Splitting Payments to Increase Blockchain Effectiveness

Mălina Bulină , Dr. Stefanie Roos , Oguzhan Ersoy
TU Delft

Abstract

Blockchain technology is the underlying mechanism that many cryptocurrencies operate on. It relies on cryptographic techniques that enforce integrity on transaction records. The records (blocks) stored are limited in size and frequency. One well-known issue regarding blockchain technology is the lack of scalability. In order to mitigate this problem, payment channels were introduced. These are considered an ”off-chain” solution as communication with the blockchain is not required for executing transactions. Nonetheless, payment channels are debit-based, thus each channel that contributes to forwarding the payment is required to have a sufficient amount of coins to route the money. If this is not the case, the transaction may not succeed. For solving this matter, the node may decide to split the payment and forward it through multiple intermediaries until the destination is reached. This paper studies how fee models can be integrated into splitting protocols. An overview of the current state of the technology is provided, followed by a proposed solution for integrating fees into splitting protocols. It has been observed that splitting the payments and charging lower fees leads to a higher number of successful transactions, an outcome that was expected. The proposed fee model relies on the payment value and a statistic about the distribution of coins owned by the involved parties. The highest success ratio was achieved when combining the proposed design with the SplitIfNecessary splitting protocol. Nonetheless, when using the same fee model, there does not seem to be any pattern between the splitting protocol and the transaction value.

1 Introduction

1.1 Blockchain Technology

In the fourth quarter of 2008, the Bitcoin whitepaper [1] was published on a mailing list consisting of people interested in cryptocurrencies, only to be later released in 2009 to the public. The paper proposed a system for electronic transactions that did not rely on the need of a trusted third party; Bitcoin transactions were executed on the peer-to-peer Bitcoin network, verified by network nodes, and stored in a distributed ledger, namely blockchain.

During the last decade, blockchain technology has experienced a growing interest due to its low operational costs, transparency, immutability, and security. It is used in healthcare, telecommunications, cryptocurrencies, preventing electoral fraud, and other industries.

Blockchain technology enables multiple parties to collaborate without the need for a central authority. It is one type of distributed ledger technology ”maintaining a continuously growing list of ordered records” [2] in which the transactions are recorded using an immutable cryptographic signature. By using immutable signatures it is ensured that no attacker can derive a valid signature and intrude into the system or alter the data saved to the block [3]. Blockchain can be viewed as a series of blocks that are linked to each other, forming a chain. Once a new block is appended to the chain, the transactions contained in it cannot be tampered [4].

One major drawback of blockchain technology is scalability. By 2019, payment systems such as Visa were able to handle on average 1700 transactions per second [5; 6], whereas the blockchain for the Bitcoin network could only hold tens of transactions per second [7]. The Bitcoin's network block time is estimated to be around 10 minutes [8]. The scalability problem is an ongoing issue that is still affecting the current state of the network.

1.2 Payment Channel Networks

In order to solve this matter, layer 2 solutions were introduced. These solutions are intended to accelerate transaction processing times and decrease the associated costs by using payment channels as means of executing transfers [9]. Blockchain is considered to be situated in the first layer; interaction with the first layer takes place only during the creation and the closure of the channels.

Whenever a transaction is to be completed in the blockchain, the involved parties make use of a payment channel. Within this channel, an arbitrary number of transactions can be executed without affecting the speed of the network: the channel does not require the transactions to be recorded in the blockchain. The channel can be set up by depositing some amount of coins in it and it may be closed by either of the involved parties. As these updates do not take place in the blockchain (thus, they are ”off-chain”), their execution is almost immediate [10].

When there is no existing channel between two parties, the amount to be sent can be routed through intermediaries. Nonetheless, the channels are debit-based, thus the maximal payment amount depends on the deposit made on the channel. Debit-based channels imply that the coins are sent in a backwards manner: first by the nodes closest to the destination node (and received by it), followed by the same nodes collecting the offered money from their neighbours; this process continues until the source node is reached. In that situation, the source finally pays its neighbours the money forwarded. Thus, the initiating node is the last one to pay. The retrieval of the previously forwarded money happens only after showing a proof of the payment. One such payment channel network that implements the aforementioned characteristics is the Lightning Network [11].

In the situation that the intermediaries do not have sufficient funds for forwarding the payment through one channel, the total amount is split and routed through multiple channels that have a lower deposit. The channels selection procedure depends on the algorithm implemented internally. This matter is discussed in subsection 3.1.

Transaction Example

An example of a payment channel network can be seen in Figure 1. The yellow nodes represent the parties participating in routing the coins, whereas the bidirectional edges illustrate the channels between the nodes. The numbers on the edges correspond to the available funds that each party can send on the respective channel (e.g.:Charlie can send at most 30 coins to Dave and at most 20 coins to Alice). If Charlie sends Dave 30 coins, the balance of the channel used would have 30 coins subtracted.

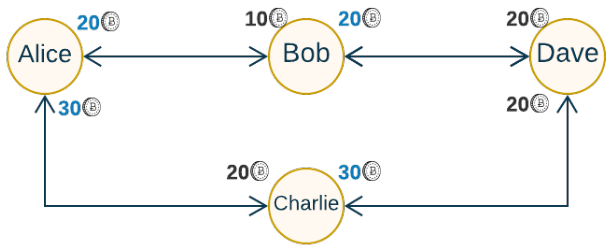

Figure 1: Example network of payment channels. Every edge has two corresponding numbers which represent the available funds on each end of the channel.

If Alice(the source) were to send Dave(the receiver) 50 coins, she could have her payment split as follows: 20 coins would go through Bob(the intermediary) and the rest of 30 through Charlie(the other intermediary). The course of the transaction works as follows: Dave generates a random number and gives the hash of that number to Alice. The source then uses her channel with Bob to send him the hash she received, together with an extra condition: in order for Bob to claim his part of the coins (that he first has to forward), he must provide a share of the preimage of the hash $h_D := H(x_D)$ before the expiration time of the contract. The preimage of h_D represents the initial number used by Dave. This is only known by the receiver and thus Bob cannot get his money back before Dave received his payment.

The same forwarding process happens with all the other intermediaries (Charlie in the described scenario). This conditional payment that has to be fulfilled in a predetermined amount of time is known as Hash-Time-Locked-Contracts (HTLC) [12; 11]. HTLC is used in one-path payments, such as in the Lightning Network, whereas in split payments the protocol was extended to allow forwarding over multiple paths [13]. In the Atomic Multi-Path (AMP) payment protocol the preimage is split into preimage shares and communicated to the designated neighbours.

Then, Bob uses his payment channel with Dave to pay him the 20 coins requested. He also adds the condition that Alice put in her request to Bob. Dave has a share of the preimage of h_D and he may claim his payment by giving Bob that share. After Dave sends Bob the number, the receiver collects his payment and Bob gets the preimage share [14].

One important thing to note is that the money collected by Dave belongs to Bob, thus Alice did not pay anything yet. As soon as Bob has his part of the preimage, he may present it to Alice and only then will she send him the 20 coins back. The exact same process that was presented above is followed by Charlie and the 30 coins forwarded through him. This procedure, of first requesting to send the money and only after offering the actual payment, represents an impediment in routing the funds, as the nodes may not have sufficient balances on their channels. This is the main reason for splitting the money through multiple nodes.

Payment Routing

Payment routing can be done in several ways. This paper describes two of them, more specifically global routing and local routing. Global routing is a three-phase protocol, whereas local routing consists of two phases.

The process for global routing works as follows. As the route is computed in the beginning, the sender knows the path to be taken by the payment with regard to the topology and network parameters (e.g. channel capacities) at the time of computing. This is followed by the establishment of the conditional payments, then by the actual transfer of the money. The channel topology and channel capacities are known by the initiator of the payment, but the specific balances of the channels are not. Channel capacities represent the total amount of funds available on the channel, whereas the balances indicate the amount of funds that each node may send (i.e. coins owned by a party).

Local routing, on the other hand, does not require the source node to compute the whole path to the destination. Each node may decide by itself how to forward the payment and whether to split the funds or not. More specifically, the process of the payment transfer works as follows: the source looks at the nodes it is directly connected to (neighbours) and decides (based on the internal splitting algorithm) if and how to divide the money through the other parties. Once the selected nodes are informed about the decision, it is their turn to do the exact same procedure of splitting and forwarding as done by the source. This process will be finished when the forwarding operation reaches the beneficiary of the payment. Two protocols that implement the previously described characteristics are Interdimensional SpeedyMurmurs [15] and ETHNA [16]. One important thing to note is that ETHNA is a non-atomic splitting protocol, thus it allows partial payments (the transaction succeeds despite the receiver not claiming the whole payment).

By implementing fees into payment channels, parties may be incentivised to help in routing the payment. It is expected to have more successful transactions if the number of nodes in the network is higher. Nonetheless, if the fees received by the intermediaries are too large, the transaction may fail because of some channel not having enough funds to route the payment. As local routing is used for computing the path, the fee paid by the source cannot be calculated based on the channel balances used for reaching the destination because these are not known beforehand. This paper proposes a solution that uses local routing for combining splitting methods with fee models. The designed protocol does not allow partial payments; the transaction is considered to be successful only if the whole payment is received by the destination.

The structure of the paper is as follows. In section 2 the methodology is presented, followed by the problem analysis and the protocol design in section 3. Section 4 describes the simulation scenario, along with the results obtained. The ethical aspects are discussed in section 5; section 6 reviews the results obtained, followed by ideas that could be used for further development in this area.

2 Methodology

In order to get a better understanding of the blockchain technology, the literature available has been reviewed. Based on the information collected, a routing algorithm that included splitting and fees was extended from an already existing simulation [17]. The extended protocol is publicly available [18]. The initially provided implementation allows to evaluate the behaviour of the network under various settings, including different splitting methods and fee models. IntelliJ IDEA was chosen as the IDE for developing the project and Python was used for plotting the results. As the simulator has been already written in Java, this programming language was further used for extending the implementation. This simulation has been conducted in order to integrate fee models with splitting protocols and assess their performance.

2.1 Routing

Split payments can be done either atomically or non-atomically. Atomic payments are successful if all the sub-payments reach the beneficiary; non-atomic payments succeed even though not the whole payment is received by the destination node. Piatkivskyi and Nowostawski [19] compared the two protocols and observed improvements in terms of the total liquidity of the payment network, but also suggested promising hints regarding how the network configuration may be more efficient (which need to be verified in a more rigorous manner).

Prihodko et al [20] discussed in-depth numerous routing algorithms including Global Beacons and Local Beacons. Another proposed solution [21] was to queue up transactions whenever a channel had insufficient funds available and use a congestion control protocol.

2.2 Fee Model

At any time t, payment channel network can be represented by a graph $G_t = (V_t, E_t)$, where V_t is the set of vertices (nodes in the network) and E_t represents the edges (payment channels), with $E_t \in V_t \times V_t$. Every edge u_{AB} with $A, B \in V_t$ has at time t balance $b(t)_{AB}$; it is bidirectional, thus $u_{AB}, u_{BA} \in E_t$; each node can use its directly connected edges to transfer payments through that channel if it has enough money to effectuate the transaction and if the channel has enough balance.

Consider again the example from section 1.2 in Figure 1. As previously described, when Alice(A) sends Charlie(C) 30 coins, the balance on the channel u_{AC} becomes $b_{AC} - 30$. It can be observed that once the balances of the channels are lower, the number of successful transactions is expected to decrease as well. Studies by van Engelshoven and Roos [22] take into account the reference point $(ref(u) = \frac{b_{AC}+b_{CA}}{2})$, to encourage transactions that have a positive effect on the balances of the channels. Other studies by Di Stasi et al [23], focusing on maintaining the channels balanced, propose a fee function that consists of applying a fixed charge and a variable part that takes into account the balance b. The imbalance of a channel $A \leftrightarrow C$ is defined as $|b_{AC} - b_{CA}|$. Low values imply that the channel is considered balanced. This model aims to avoid making payments using imbalanced channels, while also keeping the fees low.

2.3 Evaluation

With the aim of evaluating the behaviour of the implementation, two types of assessments were done: the first one implied observing the performance of the protocol using a small network topology having less than fifteen nodes, whereas the second one worked with snapshots of the Lightning Network. These files are publicly available [17; 18]. The methods would run the same transactions multiple times, save the results and compute statistics for assessing the performance of the network. The metrics used were: the number of successful payments, the average length of the path, and the median, average, minimum, maximum, standard deviation, and bounds of the 95% confidence interval of the fees during the runs. Moreover, for every transaction, the fees charged by the intermediaries and the fee offered by the source were logged. These values were represented using scatter plots and box plots. The plots are discussed in section 4.

3 Protocol Design

As highlighted in subsection 2.2, a payment channel network can be represented by a graph G_t at any time t. V_t is the set of nodes, E_t is the set of edges and for each edge u_{AB} the balance in one direction at time t is represented as $b(t)_{AB}$. For integrating fees in payment channels, there are two matters that need to be addressed: the splitting protocol and the fee model.

The following implementation concerns transactions executed using local routing as specified in section 1.2. More specifically, the routing algorithm works as follows: the sender decides to make a payment, but it does not know what the route of the coins will be. Thus, the fee is calculated based on the payment value and the channel balances to its neighbours. Afterwards, for a predetermined amount of trials, the path from the source node to the destination is computed. The payment value is split over the neighbours, according to the internal splitting algorithm (discussed in the following subsection). Then, the selected intermediaries, that have to forward the coins, compute the fee they receive. These values are recorded. If, at any point in time, it happens that the total fee received by the intermediaries is higher than the fee offered by the source, the transaction fails for this trial. After the intermediaries compute their corresponding fee, the split payment amount is forwarded to the following neighbours. This process happens until the destination node is reached. Throughout the forwarding process, if more paths lead to the same node, the total amount of funds is merged at that specific node. The protocol succeeds if the sum of the fees received by the intermediaries is lower than or equal to the total fee offered by the source and if the whole payment arrives at the destination. In the event that there are any leftover coins from the initially computed fee, these will be claimed by the destination.

The fee paid by the source for executing the transaction cannot depend on the path to be taken or on the network topology. Despite the fact that every node can see the channel capacities of the whole network, none of them knows which channels will be used for the payment.

Studies have taken into account networks having the balances of the edges distributed exponentially or normally [15; 22]. If the balances would form an exponential distribution (and they would be used for calculating the fees), there could be significant differences between the fees computed for using specific channels and the actual balances on the channels. This happens mainly because of having some channels that have really high balances and the rest having low balances. Thus, the remaining factors that could be used for computing the fee are: the value of the payment, the traffic of the network and the local information about the neighbouring nodes (which includes the number of outgoing edges, the channel balances, etc).

3.1 Splitting Protocol

A party may decide whether to split the payment or not, based on the total amount to be routed, its local view of the network, and the available balances on the outgoing channels. By not splitting the payment, the total amount of coins are forwarded only through one edge, from a node to another; thus, the balance of that channel may be significantly affected and may even remain depleted. This could lead to having a lower number of successful transactions in the near future. In some cases, such as the one previously described, the node may decide to split the payment into several sub-payments and forward them through multiple channels.

Some of the splitting methods studied in this paper are: SplitClosest which splits using the nodes that are the closest to the receiver of the payment, SplitIfNecessary that uses the neighbours with the highest balances in order to split as few times as possible. SplitIfTooHigh splits if too many funds are used by the payment, RandomSplit splits a payment value randomly between the neighbours that are closer to the destination, and ClosestNeighbour does not split; it forwards to the neighbour closest to destination if it has sufficient funds.

3.2 Fee Implementation

The fee for using a channel may be defined as a function $fee_t : E_t \times \mathbb{R} \times \mathbb{I} \to \mathbb{R}$. That is, for every channel at time t, the fee represents a value from the set of the real numbers that depends on the balance available on the channel, the payment amount and, the number of hops.

The fee function may be divided into two subproblems: the fee paid by the source and the fees earned by the intermediaries. For each of these fees, the following properties are taken into account: the amount of money forwarded, the current number of hops and the channel balances in that specific direction. Let $f_{AB}(p, h, t)$ represent the fee for using the channel u_{AB} from node A to node B; p is the payment amount that is routed and h is the current hop number (e.g.: if A were the source, $h = 1$ at node B) at time t.
$f_{AB}(p, h, t)$ should satisfy the following conditions:

$$f_{AB}(p, h, t) \geq 0, \ \forall A, B \in V_t \tag{1}$$

$$\sum_{\substack{K \in N \\ N \subset V_t}} f_{AK}(p, h, t) \geq \sum_{\substack{I \in M \\ M \subset V_t}} \sum_{\substack{J \in M \\ M \subset V_t}} f_{IJ}(p, h, t) \tag{2}$$

where A is the source, N is the set of selected neighbours of A and M is the set of all the nodes that contributed to forwarding the payment (including the source and the destination).

In Equation 1 it is stated that the fees should always be positive. If an intermediary receives a negative fee, it would have to pay in order to forward someone else's payment which does not seem to be a reasonable judgement. Equation 2 refers to the fact that the total fee, paid by the source (A), that is split between its neighbours, should always be higher than the total sum of the fees received by the intermediaries (that is, all the nodes I, J on the paths used for reaching the destination from the source node). Should this not be the case, then the intermediaries might not receive the promised payment or even subtract from the payment destined for the receiving node. Due to the fact that the full path of the payment will be known only when the destination is reached, it is not possible to divide the fee equally between nodes and thus, in some cases, there might remain leftover coins from the fee. In this situation, the destination node will also claim the leftover funds.

```
getFee(A, B, paymentValue, hops, balances, time) {
      sd = standardDeviation(balances)
      if A == source {
            // normalize sd between 0.6 and 0.8
            sd = normalize(sd, 0.6, 0.8)
            return paymentValue * sd * feeRate + baseRate
      }
      //if intermediary
      else {
            // normalize sd between 0.35 and 0.55
            sd = normalize(sd, 0.35, 0.55)
            sd /= hops + 1
            return paymentValue * sd * feeRate
      }
}
```

Figure 2: Pseudocode for the proposed fee model.

The proposed fee model is illustrated in the pseudocode from Figure 2. For each of the two cases, when the source pays the fee and when the intermediaries receive the fee, the standard deviation is normalized between different values. This is done in order to ensure that there are fewer cases in which the source did not send a fee high enough to be able to pay all the intermediaries. The values chosen for the normalization intervals have been adjusted in order to have similar fee values as the ones estimated to be used in the Lightning Protocol [24]. Nonetheless, in the Lightning Protocol, every node may choose what base rate and fee rate to use. Therefore, the values should be taken just as an estimate.

By using the standard deviation, nodes that have channels with balances that vary by a large amount receive higher fees. It is more probable for a node to have some channels with high balances if the standard deviation is big. This way, nodes are encouraged to have channels with a greater amount of funds deposited. Nonetheless, there might also be nodes having channels with a low standard deviation and still have high balances (when all the channels have similar high values). The number of hops is taken into account, particularly for the fact that if there are too many nodes contributing to the payment, the fee offered by the source may not be high enough. Through dividing the standard deviation by the number of hops, the fee received by the intermediaries is reduced by a small amount. The average number of hops using the settings described in section 2 was between 2 and 3, therefore the difference in the fees received by the nodes in the first hop do not differ by a great amount compared to the fees received by the nodes in the second and third hops.

4 Experimental Setup and Results

The protocol works as follows: the source initiates the payment, computes the fee to be paid (as described in subsection 3.2), then it decides on how to split the money, based on its current local view of the neighbours. The splitting protocols used were SplitClosest, SplitIfNecessary and ClosestNeighbour, which are discussed more in-depth in subsection 3.1. As the nodes are able to see the channel balances to which they are directly connected to, these values were saved. Then, for each set of the selected neighbours, the standard deviation of the channel balances was computed and normalised. Based on this value and on the number of hops, the fee received by the intermediaries was calculated.

When the source decides on the payment value, it has to pay a specific fee. This fee is equal to the payment amount multiplied by the normalised value and the fee rate, to which a base rate is added. For the intermediaries, the fee is equal to the payment amount multiplied by the normalised value and the fee rate. One thing to note is that the bounds for the normalisation are different for the two cases. The pseudocode for computing the fees is shown in Figure 2.

For the figures presented, the following settings were used: the initial channel capacities had an average value of 200 satoshi (sat) and were exponentially distributed; the average values of the transactions are specified for each figure; there were a total of 100 transactions. The capacities were not dynamically adjusted and there was no limit on the length of the path (even though, as previously specified, the average number of hops in a path was around 2-3). The fee model used is the proposed design (subsection 3.2), having a fee rate of 0.3 and a base rate of 1.

Figure 3 depicts a box plot for the proposed fee implementation. The values represent the fees received by the intermediaries. The chosen splitting algorithm was SplitClosest and the average transaction value was 90 sat. The median fee received by an intermediary was around 0.7 sat and the quartiles are relatively close to the median value (0.2 sat - 1.8 sat); the maximum fee received was around 4.1 sat, whereas the minimum fee was really close to 0 sat. The highest transaction had a value of around 400-500 sat. The maximum fee value of 4.1 sat is expected to be claimed by a node that had a big standard deviation of the channels. By using the standard deviation of the balances of the outgoing edges, the intermediary will have to pay a higher fee if the deviation is higher, whereas if the deviation is lower the fee would be reduced as well.

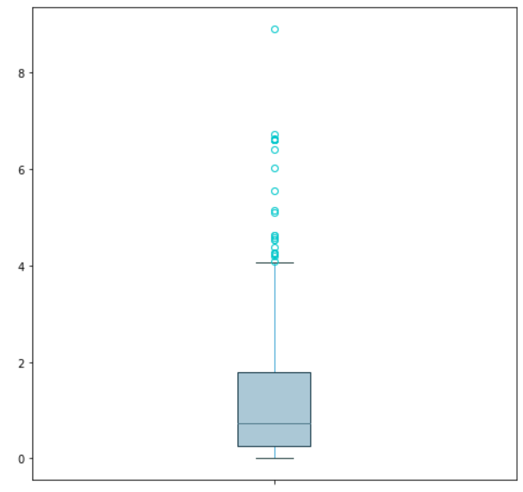

Figure 3: Box plot for the fees received by the intermediaries. Average transaction value of 90 sat.

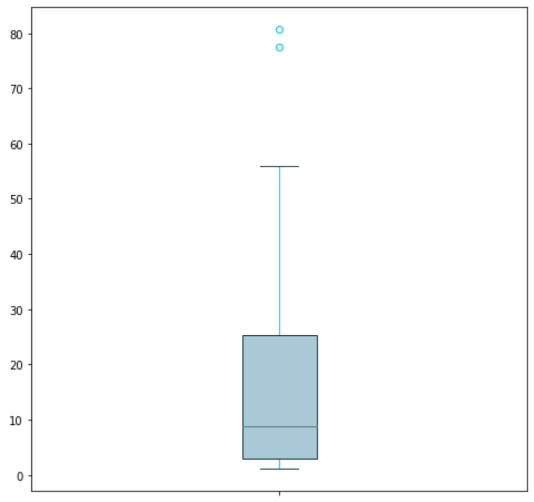

Figure 4: Box plot for the fees offered by the source. Average transaction value of 105 sat. Splitting protocol used was SplitIfNecessary.

Figure 4 shows the box plot for the fees that were paid by the source. One important thing to note is that the plot also contains transactions that failed (the outliers with values of around 75-80 sat). The average transaction value was 110 sat; the splitting algorithm used was SplitIfNecessary. The median fee offered by the source was of 8 sat. The 25th and 75th percentiles are 3 sat and 26 sat. It can be seen from the shape of the box plot that most of the values lay below the maximum value of 56 sat. The majority of the fees offered by the source node have low values (with regard to the transaction value). For instance, under the circumstances previously described, for a transaction of 201 sat, the final fee (paid by the source) was 5.75 sat, whereas for a transction of 1.69 sat the final fee was 0.29 sat.

Figure 5 illustrates the scatter plot for the fee value claimed by the intermediaries (x-axis) and the corresponding transaction value (y-axis). As specified, the fee rate for the proposed algorithm was 0.3 and the base rate 1. The splitting algorithms used for computing the plot were SplitClosest, SplitIfNecessary, and ClosestNeighbour. The light blue dots represent the successful transactions, whereas the **x** symbols in dark blue show the transactions that failed. The reason for a transaction to fail could be either the fee offered by the source was too low to pay all the intermediaries, the splitting protocol could not find nodes through which to route the payment or the whole payment did not arrive at the destination (not all the sub-payments succeeded). The blue crosses represent the average, median, the first, and third quartiles of the distribution. It can be seen that the majority of the transactions succeeded; most of the failing payments happen when the amount to be paid is too high.

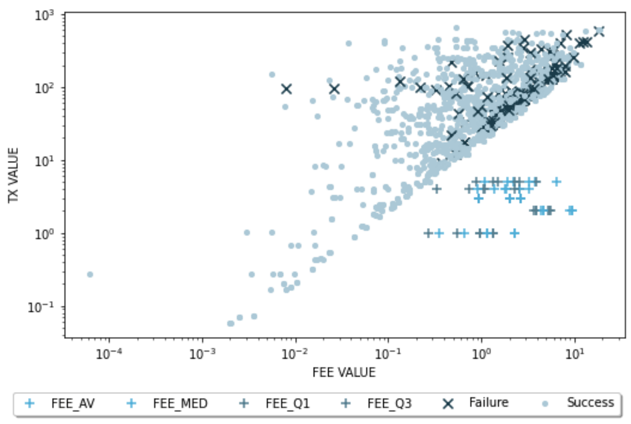

Figure 5: Scatter plot with the fee values on the x-axis and the transaction values on the y-axis. The light blue circles represent the successful transactions and the blue **x** symbols show the unsuccessful ones. The other points illustrated by crosses are the average, median and quartiles of the fees.

Figure 6 depicts both the proposed fee model and the Lightning fee model. The base rate for the Lightning model was set to 1 and the fee rate to 10^{-6}. The same routing algorithm was used for both in order to observe only the differences in the fee value. The x-axis shows the transaction value and the y-axis illustrates the success ratio. The success ratio represents the fraction of successful payments. For each of the plotted data points, the cross symbols show the results using the ClosestNeighbour splitting algorithm, the **x** symbols show the results with SplitIfNecessary and the dots show the results when SplitClosest was used. From the plot, it is clear that the proposed model has a higher success ratio, compared to the Lightning fee model. This is caused by the fact that the fees in Lightning were significantly higher compared to the proposed model. Thus, it is not reasonable to deduce that the designed fee model has better results. For 100 transactions with an average value of 105 sat, the fees offered to the intermediaries in the Lightning model were in the majority of cases 1 sat. In the proposed model, the fees claimed ranged from 0.01 sat up to 13 sat depending on the forwarded payment amount.

It can be observed in Figure 6 that, for transactions with a value higher than 95 sat, the proposed fee model has the highest success ratio when the SplitIfNecessary protocol is used. For the Lightning fee model, this statement can also be claimed for most of the cases. SplitIfNecessary appears to have the highest success ratio when chosen as the internal splitting protocol, whereas the ClosestNeighbour algorithm seems to yield the lowest success rate for the majority of cases.

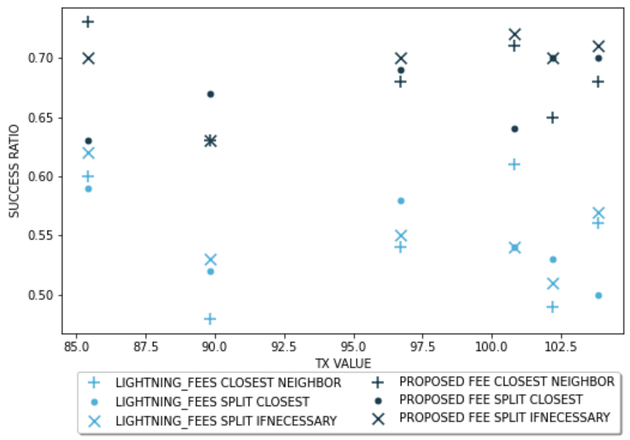

Figure 6: Plot representing the fraction of successful payments. The x-axis shows transaction value and the y-axis depicts the success ratio. The lightning protocol is highlighted with light blue, whereas the proposed fee model is in dark blue.

5 Responsible Research

With the purpose of conducting this study, no personal information was used. The experiments discussed in this paper can be reproduced using the extended implementation [18]. As previously specified, the proposed fee model had a base rate of 1 and a fee rate of 0.3. One important thing to note is that all the outliers plotted in section 4 have been included; no data points were omitted, despite the possibility of them belonging to an unsuccessful transaction. A snapshot of the Lightning Network was provided for evaluating the performance of the design. It is publicly available at [17]. The information contained in the files included the public key of the node, the alias, and the Internet Protocol (IP) address.

It is important to mention that, by using payment channels, the number of interactions with the first layer (blockchain) is reduced. Thus, by supporting the usage of payment channels, there might be a decrease in the number of transactions that need to be validated by the miners. As cryptocurrency mining represents a source of income for a variety of people, this may come out as an impediment. If the number of transactions recorded is significantly lower and the fees are reduced as well, mining will not be as profitable as it is considered now.

As the public key and IP addresses are known for every node in the network, the Blockchain technology is considered to be pseudonymous [25]. Some cryptocurrencies including Bitcoin are often viewed as a means of executing payments that are untraceable. This might incentisize users to carry out lawbreaking transactions without being traced. One of first large-scale criminal enterprises that allowed transactions using Bitcoin was Silk Road, an illegal marketplace that operated on The Onion Router (TOR) network [26]. In 2013, the website had been shut down and its assets were seized. Nonetheless, there still exist marketplaces supporting criminal activities and that have not yet been seized.

Despite the drawbacks of the use of this technology, blockchain may provide a decentralized framework for sharing electronic health records. It creates one ecosystem of patient data that can be referenced by hospitals, pharmacies and all the other parties involved in the treatment which can lead to faster diagnoses. This has already been implemented by a few companies that help healthcare industries store digital records. It is of great importance to educate the users and make them understand how this technology can improve the quality of the existing services instead of supporting the use of blockchain for criminal activities.

6 Discussion

As this research is concerned with blockchain technology and the way payment channel networks behave under different circumstances, various factors need to be taken into account.

First of all, the way fees are implemented into splitting protocols should not affect the performance of the network or any individual during the payment process. More specifically, other parties should not be affected when sending a payment by other nodes in the network which are also in the process of transferring funds. The splitting protocol should try to modify the balances on the channels as little as possible, in order to favour more successful transactions.

Secondly, by integrating fees into payment channels, the anonymity of the nodes contributing to routing the payment should be preserved. When a payment is forwarded by multiple nodes in the network, the parties involved should know as little as possible about each other. Nodes that are directly connected may be aware of the identity of each other, but this information should not be leaked. Moreover, the amount of funds rewarded to each contributing node should not be known by anyone, except for the party in cause.

Also, the splitting protocol should not favour any specific individual under any circumstances. That is, nodes may not be able to make any changes to the protocol or their identity in order to receive a higher fee, nor should the source be able to pay a smaller fee in order to have their payment routed. The fees should be computed solely based on the network parameters and any static settings, but not based on the specific individual contributing towards completing the transaction.

Moreover, the nodes involved should not be able to steal any of the funds to be routed. The payment amount and fees should be divided to each individual based on their contribution.

In the event that one of the parties involved decides that an error has been made, a dispute should be opened and propagated to the first layer (i.e. blockchain). From there on, the discussion should be settled by the miners in a trusted manner and the dishonest party should be discovered.

7 Conclusions and Future Work

This study illustrated how fees can be integrated with splitting protocols. Implementing an efficient fee model is not a trivial task; it has been observed that the routing algorithm, the splitting protocol, and the fee model significantly affect the success rate of transactions. It is important to find a combination between the studied algorithms in order to achieve a high success ratio.

The proposed fee model, that takes into account the standard deviation of the channel balances, seems to yield fees that cover a wide range of values (e.g. in one case the fees could go from 0.01 sat up to 13 sat). It has been noticed that there is an increase in the success ratio of transactions if the fees are reduced. This was an expected outcome, as the intermediaries have higher chances of forwarding the payment if the amount is lower. Deciding on the fee value paid by the source has a significant impact on the outcome of the transaction.

After several trials, it seems that the Lightning fee model yields the best results when combined with the SplitIfNecessary algorithm. For transactions having values lower than 120 sat, the designed model worked best when SplitClosest was used. Both SplitIfNecessary and ClosestNeighbour, combined with the studied model, resulted in 0.65 (and above) success ratio in most of the cases, no matter the transaction value. Nonetheless, no patterns, with regard to the transaction amount and the splitting protocol, were observed.

It apears to be more probable for a transaction to be successful and have lower fees if the route of the payment is known before computing the source fee. Although, in this case, the channel balances should be dynamically adjusted in order to avoid unexpected changes in the balances.

As a proposal for future research, more attention should be given to the routing algorithm. The fee model can depend on the way the path to the destination is computed, thus it is important to first decide how this will be done. Due to the fact that the proposed fee computation works for both cases of local and global routing, the algorithm can be extended to include other parameters. One candidate solution is presented in [20], where the route is computed using a dynamic approach. Each node has a list of designated beacons through which they can expand their awareness of the network. This way, the node has an updated view of the network by changing its internal routing table based on the information known by the selected beacon. Another proposed solution would be to change the fee model, such that each node may decide on its base rate and fee rate. These values could take into account, for each node, the number of payments that they helped forwarding, the average balance they have on the outgoing channels or even some statistic that quantifies the node's active and running time. This could be extended, in order to have a means of characterising each node's level or trustworthiness.

References

[1] Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System. *Cryptography Mailing list at https://metzdowd.com*, March 2009.

[2] Kurt Fanning and David P. Centers. Blockchain and Its Coming Impact on Financial Services. *Journal of Corporate Accounting & Finance*, 27(5):53–57, 2016. https://doi.org/10.1002/jcaf.22179.

[3] Kaushiki Srivastav. A Guide to Blockchain Immutability and Challenges - DZone Security, Jan 2021. https://dzone.com/articles/a-guide-to-blockchain-immutability-and-chief-chall.

[4] Zibin Zheng, Shaoan Xie, Hong-Ning Dai, Xiangping Chen, and Huaimin Wang. An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends. 2017. https://doi.org/10.1109/BigDataCongress.2017.85.

[5] VisaNet: Electronic Payments Network. https://usa.visa.com/about-visa/visanet.html.

[6] Kenny L. The Blockchain Scalability Problem & The Race for Visa-Like Transaction Speed. Towards Data Science, Jul 2019. https://towardsdatascience.com/the-blockchain-scalability-problem.

[7] Kyle Croman, Christian Decker, Ittay Eyal, Adem Efe Gencer, Ari Juels, Ahmed Kosba, Andrew Miller, Prateek Saxena, Elaine Shi, Emin Gün Sirer, Dawn Song, and Roger Wattenhofer. On Scaling Decentralized Blockchains (a position paper). pages 106–125, Jan 2016. https://doi.org/10.1007/978-3-662-53357-4_8.

[8] Ivo Stoepker, Rowel Gundlach, and Stella Kapodistria. Robustness Analysis of Bitcoin Confirmation Times.

SIGMETRICS Perform. Eval. Rev., 48(4):20–23, May 2021. https://doi.org/10.1145/3466826.3466834.

[9] Rakesh Sharma. Bitcoin's Lightning Network: 3 Possible Problems. Can Lightning Network Solve Bitcoin's Scalability Issues? https://www.investopedia.com/tech/bitcoin-lightning-network-problems, May 2021.

[10] S. Dziembowski and P. Kedzior. Non-Atomic Payment Splitting in Channel Networks. 2020.

[11] Joseph Poon and Thaddeus Dryja. The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments. Jan 2016.

[12] Majid Khabbazian, Tejaswi Nadahalli, and Roger Wattenhofer. Timelocked Bribing. 2020. https://eprint.iacr.org/2020/774.

[13] Olaoluwa Osuntokun. AMP: Atomic Multi-Path Payments Over Lightning, Feb 2018. https://lists.linuxfoundation.org/pipermail/lightning-dev/2018-February/000993.html.

[14] Bitcoin Wiki. Hash Time Locked Contracts. https://en.bitcoin.it/wiki/Hash_Time_Locked_Contracts.

[15] Lisa Eckey, Sebastian Faust, Kristina Hostáková, and Stefanie Roos. Splitting Payments Locally While Routing Interdimensionally. *IACR Cryptol. ePrint Arch.*, 2020:555, 2020. https://eprint.iacr.org/2020/555.

[16] Pawel Kedzior Stefan Dziembowski. Non-Atomic Payment Splitting in Channel Networks. 2020. https://eprint.iacr.org/2020/166.pdf.

[17] PaymentRouting. Initial Implemenation. https://github.com/stef-roos/PaymentRouting.

[18] PaymentRouting. Extended Implemenation. https://github.com/b-malina/PaymentRouting.

[19] Dmytro Piatkivskyi and Mariusz Nowostawski. *Split Payments in Payment Networks: ESORICS 2018 International Workshops, DPM 2018 and CBT 2018, Barcelona, Spain, September 6-7, 2018, Proceedings*, pages 67–75. September 2018. https://doi.org/10.1007/978-3-030-00305-0_5.

[20] Pavel Prihodko, S. Zhigulin, Mykola Sahno, A. Ostrovskiy, and O. Osuntokun. Flare : An approach to routing in lightning network white paper. 2016.

[21] Vibhaalakshmi Sivaraman, S. Venkatakrishnan, Kathleen Ruan, Parimarjan Negi, Lei Yang, Radhika Mittal, G. Fanti, and M. Alizadeh. High Throughput Cryptocurrency Routing in Payment Channel Networks. 2020. https://arxiv.org/pdf/1809.05088.

[22] Yuup van Engelshoven and Stefanie Roos. The Merchant: Avoiding Payment Channel Depletion through Incentives. *ArXiv*, abs/2012.10280, 2020. https://arxiv.org/pdf/2012.10280.pdf.

[23] Giovanni Stasi, Stefano Avallone, Roberto Canonico, and Giorgio Ventre. Routing Payments on the Lightning Network. pages 1161–1170, July 2018. https://doi.org/10.1109/Cybermatics_2018.2018.00209.

[24] Nida Khan and Radu State. *Lightning Network: A Comparative Review of Transaction Fees and Data Analysis*, pages 11–18. June 2019. https://doi.org/10.1007/978-3-030-23813-1_2.

[25] Virginia Corona. Bitcoin Anonymity and Blockchain, April 2021. https://scholarlyoa.com/bitcoin-anonymity-and-blockchain.

[26] Sesha Kethineni and Ying Cao. The Rise in Popularity of Cryptocurrency and Associated Criminal Activity. *International Criminal Justice Review*, 30(3):325–344, 2020. https://doi.org/10.1177/1057567719827051.