Networks, p.107098.

[21] https://explorer.acinq.co/

[22] Khalil, R. and Gervais, A., 2017, October. Revive: Rebalancing o-blockchain payment networks. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (pp. 439-453). ACM.

[23] Prihodko, P., Zhigulin, S., Sahno, M., Ostrovskiy, A. and Osuntokun, O., 2016. Flare: An approach to routing in lightning network. White Paper.

[24] Malavolta, G., Moreno-Sanchez, P., Kate, A. and Maei, M., 2017, February. SilentWhispers: Enforcing Security and Privacy in Decentralized Credit Networks. In NDSS.

[25] Roos, S., Moreno-Sanchez, P., Kate, A. and Goldberg, I., 2017. Settling payments fast and private: Ecient decentralized routing for path-based transactions. arXiv preprint arXiv:1709.05748.

[26] diar.co, \Lightning Strikes, But Select Hubs Dominate Network Funds, "https://diar.co/volume-2-issue-25, June 2018

[27] Ford, L. R., Fulkerson, D. R., 2009. Maximal ow through a network. In Classic papers in combinatorics (pp. 243-248). Birkhauser Boston.

[28] Zhang Y, Yang D, Xue G. Cheapay: An optimal algorithm for fee minimization in blockchain-based payment channel networks. In ICC 2019-2019 IEEE International Conference on Communications (ICC) 2019 May 20 (pp. 1-6). IEEE.

[29] Yu R, Xue G, Kilari VT, Yang D, Tang J. Coinexpress: A fast payment routing mechanism in blockchain-based payment channel networks. In2018 27th International Conference on Computer Communication and Networks (ICCCN) 2018 Jul 30 (pp. 1-9). IEEE.

[30] Piatkivskyi D, Nowostawski M. Split payments in payment networks. In-Data Privacy Management, Cryptocurrencies and Blockchain Technology 2018 Sep 6 (pp. 67-75). Springer, Cham.