Increase of long-term success ratio using novel splitting methods in Interdimensional SpeedyMurmurs

Han Heijmans[1] , **Stefanie Roos**[1] , **Oğuzhan Ersoy**[1]
[1]TU Delft

Abstract

Popular cryptocurrencies lack scalability. Payment Channel Networks (PCN's) allow a large increase in transaction throughput. However, transaction failures are frequent when transactions are routed through multiple intermediaries. A previously introduced protocol, Interdimensional SpeedyMurmurs, increased the success ratio of transactions in PCN's by allowing routing to be more flexible and transactions to be split into multiple smaller transactions. This protocol also introduced several methods of splitting transactions that increased the success ratio even further in the short term. The splitting methods were not good for the short term as they often depleted channels completely. This paper aims to introduce novel methods of splitting transactions that easily integrate into Interdimensional SpeedyMurmurs in order to increase the success ratio in the long term. A thorough evaluation of the novel splitting methods was conducted using simulations on a snapshot of a real world used PCN. In one scenario an increase of 2.4% over previous methods was achieved.

1 Introduction

The most popular blockchains, like Bitcoin [5] and Ethereum [13], are limited in their scalability [12]. Even now, when cryptocurrency are not massively adopted, this limited transaction throughput results in high fees and long transaction times. Certain layer-two protocols, like Lightning [7] for Bitcoin, try to mitigate this problem by use of payment channel networks. Using such a payment channel network, users can exchange funds off-chain only using the blockchain for instantiating a payment channel, resolving disputes and closing a payment channel. This has the potential to drastically reduce the load on the blockchain.

Parties often do not want to instantiate too many payment channels due to the fact that it locks a certain amount of currency. In case two parties want to exchange funds whilst not having a direct payment channel between them, they can route the transaction through multiple intermediaries. These transactions fail frequently however, due to insufficient channel capacities and the routing being done entirely at the initial sending party, or source of the transaction, which has no knowledge of the channel balances of the intermediaries.

A protocol called Interdimensional SpeedyMurmurs (INTSM) increased the success ratio significantly. It does so by having each intermediary on the path decide what the next step will be and by using a improved routing algorithm, based on SpeedyMurmurs [8], which is more flexible allowing for more possible paths to be taken [2]. To increase the success ratio even further the transaction can be split over multiple paths using several splitting methods that were introduced with the protocol.

Other protocols exist that allow for multi-pathrouting [6; 9; 1]. They all have a different approach. One removes the requirement of atomicity, meaning that transactions still succeed if not all of the transaction value is received. And in a lot of protocols the path is fully decided by the source.

INTSM achieved a high success ratio in the short term. What we mean with short term is that after initializing the PCN in the simulator and performing a payment on this PCN, the PCN is reset before another payment is performed.

In this research we aim to introduce and evaluate novel splitting methods to increase the long term success ratio of transactions in payment channel networks, meaning after a payment is performed the effects of the payment remain on the PCN when performing every subsequent transaction.

The current splitting methods which were introduced with the INTSM protocol split ties, which occur frequently, randomly. Also they deplete a lot of channels and make them imbalanced which negatively affects the success ratio [2; 9; 11; 4]. The splitting methods in this paper improve upon these methods by splitting these ties more strategically. They showed an increase between 0.8% and 2.4% with respect to the already existing splitting methods in INTSM.

We will start with explaining how payment channel networks work and how the current protocol works on which we intent to build. Then we shall continue by going over some splitting methods that already exist and why the might or might not be beneficial for specifically increasing the long-term success ratio of transactions. After going over the existing splitting methods we shall go over the novel splitting methods. We will then evaluate these splitting methods to determine by how much the splitting methods increase the success ratio.

2 Background

2.1 Payment Channel Networks

Payment channel networks (PCN's), like Lightning [7], are protocols that mostly run off-chain while maintaining the guaranties that the underlying blockchain provides. A PCN is essentially a network of payment channels (PC's). On a PC parties could perform many transactions without broadcasting any of them to the blockchain. When two parties are looking to exchange funds without having a direct PC between them, they can route this payment through multiple intermediaries.

To create a PC between parties an initial transaction is required on the blockchain locking in a certain amount of cryptocurrency. The amount every party chooses to lock in constitutes the initial balances on the PC. Payments made on the PC can not exceed the balance of the sending party and changes the channel balance with a value equal to the transaction value. So if A sends v coins to B, B's balance will now be v higher and A's v lower. The capacity of a PC is all the balances of the parties added up. To change the balances all participating parties have to unanimously agree on the updated balances. Upon termination of a PC a transaction is broadcast to the blockchain containing the final balances.

In case of the Lightning network [7], when a payer and payee want to perform a transaction the payee, or receiver of the transaction, sends a hash of a randomly generated value to the sender. Upon receiving this hash the sender creates a Hash Time Locked Contract or HTCL to the next intermediary on the path. A HTCL is a conditional payment that gets performed upon showing the preimage of the hash. To create a HTCL the creator locks in the value of the payment and can therefore not use this for any other payments. When a final HTCL reaches the receiver, the receiver shows the preimage of the hash to redeem the funds from this HTCL and this preimage is now known to the creator of this HTCL. All intermediaries will do so likewise, using the preimage, until the final HTCL, from the original sender of the payment, is redeemed.

There are many PCN's that all work in a different way with regard to routing or how balances are updated on the channels [3]. When looking at routing, one usually distinguishes between two types of routing: *global routing* and *local routing*. With global routing the party sending funds decides the path the transaction will take through the PCN entirely. This party will therefore need to know the topology of the entire PCN. A downside of this approach is the fact that the sending party does not know if all the channels in the path have sufficient balance which in turn can result in failing transactions. Only the initial balance and the capacity of each channel is known as this is displayed on the blockchain due to the initial transaction needed for initializing the PC. Local routing protocols determine the path the transaction takes in a distributed fashion. Every intermediary node decides for themselves what the next step in path will be.

2.2 Interdimensional SpeedyMurmurs

Interdimensional SpeedyMurmurs (INTSM) [2] is a protocol that uses local routing. It uses a more flexible method of determining the next step compared to shortest path routing. It also allows for transactions to split while maintaining atomicity, meaning if one partial payment fails the entire payment will fail. Routing a transaction in this protocol happens in two steps: generating a set of candidate PC's and then allocating the transaction value over these candidates (splitting).

When generating a set of candidate channels, some measure of closeness is needed so that the transaction moves towards the receiver. Also loops have to be prevented for obvious reasons. Due to these facts, when considering a channel to be placed in the candidate set, the node at the other side of the PC needs to be closer to the destination than the current node and must not have been visited before. INTSM's distance measure generates several spanning trees from the PCN, the amount of which is called the dimensionality of INTSM. If the node connected on the other side of the PC is closer in at least one of these spanning trees, the PC is added to the set of candidate channels. Another distance measure (which is also used in Lightning [7]) that was evaluated was the hop distance which indicates the least amount of hops towards the destination.

After a set of candidate channels is created, the protocol can choose how to allocate the transaction value over these channels. In order to do so several methods of splitting the transaction were also introduced and evaluated in INTSM.

- **No Split**: As the name suggests, it does not split the transactions. It considers the candidate channels in increasing order of closeness, splitting ties randomly, and selects the first channel with enough balance. If no such channel exists the transaction fails.

- **Split By Distance (Dist)**: Like No Split it considers the channels in increasing order of closeness. This method however, allocates a part of the transaction value to the channel even when the balance is not high enough to accommodate the total transaction value. It continues to do so until either no candidate channels are left, which means there is not enough balance to forward the payment and the transaction fails, or until all the transaction value has been allocated.

- **Split If Necessary (IfN)**: In case there exists a channel that has enough balance this method corresponds to No Split. When no such channel exists however, candidate channels are considered in decreasing balance. Like Dist it allocates the transaction value until either the transaction fails as the total balance is not high enough, or all the transaction value has been allocated.

After reevaluating these splitting methods with the same parameters in INTSM, IfN performed the best in the long term with a average success ratio, taken from the last one hundred thousand transactions, of around 64 percent. Dist performed better at the initial few hundred thousand transactions and ended up with dipping slightly lower than IfN with an average success ratio of around 63 percent as seen in figure 1. According to the INTSM paper this could be the case due to Dist depleting to many channels.

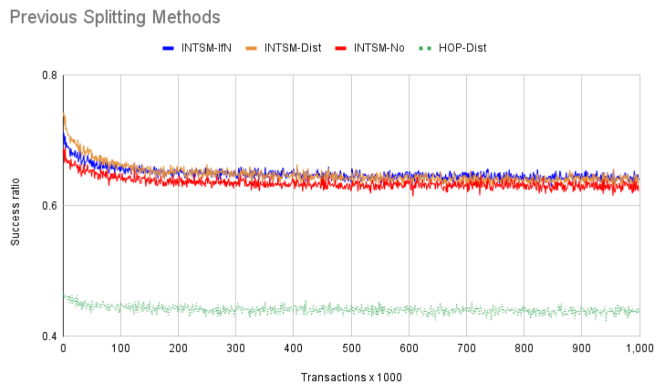

Figure 1: The three splitting methods evaluated with INTSM and Hop distance. Ran with one million transactions and INTSM dimensionality equalling 5

3 Novel Splitting Methods

There are a multitude of factors that are thought to influence the success ratio of transactions. Transaction size and channel capacities are the first indicators one thinks of, but also the amount of splitting and path length can be a heuristic indicating the success ratio.

Every partial payment can fail and if even one if these partial payment fails the entirety of the payment will fail due to the atomicity guarantee of the protocol. More partial payment can therefore negatively affect the success ratio. However, each of the partial payments themselves have a higher chance of succeeding as the size of the partial payments is strictly lower than that of the whole payment. Some balance in the amount of splitting is therefore necessary.

Channel balances change while transactions take place making them imbalanced or depleted. Having a balanced channel, meaning both directions of the channel have a similar balance, would allow for the highest success ratio under the assumption that both nodes are expected to send an equal amount of transactions and that the transaction sizes are skewed to the lower end.

Longer paths lock in more collateral due to the nature of payment channel networks. Each intermediary has to keep collateral locked until either the time runs out or the payment succeeds. When transactions keep coming to a node, more and more currency is locked as collateral effectively lowering the balance the node can use for payments coming in.

The splitting methods proposed in the INTSM paper considered closeness to the receiver and size of the balances of the directly connected channels. Also ties were broken randomly and since ties occur quite frequently in Split By Distance this seemed like a good place for improvement.

3.1 Tie Breaking Approach

Split By Distance often has ties. The candidate channels are considered by decreasing closeness and there are often channels that are equally close. In the current implementation these tied candidate channels are picked at random.

Several heuristics can be considered when determining how to split ties. Considering the largest balance first is the most obvious way that comes to mind, as it might lessen the amount of necessary splitting, but did seem to show an improvement as can bee seen in figure 5 in appendix A.

One of the most intuitive ways of retaining the success ratio in the long term is by keeping the channels balanced. When splitting ties to prefer channels such that the average imbalance of the entire network is as low as possible however the success ratio did not seem to increase in combination with INTSM. Figure 6 and 7 in appendix A show how the average network imbalance drops but the success ratio performed worse. This result was rather surprising and counter intuitive.

Looking ahead at the next nodes on the possible paths and using information on channel balances and capacities seemed more promising. Looking at the balances indicates the possibility of next nodes being capable of forwarding a payment. Also nodes which have more balance are more likely to also have more channels. More channels can allow for more flexible routing at these nodes. Looking at capacities can also be some indication of the aforementioned qualities but does so to a lesser extend as they do not get updated after transactions occur and do not accurately reflect the balances on the channel. We will now introduce two ways of breaking ties which use channel balances and capacities.

- **Split By Distance Look-ahead Capacity** (SDLC): SDLC breaks the ties by determining the total potential capacity of the other node connected to the PC. The channel with the most look-ahead capacity is the winner of the tiebreak. To elaborate further, when a tie occurs the algorithm will look ahead at all the nodes connected to the tied channels. For each of these nodes their own candidate channel set is first generated and the capacity of these channels are added up.
- **Split By Distance Look-ahead Balance** (SDLB): SDLB considers the balances of the connected nodes contrary to SDLC which only looked at the capacities. As mentioned before only the capacities and initial balances of the channels are publicly known. This approach requires a slight modification of this by allowing each node to also know the balances of their neighbours. This approach uses more useful information at the cost of some amount of privacy.

3.2 Breaking Ties in IfN

In case the IfN splitting method has a tie, the tie is also broken at random. Tied balances occur very infrequently however as they can have a large more continues range of values in comparison to distances. Therefore, thinking about improving this method by splitting ties seems less obvious and therefore a new way of determining if channels are tied is required.

IfN tries to split transactions only in the case it is necessary to do so as no channel with enough capacity to support the whole payment exists. It thereby minimizes the amount of splits greedily on every step of the routing. This does not mean that it will always require less splits than Dist however. By greedily minimizing the amount of splits IfN can use longer paths to reach the receiver that might end up requiring even more splits in the end.

By keeping the amount of splits as small as possible it can deplete less channels and creates less chances for a partial transaction to fail. This motivates our design to split as little as possible. We therefore consider channels to be tied if they can form a set with other channels such that the transaction can be performed and the size of the set is no larger than the minimal amount of channels required to perform the transaction. Now that we have a way of determine ties lets introduce the splitting algorithm.

Split If Necessary Distance (SND): Upon receiving the set of candidate channels the algorithm first determines the minimal amount of splits possible by considering the channels in decreasing balance and adding up the channels until the cumulative channel balances exceeds that of the transaction value.

Now that the minimal number of splits s is determined the algorithm will iterate over all possible channel subsets of size s and check if they have enough balance to perform the transaction. This could be a rather time intensive step as the time complexity of this step in the worst case is $O(2^n)$ where n is the amount of candidate channels. Depending on how many splits will be required in practice will therefore determine the usability of this approach. Out of these subsets the set with the minimal amount of cumulative distance, meaning the distance of each channel added up, is chosen. Now the transaction gets split over the channels in this set.

4 Evaluation and Results

This section covers the evaluations performed on the splitting methods in combination with the INTSM protocol and the INTSM and hop-distance (HOP) closeness measures. As indicated by the title of this research, our main metric we will consider is the success ratio of transactions. We define the success ratio as:

$$r = \frac{t_s}{t_t}$$

Where r is the success ratio and t_s and t_t the transactions which succeeded and the total amount of transactions respectively.

Since we are mainly looking to see the long term success ratio, we will only look at the last hundred thousand transactions for determining the increase in success ratio in comparison with the Dist and IfN methods.

4.1 Simulator

To perform the evaluations we used an a simulator called GTNA previously extended to include the INTSM protocol and now also with the novel splitting methods[1].

The sender and receiver of transactions are randomly chosen. Every node in the network has the same chance to be selected as a sender or receiver.

The simulator can perform both dynamic and static simulations. In static simulations each time a transaction is performed, the PCN is reset to its original state before the transaction took place. Dynamic simulations keep the changes to the PC balances for every subsequent transaction. We consider the static variant to be short term whereas the dynamic variant we also call long term. Dynamic simulations are more representative of the real world and these kinds of simulations are the focus of this evaluation.

Transactions are simulated one after the other, meaning a subsequent transaction only takes place after the previous one either succeeded or failed. This also entails that no concurrent transactions can take place. This could affect the measured success ratio by displaying a higher success ratio than would be the case in a current scenario. The HTCLs that are used each lock in a certain amount of currency which lowers the balance that can still be used for a certain period of time. These simulations without including concurrency are therefore less realistic but due to time constrains concurrent simulations where not performed and would therefore be interesting for future work.

4.2 Parameters

The success ratio of transactions largely depends on the parameters used in the evaluation: the transaction sizes, distribution of channel capacities and balances, the topology of the PCN and which algorithms were used for routing the transaction.

- **Transactions**: The transactions are randomly generated from an exponential distribution $T = Exp(\lambda)$. The impact of the transaction size on the success ratio fully depends on the sizes of the capacities. The effect of allowing transactions to be split becomes most apparent when transactions are relatively large since small transactions will usually not require splitting. In the evaluations $\frac{1}{\lambda} = \frac{\mathbf{E}[C]}{2}$ where C is the distribution and therefore $\mathbf{E}[\mathbf{C}]$ the expected value of the distribution. Basically the expected value of the transactions is half that of the capacities. This transaction size was somewhat arbitrary since no transaction data is available but we chose the size to highlight the impact for splitting. If transactions are relatively small compared to the channel balances splitting will occur more rarely.

- **Channel capacities and balances**: Like the transactions, the channel balances are generated from an exponential distribution $C = Exp(\lambda)$. Here lambda could really be anything as only the relative size to the transaction size really matters in this work but lets say $\frac{1}{\lambda} = 200$ which was also chosen in the INTSM paper. This is the average balance for each direction of a channel. The average capacity is therefore 400.
- **Topology**: The PCN topology used is a snapshot of the real-world PCN, the Lightning network [2]
- **Algorithms**: We evaluated both tie splitting protocols SDLC and SDLB in combination with the INTSM and HOP closeness measures. INTSM uses a certain amount of spanning trees also called the dimensions. For the evaluations the amount of dimensions was set to five as

[1]https://github.com/HanHeijmans/PaymentRouting

[2]https://gitlab.tu-berlin.de/rohrer/discharged-pc-data/blob/master/snapshots/lngraph_2020_04_01__01_00.json.zst

it was shown in previous work [2] that using more than 5 dimensions did not increase the success ratio by that much.

4.3 Results

The evaluations were ran twenty times (in case of SND fifty times as evaluating it was less time consuming) and averaged to eliminate noise in the generated data that existed due to the many random elements in our simulation.

- SDLC: In combination with the HOP measure SDLC start out with around 6% higher success ratio but eventually converges to HOP-Dist. This could be possible since the capacities start out as a good heuristic initially but as more and more transactions take place the underlying balances change and the capacity provides less useful information.

  When comparing SDLC with Dist in combination with INTSM, a slight increase is present and seems to reduce slightly but still remain above Dist with 1.4%. Also with respect to INTSM-IfN, SDLC shows an increase of around 0.8%. These results can be seen in figure 2 and has a standard deviation between runs of $4.7 * 10^{-5}$.

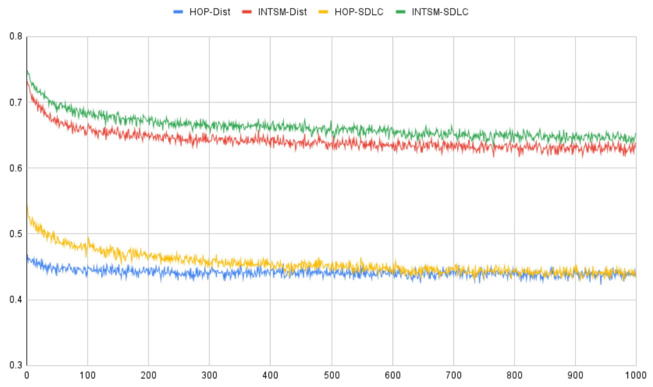

Figure 2: Evaluation of SDLC

- SDLB: As can be seen in figure 3, SDLB in combination with HOP gainst a significant increase of about 11.6% in total.
  When we look at SDLB and Dist with INTSM an increase of 2.4% was achieved. Also SDLB had a 1.8% higher success ratio than INTSM-IfN. Here the standard deviation between runs was $7.9 * 10^{-5}$

- SND: With respect to IfN, SND did not show any improvements when combined with HOP. With INTSM it did show an improvement over Dist and IfN of 1.3% and 0.7% respectively. The exact result can be seen in figure 4 with a standard deviation of $2.3 * 10^{-5}$.

5 Responsible Research

In order to allow for reproducibility of this research the code of the simulator together with the INTSM protocol and split-

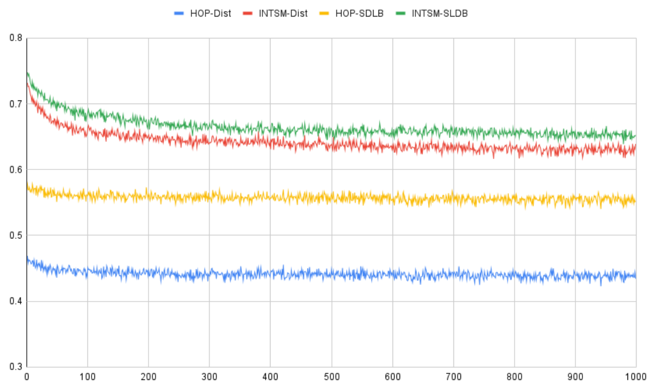

Figure 3: Evaluation of SDLB

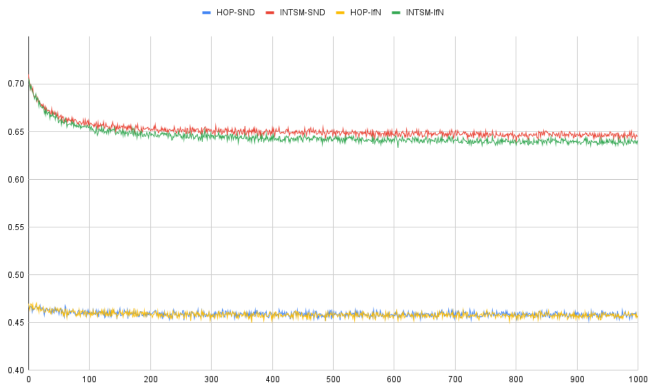

Figure 4: Evaluation of SND

ting methods is provided[3]. Since the algorithms and simulation contains some amount of randomness, the simulation requires multiple rounds to be performed and averaged over. The evaluation section contains the information needed to recreate the figures shown in this paper.

Split By Distance Look-ahead Balance, which is the method resulting in the highest increase of success ratio, requires extra knowledge of the balances which are unavailable in the current implementations of Lightning and INTSM. Sharing your balances to your neighbours reduces your privacy with regard to you current balance on every one of your channels. This allows certain parties with malicious intentions to get details on when transactions are performed and the size of these transactions. There is a kind of attack possible on the Lightning network to reveal channel balances called a Balance Discovery Attack which proves to be very effective [10]. A more pressing concern is the fact that this also affects the privacy of parties which are connected to the nodes you are connected to without those nodes being directly connected to you. Participants of the network therefore lose control over their privacy to a certain degree.

More general ethical considerations are with regard to the

[3]https://github.com/HanHeijmans/PaymentRouting

fact that the research slightly improves the success ratio of transactions within the INTSM protocol. This could indirectly effect the efficiency and usefulness of a cryptocurrency like Bitcoin. Cryptocurreny can make it harder for governments to control the currency for better or worse. Refunds can generally not be performed due to lack of a central trusted entity allowing parties with malicious intend to have more control over their unlawfully acquired currency. Cryptocurrency also enable people to not have to trust in central banks to keep their money safe and returns control to the individuals. In some countries where the its currency is in hyperinflation cryptocurrency also provide a good alternative.

6 Conclusions and Future Work

This paper introduced three novel splitting methods that can be directly combined with the Interdimensional Speedy-Murmers protocol. It was shown that an increase in long term success ratio was achieved over already existing methods.

Concurrency is not considered in the evaluations and further simulations including concurrency would better reflect the real world situation.

References

[1] Stefan Dziembowski. Non-atomic payment splitting in channel networks. 2020.

[2] Lisa Eckey, Sebastian Faust, Kristina Hostáková, and Stefanie Roos. Splitting payments locally while routing interdimensionally. Cryptology ePrint Archive, Report 2020/555, 2020. https://eprint.iacr.org/2020/555.

[3] Lewis Gudgeon, Pedro Moreno-Sanchez, Stefanie Roos, Patrick McCorry, and Arthur Gervais. Sok: Layer-two blockchain protocols. In Joseph Bonneau and Nadia Heninger, editors, *Financial Cryptography and Data Security*, pages 201–226, Cham, 2020. Springer International Publishing.

[4] Rami Khalil and Arthur Gervais. Revive: Rebalancing off-blockchain payment networks. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, page 439–453, New York, NY, USA, 2017. Association for Computing Machinery.

[5] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. https://bitcoin.org/bitcoin.pdf.

[6] Olaoluwa Osuntokun. Amp: Atomic multi-path payments over lightning. https://lists.linuxfoundation.org/pipermail/lightning-dev/2018-February/000993.html, 2018.

[7] Joseph Poon and Thaddeus Dryja. The bitcoin lightning network: Scalable off-chain instant payments. https://www.bitcoinlightning.com/wp-content/uploads/2018/03/lighting-network-paper.pdf, 2016.

[8] Stefanie Roos, Pedro Moreno-Sanchez, Aniket Kate, and Ian Goldberg. Settling payments fast and private: Efficient decentralized routing for path-based transactions, 2017.

[9] Vibhaalakshmi Sivaraman, Shaileshh Bojja Venkatakrishnan, Mohammad Alizadeh, Giulia Fanti, and Pramod Viswanath. Routing cryptocurrency with the spider network. *Proceedings of the 17th ACM Workshop on Hot Topics in Networks.*, pages 29–35, 2018.

[10] Gijs van Dam, Rabiah Abdul Kadir, Puteri N. E. Nohuddin, and Halimah Badioze Zaman. Improvements of the balance discovery attack on lightning network payment channels. In Marko Hölbl, Kai Rannenberg, and Tatjana Welzer, editors, *ICT Systems Security and Privacy Protection*, pages 313–323, Cham, 2020. Springer International Publishing.

[11] Yuup van Engelshoven and Stefanie Roos. The merchant: Avoiding payment channel depletion through incentives, 2020.

[12] Jan Vermeulen. Bitcoin and ethereum vs visa and paypal - transactions per second. https://mybroadband.co.za/news/banking/206742-bitcoin-and-ethereum-vs-visa-and-paypal-transactions-per-second.html, 2017.

[13] Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. http://gavwood.com/paper.pdf, 2014.

A Other Splitting Methods

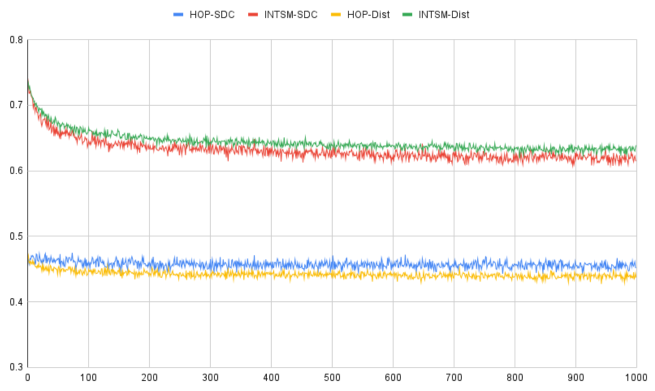

Figure 5: Evaluation of SDC

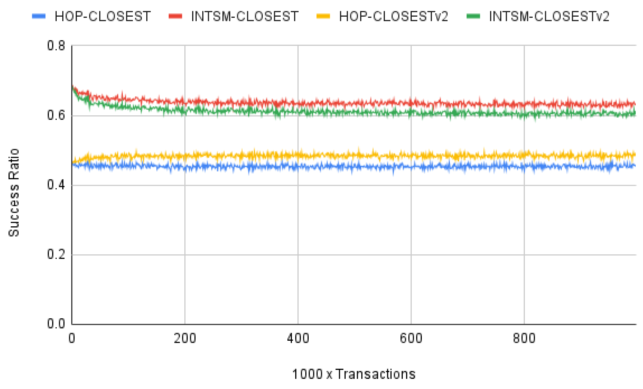

Figure 6: Success ratio evaluation of splitting when rebalancing channels

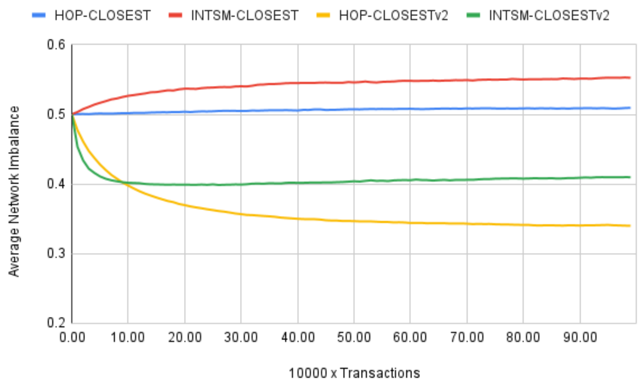

Figure 7: Average network balance