Profitable Fee Controller for Payment Channel Networks

Anupa De Silva$^{1[0000-0003-2711-4211]}$, Subhasis Thakur$^{2[0000-0001-6579-724X]}$, and John Breslin$^{3[0000-0001-5790-050X]}$

Data Science Institute, National University of Ireland Galway, Ireland
{anupa.shyamlal, subhasis.thakur, john.breslin}@insight-centre.org

Abstract. Payment channel networks (PCN) are one of the promising second layer solutions for the scalability issue of cryptocurrencies. They offer high throughput and low-cost transactions for the users. However, practical and design issues of PCN hinder users from embracing the technology. This paper focuses on how to increase user adoption by boosting revenue as transaction mediators. Our approach is twofold; reduce the cost of refilling channels due to channel exhaustion and collectively acquire a maximum gain from all channels belonging to the respective nodes. For that, we devise a fee controller based on Model Predictive Control (MPC), where we formulate the problem as an Optimal Control Problem (OCP). It is optimized to obtain maximum profit while proactively alarming the payers about the exhaustion through a dynamic fee mechanism. We evaluate the proposed fee controller's effectiveness, robustness, and reaction against congestion compared with the conventional static fee controller and OptimizedFee controller. Our experimental simulations using our agent-based PCN simulator, Sala, demonstrate that the proposed fee controller successfully improves the revenue and controls channel exhaustion simultaneously.

Keywords: Cryptocurrency · Payment Channel Networks · Layer-two · Model Predictive Control · Fee Controller

1 Introduction

Industries are rapidly embracing blockchain technology regardless of what domain they are in, tech or non-tech. Primarily, they benefit from its traceable, decentralised, and immutable infrastructure. However, the Blockchain Trilemma points out that prioritising blockchain's security and decentralised nature restricts blockchain's scalability. Among many proposals [4] to control the scalability issue, layer-two protocols are prominent as an orthogonal scaling solution that can absorb the robustness of the underlying blockchain layer (i.e., layer-one) and operate independently. PCN emerged as a promising layer-two solution introduced to minimise blockchain latency and cost issues and increase the throughput. PCN facilitate unlimited miniature transfers with instant settlements, making it ideal for micropayments. Micropayments are not feasible on decentralised cryptocurrencies (e.g. Bitcoin) due to their slow transaction confirmation and expensive fees incurred by time-consuming and high-cost computations to build the consensus. Theoretically, PCN are only bounded by the communication overhead between participants. Hence, it can operate fast and economically with high throughput yet provides the same security as if the transfer happened on the chain.

The cryptocurrency community is increasingly embracing PCN due to the cryptocurrency price surge. However, the general comment on PCN in academia is that PCN have built-in design issues and demand a wider user adoption to overcome them. In contrast, users expect promising outputs to invest in PCN. Both raise a chicken and egg problem. Most PCN implementations are decentralised. The findings of Waugh et al. in [12] confirm that the PCN nodes are unavailable for an extended period. It is a common phenomenon for decentralised networks where nodes are free to join and leave the network. For example, such turnover can be seen in the Tor network [2] where participants are volunteering. Tor network operation is nearly similar to PCN, yet we cannot see a notable distinction between incentivised (i.e. PCN) nodes and volunteer nodes(i.e. Tor). Therefore, decentralisation is not the only culprit. In this paper, we mainly focus on channel exhaustion which is one of the practical issues in PCN caused by the imbalanced flow of payments. An intuitive phenomenon is that when a node is neighbouring a merchant node, the transaction flow tends to skew towards the merchant node. As a result, the channel's fixed capacity becomes limited to cater to new transactions requiring refilling again and again. Refilling demands an on-chain operation which takes time and an expensive fee. It causes channel downtime and an additional cost to the channel owner. Such channel exhaustions primarily affect the intermediaries who joined the network to benefit through transaction mediation. Controlling this issue enables intermediaries to sustain the channel for an extended period and maximise revenue.

This paper aims to strengthen the PCN user base by empowering intermediary nodes in the network. We devise an automated fee controller to manage the inbound and outbound cash flow, which avoids or instantly recovers from channel exhaustion. It is mainly beneficial for intermediary nodes to survive in the network for an extended period and reduce on-chain fees incurred by refilling. At the same time, it is designed to achieve the fundamental objective of an intermediary, gaining maximum utility of the channels, thus, yielding the maximum possible profit. Our contribution extends to a PCN simulator (called Sala), introduced to evaluate the proposed mechanism. Sala is a multi-agent-based simulator that can generate PCN as a graph-based network and simulate the payment execution and malicious behaviours.

The rest of the paper is organised as follows. First, we present the operations of PCN in general and relevant issues, followed by existing research related to fee policies and channel exhaustion. Then, we formulate the optimisation problem for Model Predictive Control and present the design of the proposed controller. Using Sala, we perform experiments on the controller's effectiveness, robustness, and reaction to congestion, then an evaluation by comparing it with two other controllers.

2 Background

PCN operate by creating channels between peer-to-peer couples. Each couple collectively stores a locked deposit on the blockchain using a Multisig Smart Contract. The smart contract is programmed in such a way that this deposit can be released either with the consent of both participants (both should digitally sign) or after the expiration of the lock. The speciality is that the participants can keep updating their respective funds adhering to the same agreement but refrain from recording it on the chain. Every update is publishable; hence, at any point in time, both can publish the latest state on-chain individually (i.e., after a dispute) or collaboratively and close the channel. Hence, the offline consensus is built through the threat to publish. It is, in a way, double spending where the latest one gets rejected if one participant pushes an old state to the chain. Rationally, it is a state which is more beneficial for the publisher. To avoid that, we can include a time-lock that makes each state update valid after a specific time; hence the victim gets a time window to push the latest on the chain.

Creating a channel for every pair is also a costly operation on the chain. Therefore, when a couple has not established a direct payment channel, we can route the payment through a set of intermediaries who have already opened channels. This synchronisation of channels can further reduce the creation of new on-chain transactions. For that, PCN extend the same smart contract using an additional lock using the hashed value of a secret. The process starts with the payment receiver generating a secret and sending its hash value to the sender. The sender finds a path to the receiver through connected channels. Each participant, including the original sender, must compensate the peer in return for the secret of the hash value before a set expiry time. They should follow the path to the original receiver to obtain the secret from the original receiver, who is compensated by the actual amount at the end of the path. Intermediaries return to the original sender with the preimage, where they can unlock a fee for the brokerage in addition to the actual amount. In general, there is a base fee plus a fee proportional to the transferred amount. Once the sender receives the secret, they can verify and prove the transaction completion.

2.1 Payment Participants

We identify three types of entities in PCN with different intentions. Current literature has given less or no attention to recognising their perspectives distinctively. They are the payment participants, defined as payers, payees and intermediaries, who can also overlap. Payers and payees actuate the coin liquidation within the network. The liveness of the network depends on their existence. Payers initiate transactions through the nodes and expect security, reliability, and privacy from PCN. A low transaction fee is also an essential requirement. According to the literature [10], there is a trade-off between transaction cost and privacy for payment participants to be beneficial. Payees are primarily merchants who expect the payments through the network and intend to engage with a larger audience. Merchants can also cause an imbalance in the network because of continuous payment in-flows. Intermediaries intend to earn by providing brokerage services to the network. They are profitable based on transaction fees and the position of the network.

2.2 PCN Issues

Specific PCN design considerations cause inevitable reliability and availability issues for the users. PCN nodes are not revealing their actual balances in real-time but the initial capacities (on-chain deposit from both participants). However, it is a requirement for all payment participants to hold sufficient funds to forward the transaction. Payers have to guess from the capacities when the actual balances are concealed or learn it from trial and error before a successful transaction. For example, the channel selection criteria can be set so that the transaction amount must be less than the channel capacity. Although it is necessary to be true, it is not sufficient to qualify a channel because of the disparity of balances. If failed, the payer has to choose another path, probably with higher fees. The empirical study on a real-world PCN in [12] reveals that the transactions fail mainly due to insufficient balance. The conventional view on this issue is that disclosing the balances is a threat to the privacy of the nodes. However, there are practical attacks [6, 5] to expose the channel balances by manipulating the free rides offered by PCN. It is possible to hop from payer to payee at no cost and gain usable insights about the nodes and transactions. For example, the attacker can initiate a payment with an invalid hash lock for a known or unknown recipient. This bogus payment inevitably fails, and refunds (i.e., free attack); still comes back with manipulatable information.

Apart from the design issues, there are practical complications in PCN. Imbalance transaction flow of the network is one of them. When the payment flow gets biased in one direction, a disparity is created between the inbound and outbound of the node, and the outbound flow eventually becomes unusable. Since they are on two separate transactions on the chain, they cannot be adjusted offline, although both belong to a single person. It takes at least one on-chain transaction unless the channels are created within a channel factory [1]. An unbalanced network is the root cause of most PCN complications. It intensifies transaction failures as a consequence of the balance concealment. Nodes in an unbalanced transaction flow have to face frequent re-fillings or closures of channels and bear the respective on-chain fees. It deteriorates the revenue of the nodes, particularly the intermediary nodes. Further, it affects the availability of the nodes considering the overall network. Another inter-related issue is the traffic congestion in channels. Congestion occurs when there are unsettled transactions piled up, and as a result, the channel cannot accommodate other transactions. In the payment execution, participants must wait for the secret to arrive. Till then, the locked-up funds are also not usable. On-hold time is a crucial factor as it is a missed opportunity to engage in a different payment execution which causes a loss of revenue. An attacker can also leverage this along with the free rides mentioned above and deploy an attack to handicap a channel.

3 Related Works

Current literature consists of several techniques to handle channel exhaustion issues in PCN, including refilling and building channel factories. When the channel is depleting or depleted, the channel participants have to record the updated amounts on the chain and close the channel. If they are willing to restart, two on-chain operations will require closing and re-opening the channel. However, the splicing technique can limit them to one operation, yet it still costs one on-chain transaction fee. Instead, they can adjust their balances by making circular payments to themselves. Pickhardt et al. [9] device an optimisation algorithm to maintain a balanced network by making circular payments. It still costs the intermediary fees. REVIVE by Khalil et al. [7] introduces a free of charge(subject to the responsiveness of the nodes) re-balancing mechanism without closing the channels. Here, PCN nodes collectively shift their funds in a circular path and refill their balances. However, this method introduces a centralised party to the network.

Creating Channel Factories [1] is a safety measure to handle the depleted channels in the channel creation stage. In channel factories, more than two parties collaborate in the initial funding and later perform bilateral transactions reducing the space taken on-chain by offline adjustments to the funds. However, when the number of involved parties increases, the number of disputes surges and the channel factory begins to operate as a set of normal bilateral channels.

The skewness of transaction flow is the primary cause of channel exhaustion. Therefore we can control the channel exhaustion by preventing skewness. The current research works realise this by introducing different routing protocols and fee policies. Thakur et al. propose a flocking based routing protocol for a balanced network and empirically analyse its effectiveness in [11]. Stasi et al. device a fee policy for a sustainable and balanced network focusing on multi-part payments [3]. They measure the imbalance of a channel by the difference in channel balances and devise a fee calculation algorithm. Channel imbalance decides which proportion of payment amount to be charged by the payer. They intend to maintain equal balances in both peers, making payers effortlessly determine if a node holds sufficient funds. As a result, it increases the chances of payment execution being successful. They observe a significant reduction in network imbalance when using the proposed policy in a simulated network.

In summary, we can find preventive (i.e. Channel Factories), proactive(i.e. balanced routing), and reactive (i.e. re-balancing) solutions to handle exhausted channels. In the bidirectional payment networks, asymmetric flow is a recurrent event. Therefore, preventive or reactive methods are not always sufficient as a long term answer. In contrast, proactive strategies can demonstrate promising results for an extended period. The existing works minimise the balance difference (i.e. a central balance) to maintain a balanced network, increasing reliability. However, a central balance also implies limited channel liquidity, which could affect the income of the intermediary nodes.

4 MPC based Fee Controller (MPC-FC)

Intermediary nodes stake their coins to join the network to gain financial earnings by facilitating PCN payment execution. In designing the fee controller, we prioritise maximising their profit while controlling the channel exhaustion.

The process of controlling the channel exhaustion is intuitive, such that we manage the outbound coin flow using the fee rate adjustment. When the outbound balance depletes, we raise the fee to discourage users from choosing the channel in their path selection until inbound traffic reinforces and restores the depleted node. In reality, it is more complex as a node can hold several channels. For instance, recovery from depletion expects to satisfy both outbound flow restrictions and attract inbound. The node cannot do the latter as it has no control over the inbound flow as it does in the outbound flow (its peer controls the fee). However, the flow should exit from one of the channels owned by the same node, which means lowering the fees of the rest of the channels can incentivise inbound transactions. Still, it is a trade-off as reducing fees also affects the total income of the node. We design the proposed fee controller, MPC-FC, as an optimisation problem where channels collaborate to maximise the revenue while controlling the channel exhaustion.

4.1 Model

Assume a node engages with n other nodes, creating n number of channels with capacities denoted by the vector C. At time step k, we define the following states of the node.

Definition 1. *Let M_k be an $n \times n$ matrix denoting monetary values of incoming and outgoing transactions to and from each peer through the node at time step k.*

For example, $m_{i,j} \in M$ holds the total monetary value of the transactions which comes from j^{th} peer and exits to i^{th} peer through the node.

Definition 2. *Let $n \times 1$ matrix, F_k denotes the fees charged by the node for each channel as a portion of the transaction amount being mediated and $\forall f \in F_k,\ 0 \leq f_{min} \leq f \leq f_{max} \leq 1$.*

If we denote the movable balances with each peer by the matrix B_k at time step k,

$$B_{k+1} = B_k + \Delta B \tag{1}$$

where $\mathit{\Delta} B = sumv(M_k)^T + sumh(M \times (F_{k-1} - J_{n,1}))$ which includes incoming payments and fees and exclude outgoing payments. *sumv*, *sumh* and *sum* are the functions to calculate vertical, horizontal and total sums of a matrix respectively. J indicates a matrix of ones.

Let the profit of the node denoted by p_k at time step k.

$$p_{k+1} = p_k + sum(F_{k-1} \times M) \tag{2}$$

In this paper, we define M_{k+1} as follows where, $norm(F)$ is the normalized fee rates and O_{max} is the maximum outbound values ($J_{n,n}B$). w_m is the weight we set depending on the traffic. In simulations, we have set the $w_m = s/n$ where s denotes the frequency of fee controller is run.

$$M_{k+1} = w_m \times O_{max} \times norm(F) \tag{3}$$

Here, we manipulate M_{k+1} such that the controller has to raise the fee when the outbound balance is low and vice versa to maximize the profit.

4.2 Optimization Problem

We design the following optimal control problem based on the above.

$$min \sum_{k=0}^{H} -p_k + w_f \times \mathit{\Delta} F \tag{4}$$

such that: equations 1, 2, 3, and

$$J_{n,1} \times f_{min} \leq F \leq J_{n,1} \times f_{max} \tag{5}$$

$$O_{n,n} \leq M \leq J_{n,1} \times C \tag{6}$$

$$O_{n,1} + a_{min} \leq B \leq C - a_{min} \tag{7}$$

$$B_0 = J_{n,1} \times C, p_0 = 0, M_0 = O_{n,n} \tag{8}$$

Here we maximize the profit which is subjected to satisfy the inequalities in 5-7. w_f is the weight that can prioritize the fee fluctuations. O denotes the matrix of zeros, and we use a_{min} to set lower and upper margins for the balances to avoid exhaustion.

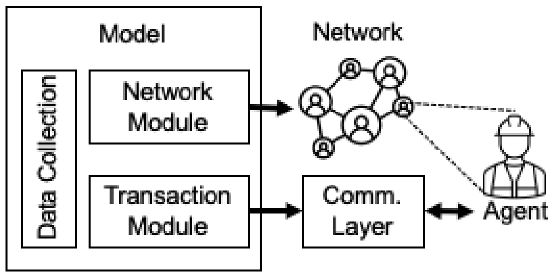

Fig. 1. Modeling PCN into an agent base simulator, Sala

4.3 Model Predictive Control

The fee controller uses the MPC technique to repeatedly solve the above OCP and obtain the optimal fee rates for the channels of the selected node. It starts with initial states as mentioned in equation 8 where initial balances are the individual channel capacity, profit, and transaction amounts are all zero. Then, for a given frequency and states, MPC computes the control input, F, by solving the OCP for the predicted H number of future steps. Then it applies the first fee vector of the computed control sequence and observes the states (M, B, and r), which are fed back to the system to recompute the OCP. This continues in every s step of the simulator.

We implemented MPC-FC using the do-mpc library[1], which is based on CasAdi[2] and IPot solver[3].

5 Simulator

This section introduces the agent-based simulator, Sala, which was developed to simulate the function of a payment channel network. It models nodes as agents who interact with each other to execute payments. They establish and maintain payment channels among peer nodes and then initiate, receive, and mediate the transactions. We can control the fees and monitor the revenues and balances of those nodes. In the implementation[4] of Sala, we use Mesa library[5] to model the PCN nodes as agents. The overall process of Sala is as follows.

As depicted in Figure 1, Sala consists of a collection of configurable modules and components. The network module initializes the payment nodes and channels as a graph-based network. Then the transaction module generates the transactions for the network. Both accommodate the different behaviours of

[1] www.do-mpc.com
[2] web.casadi.org
[3] www.coin-or.github.io/Ipopt
[4] www.github.com/anupasm/SALA
[5] mesa.readthedocs.io

three types of agents in PCN (i.e. payers, payees and payment mediators). Further, they can act independently or work on a given set of data, which allows repeating the experiments on the same data set (i.e. transactions). Those algorithms are customizable, either probabilistic or static (e.g. assigning channel capacity, payer and payee selection). The communication layer is responsible for the agent communication of transaction information. Additionally, data collection and visualization components operate along with all the operations to collect and display the real-time network and its data.

Once the nodes and channels are determined, the system advances step by step, and in each stage, agents act simultaneously. With the help of the communication layer, an agent can initiate a payment, forward the transaction, temporarily lock up the respective transaction amounts, provide the valid secret (i.e. payee), verify the validity, pass back the transaction and update the balances. In the meantime, they also monitor the expiration times included in the transactions. They can also be configured to restart the transaction on an alternate path in case of failure (i.e. timeout, insufficient balance, invalid secret).

Sala is configurable to model selected agents to act according to the customized instructions. For example, we can set different fee controllers for the agents to determine their charge according to the circumstantial parameters (i.e. transaction amount, balance). We have integrated Static Fee Controller (S-FC), OptimizedFee Controller (OF-FC) [3], and the proposed MPC based fee controller (MPC-FC). Further, there is a behaviour component to model the behaviour of malicious agents who can engage with nodes and inject bogus transactions.

This section introduces the agent-based simulator, Sala, which was developed to simulate the function of a payment channel network. It models nodes as agents who interact with each other to execute payments. Agents establish and maintain payment channels among peer nodes and then initiate, receive, and mediate the transactions. We can control the fee strategy and monitor the revenues and balances of those nodes. In the implementation[6] of Sala, we use Mesa library[7] to model the PCN nodes as agents. The overall process of Sala is as follows.

As depicted in Figure 1, Sala consists of a collection of configurable modules and components. The network module initializes the payment nodes and channels as a graph-based network. Then the transaction module generates the transactions for the network. Both accommodate the different behaviours of three types of agents in PCN (i.e. payers, payees and payment mediators). Further, they can act independently or work on a given set of data, which allows repeating the experiments on the same data set (i.e. transactions). Those algorithms are customizable, either probabilistic or static (e.g. assigning channel capacity, payer and payee selection). The communication layer is responsible for the agent communication of transaction information. Additionally, data collection and visualization components operate along with all the operations to collect and display the real-time network and its data.

[6] www.github.com/anupasm/SALA

[7] mesa.readthedocs.io

Once the nodes and channels are determined, the system advances step by step, and in each stage, agents act simultaneously. With the help of the communication layer, an agent can initiate a payment, forward the transaction, temporarily lock up the respective transaction amounts, provide the valid secret (i.e. payee), verify the validity, pass back the transaction and update the balances. In the meantime, they also monitor the expiration times included in the transactions. They can also be configured to restart the transaction on an alternate path in case of failure (i.e. timeout, insufficient balance, invalid secret).

Sala is configurable to model selected agents to act according to the customized instructions. For example, we can set different fee controllers for the agents to determine their charge according to the circumstantial parameters (i.e. transaction amount, balance). We have integrated Static Fee Controller (S-FC), OptimizedFee Controller (OF-FC) [3], and the proposed MPC based fee controller (MPC-FC). Further, there is a behaviour component to model the behaviour of malicious agents who can engage with nodes and inject bogus transactions.

6 Experiments and Results

Our objective is to investigate how efficiently the proposed fee controller reduces channel exhaustion and calculate the cost incurred as a consequence. We empirically examine the proposed fee controller using the Sala simulator. Further, we examine the robustness of the fee controllers and how they react to the congestion in the channel.

We do not use real-world PCN transaction data (which is not available as well) for the experiments; instead, we use extensive simulations to simulate PCN behaviour by randomising transaction parameters (i.e. payer, payee, and value). Those simulations were carried out at a fixed number of nodes, capacities, limits of transaction values and fee percentages. However, our result evaluation is independent of such parameters as we evaluate fee controllers proportionally.

6.1 Setup

In the overall setup, we randomly generated networks of 50 nodes connected with six other peers on average. Each channel capacity is identical, and both peers contribute equally ($c = 100$). Also, we assign the three identified roles, merchant, intermediary and client, to each node to the proportion 1:2:2. For a realistic simulation, we assume that initiating and receiving transactions depends on the node's role. For example, we set a high probability for merchants to receive more incoming payments. And those payments are randomly generated in between $m_{min} = 4$ and $m_{max} = 10$. For simplicity, we assume the base fee is negligible and common for all nodes. Then, we set the static fee rate to 10% of the transaction value whereas MPC-FC and OF-FC charges $f_{min} = 5\%$ to f_{max}=15%. We also assume that payers always choose the cheapest and shortest path, which is intuitive.

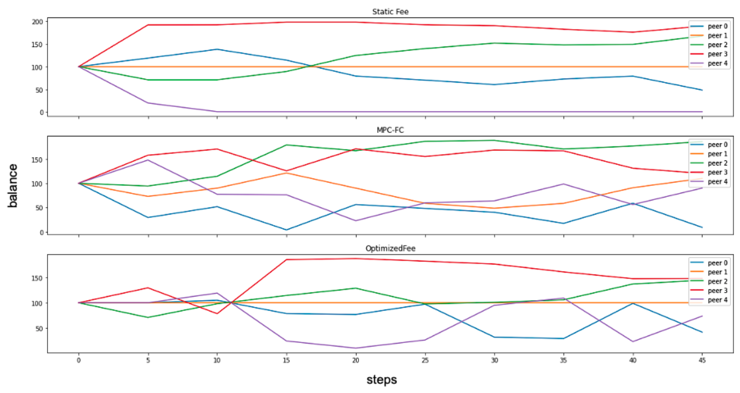

Fig. 2. An example of how the channel balances evolve with three controllers.

In each simulation step, we monitor the collected fees, balances held by each node from the completed transactions, and the values of the incoming and outgoing transactions that pass through the node. We derive three variables from those data to analyse channel exhaustion. The first one is the total exhausted steps in the form of the step counts when the balances of related channels drop below m_{min}. Setting the lower limit to zero is impractical as the balance does not drop below that. We define average recovery time, extending the total exhausted steps where we calculate the average of exhausted segments to gain an insight into the recovery time. The last variable is the dropped fees from the transactions that could not pass because of insufficient balance. All values were extracted after $s = 500$ number of steps.

6.2 Exhaustion and Revenue

The first experiment compares how the three controllers manage exhaustion and revenue generation. We repeat the same simulation three times with one of the three controllers each. As described in the overall setup, the first run generates the network and transactions and lets all nodes charge a static percentage of transacted value from all transactions. In the second and third runs, we embed the MPC-FC and OF-FC controllers inside the exhausted node, respectively and simulated the same network and the same set of transactions, excluding the transaction paths. We let payers determine the transaction paths independently in real-time to pick the cheapest and avoid expensive paths.

Figure 2 demonstrates an example run of how channel balances of a node got changed with different fee controllers. The selected node is connected with five peers. Under the static fee controller, the channel with the fourth peer channel gets exhausted immediately and remains so. Other controllers managed to recover instantly; however, as anticipated, the OF-FC controller tries to keep the balances central, whereas the proposed controller scatters the balances within limits. We observe this in most of the simulations.

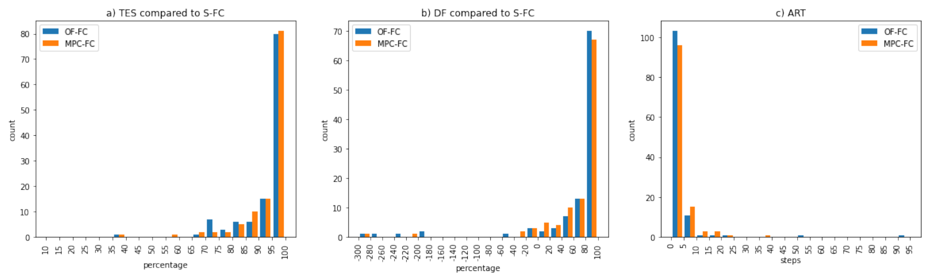

Fig. 3. Histograms of three parameters, indicating different aspects of exhaustion of 119 simulations. a) Percentage distribution of total exhausted steps reduction compared to the respective S-FC based simulation. b) Percentage distribution of dropped fees compared to the respective S-FC based simulation. c) Distribution of average recovery time in steps.

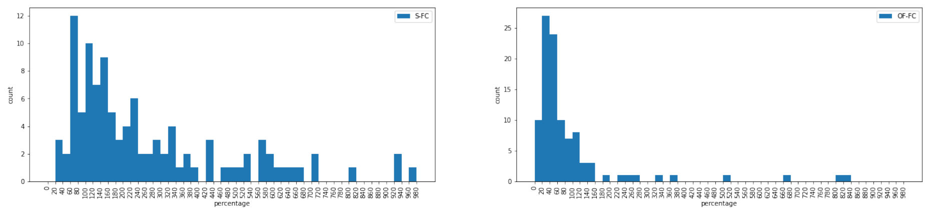

Fig. 4. MPC-FC collected fees compared to S-FC and OF-FC in 119 simulations.

Figure 3 presents the distribution of percentages of total exhausted steps reduction, dropped fees reduction, and average recovery time figures derived from 119 number simulations. Total exhausted steps and dropped fees values of OF-FC and MPC-FC simulations are given in proportion to S-FC based runs. OF-FC and MPC-FC demonstrate nearly similar behaviour in all three sections. Both limit the exhaustion to around 5% compared to S-FC, and most importantly, average recovery time values indicate that most nodes could recover from exhaustion after around five steps. The same applies to dropped fees figures, where both controllers reduce the dropped transactions amid exhaustion by approximately 80%. Figure 3 indicates the fees collected by the proposed MPC-FC compared to the S-FC and OF-FC, respectively. MPC-FC achieves higher collected fees in all 119 simulations, and on average, it is a 231.32% and 56.34% raise in profit compared to S-FC and OF-FC, respectively.

Overall, MPC-FC surpasses the other two controllers in terms of collecting fees. Both OF-FC and MPC-FC perform significantly better than the S-FC in managing exhaustion. We observe a similar behaviour between OF-FC and MPC-FC in all aspects considered.

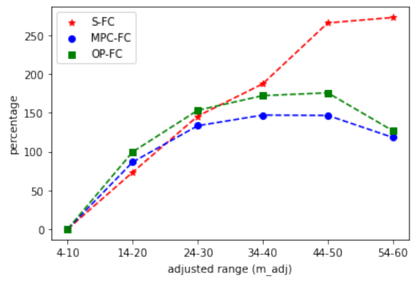

Fig. 5. Average percentages of collected fees compared to the initial run.

6.3 Robustness

Secondly, we intend to inspect the robustness of the fee controllers with the increasing transaction values. All three controllers are sensitive to transaction amounts, and on the other hand, higher amounts can intensify the exhaustion. To test that, we repeat a single simulation multiple times with the same network and same transaction except that a static value, m^i_{adj} is added to the initial value of each transaction. We increase the static value along the duplicate runs hence, transaction values are adjusted in between $m_{min} + m^i_{adj}$ and $m_{max} + m^i_{adj}$.

Figure 5 gives the average percentages of collected fees compared to the initial run from 67 simulations. S-FC demonstrates approximately linear development, whereas the other two gradually increase the income initially, which eventually deteriorates. Although MPC-FC has the lowest acceleration rate, it is still 126.73% and 43.21% higher than S-FC and OP-FC, respectively, in the last run.

6.4 Congestion

Finally, we intend to examine the reaction of a channel in the presence of a stream of injected transactions that cause congestion in it. It helps to understand how the proposed MPC-FC works with congestion. Besides, this scenario is analogous to an existing attack on the real network, congestion attack [8] where bogus transactions are generated to create congestion on a selected channel. To realise this, Sala facilitates putting up malicious nodes and programming them to initiate bogus transactions through a targeted channel. The process is as follows.

Let x_1 and x_2 denote the malicious nodes whereas the channel between v_1 and v_2 is the victim. To deploy the attack, x_1 and x_2 engages with v_1 and v_2 respectively to establish the path $x_1 - v_1 - v_2 - x_2$. We also assume that x_1 connects with other intermediary nodes as well. Then, x_1 initiates bogus transactions through that path. The malicious recipient, x_2 on the other end, provides an invalid pre-image, making those transactions return as failures. The respective values are locked up until that and not utilisable for other transactions. It causes channel congestion and cripples the channel. The attacker has

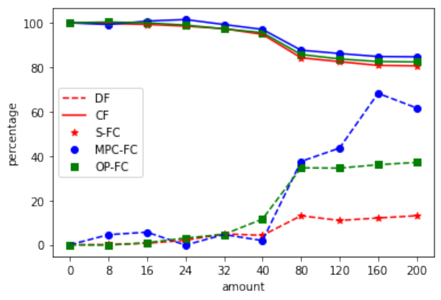

Fig. 6. Average percentages of collected fees and dropped fees at different congestion levels.

to maintain a stream of transactions to persist the attack as the victim resets the locked amounts subsequently. Apart from the initial channel creation cost, bogus transactions do not cost intermediary charges for the attacker because they fail eventually.

Each test consists of six simulations in the experiment and is carried out for all three controllers. First, we identify the most valuable channel as the victim and deploy x_1 and x_2 with extra initial capacity ($c = 300$). x_1 initiates bogus transactions worth 8 to 200 in each step. They are segmented by the minimum transaction value (m_{min}) to avoid early rejections by the victim due to insufficient balance. The initial run is the control simulation where x_1 does not initiate the bogus transactions. Figure 6 summarizes the results from 67 test runs. Collected fees and dropped fees figures are compared to the respective initial run against the balance forged by the attacker. Collected fees decline approximately by 20% for all controllers and take the same shape. The congestion causes higher drop rates, especially when the forged amount is half the node capacity. It is when the congestion reaches the maximum limit. According to the dropped fees figures, MPC-FC shows comparably higher sensitivity towards congestion. However, in the last run, its collected fees ar e 36.37% and 23.16% higher than S-FC and OF-FC, respectively.

7 Evaluation

This section evaluates the proposed controller from different aspects by benchmarking with the other two using the observations obtained in the experiment section. Primarily, we focus on evaluating the gain and the exhaustion using values of four variables, Total Exhausted Steps, Average Recovery Time, Collected Fees, and Dropped Fees. Secondly, we will analyse the outcome of the experiments on fee controllers that faced the stream of transactions and elevated fee ranges.

The intermediaries in PCN expect to make the most of its limited investment. Here, we proposed an approach similar to OF-FC to adjust fees to control the balance. However, we refrain from the idea of keeping the balance central and considering channels in isolation. All the channels belonging to the relevant node collaboratively adjust the fees to maximise the profit. Therefore, according to the test run results of collected fees, the proposed MPC-FC surpasses the OF-FC by more than 50% margin, and it is more than two times what the static controller collected. The reason for the significantly low figures given by S-FC is that its channels remained exhausted for an extended period. We did not refill the channel upon exhaustion for simplicity; otherwise, S-FC based node should bear an additional cost (i.e. on-chain fee) which should be deducted from the profit.

The next and the main objective of this research is to minimise channel exhaustion. For that, total exhausted steps indicate how well the channel avoids exhaustion. In solving the OCP of MPC-FC, we have set constraints to maintain a minimum balance. According to the outcome of the simulations, OF-FC and MPC-FC equally perform well in dropping the total exhausted steps figures significantly even though there is a subtle difference. There are occasions where it is inevitable to avoid balance hitting bottom. However, it is not an issue as long as it can recover. Average recovery time indicates how fast a channel can recover when it gets exhausted. Fees are raised in OF-FC and MPC-FC when the channel balance deteriorates. In the meantime, the input stream flows in the reverse direction and raises funds. The average recovery time results in Figure 3 indicates that both can recover from exhaustion within around five steps. OF-FC and MPC-FC demonstrate nearly similar performance. It does not occur with a fixed fee as in S-FC, and as a result, the channel rarely recovers from the exhausted state. Putting both outcomes together, we can derive that both OF-FC and MPC-FC can lessen the exhaustion and instantly recover from exhaustion.

Even though a channel is not exhausted, it can remain at a low balance and drop the incoming transactions. Therefore, dropped fees represent the lost income because of insufficient balance from a node's perspective. Dropped fees also imply the Transaction Failure Rate (TFR) in the view of the network reliability. The literature review found research works promoting PCN's balancedness to reduce the TFR, including the OF-FC. OF-FC tries to keep the balance central by adjusting the fees. In contrast, the proposed MPC-FC tries to stir the channel balances to secure maximum profit. Therefore, it is intuitive to expect higher TFR (in this case, dropped fees) in MPC-FC. However, dropped fees results in the Figure 3 suggest otherwise where we cannot see a significant difference between OF-FC and MPC-FC. The primary reason is that MPC-FC signals PCN senders using the channel by increasing the fees and vice versa. It is similar to revealing balances, but, in this case, higher fees actively discourage the senders. Hence, MPC-FC managed to control dropped fees to a reasonable extent. To summarise, MPC-FC has simultaneously achieved the main objectives of an intermediary, higher collected fees and lower total exhausted steps, average recovery time, and dropped fees compared to S-FC and OF-FC.

The second experiment inspects the three fee controllers against different scales of transaction values. Naturally, the node income should linearly grow as the fees are calculated proportionally to the transaction amounts. Accordingly, our simulation results demonstrate an approximately twofold increase in income when the transaction amount doubles. However, the increase drops for the higher values in both MPC-FC and OF-FC. At the same time, an acceleration of dropped fees is also observable. The primary reason is that the higher values deteriorate the balances quickly and cause rejection of higher valued transactions. PCN's inability to transfer higher valued transactions is a common issue. However, PCNs are primarily used for micropayments, and it is recommended to use multi-part transactions for larger values. Moreover, the results in Figure 5 also imply how much balance a node should hold for MPC-FC and OP-FC to perform well. There are significant drops in increase when the transaction values range around half of the channel capacities. Hence, we can derive that the initial capacities should be at least twice the nominal value of transactions.

We also simulated a congested channel to gain an insight into the robustness of fee controllers against congestion. According to Figure 6, it is inevitable to avoid the victim channel being crippled by the attack. When the fabricated congestion reaches the capacity of the victim node, the dropped fees surge and MPC-FC is the most affected. However, in terms of collected fees, MPC-FC is still in the lead, suggesting MPC-FC performs better even under the worst-case attack scenario from the node's perspective.

Fee policy communication is crucial with MPC-FC and OF-FC because of the decentralised nature of PCN. Both MPC-FC and OF-FC are dynamic controllers that alter the fees according to the dynamic states of the system. Therefore, the frequency of communicating the fee rates should be designed to avoid flooding the system. Moreover, nodes with MPC-FC and S-FC do not necessitate exposing their balances to the network, whereas OP-FC implicitly expects private balances to be public. Hence MPC-FC can reduce transaction failures, protecting the privacy of the PCN nodes simultaneously.

The future direction includes inspecting the network-wide effect of using the proposed controller. Here we have explored individual usage of the fee controller. However, the same concept is applicable in a distributed way to achieve network-wise goals. It is important to examine how it will affect the transaction reliability and payers' cost in PCN, in general.

8 Conclusion

Payment Channel Networks have the potential to become the solution for micropayments, while mainstream cryptocurrencies have become almost infeasible to perform day-to-day transactions without compromising security. However, there are few design and practical issues in PCN that hinder user adoption and consequently limit PCN's true potential. This paper attempts to mitigate one of such barriers: we empower users by reducing their costs caused by exhaustion and improving the gain by operating an intermediary node. We propose a Model

Predictive Control based fee controller and develop a PCN simulator Sala to evaluate the proposed mechanism. Our simulation results suggest that the proposed controller elevates user revenue compared to existing fee controllers and considerably decreases channel exhaustion and transaction failures. This result empirically attests that central balance reduces the node revenue. On the other hand, central balance is not the only solution to reduce transaction failures caused by insufficient balance. Further, we evaluate how these fee controllers act against the elevating transaction values and congestion. Future work intends to improve the fee controller to operate collaboratively with other nodes in a distributed manner, achieving user and network-wide objectives.

Acknowledgements This publication has emanated from research supported in part by a grant from Science Foundation Ireland and the Department of Agriculture, Food and the Marine on behalf of the Government of Ireland under Grant Number SFI/16/RC/3835 (VistaMilk), and also by a grant from SFI under Grant Number SFI/12/RC/2289_P2 (Insight). For the purpose of Open Access, the author has applied a CC BY public copyright licence to any Author Accepted Manuscript version arising from this submission.

References

1. Burchert, C., Decker, C., Wattenhofer, R.: Scalable funding of bitcoin micropayment channel networks. Royal Society open science **5**(8), 180089 (2018)
2. Das, A., Borisov, N., Mittal, P., Caesar, M.: Re3: Relay reliability reputation for anonymity systems. In: Proceedings of the 9th ACM symposium on Information, computer and communications security. pp. 63–74 (2014)
3. Di Stasi, G., Avallone, S., Canonico, R., Ventre, G.: Routing payments on the lightning network. In: 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData). pp. 1161–1170. IEEE (2018)
4. Hafid, A., Hafid, A.S., Samih, M.: Scaling blockchains: A comprehensive survey. IEEE Access **8**, 125244–125262 (2020)
5. Herrera-Joancomartí, J., Navarro-Arribas, G., Ranchal-Pedrosa, A., Pérez-Solà, C., Garcia-Alfaro, J.: On the difficulty of hiding the balance of lightning network channels. In: Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security. pp. 602–612 (2019)
6. Kappos, G., Yousaf, H., Piotrowska, A., Kanjalkar, S., Delgado-Segura, S., Miller, A., Meiklejohn, S.: An empirical analysis of privacy in the lightning network. In: International Conference on Financial Cryptography and Data Security. pp. 167–186. Springer (2021)
7. Khalil, R., Gervais, A.: Revive: Rebalancing off-blockchain payment networks. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. pp. 439–453 (2017)
8. Mizrahi, A., Zohar, A.: Congestion attacks in payment channel networks. In: International Conference on Financial Cryptography and Data Security. pp. 170–188. Springer (2021)

9. Pickhardt, R., Nowostawski, M.: Imbalance measure and proactive channel rebalancing algorithm for the lightning network. In: 2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC). pp. 1–5. IEEE (2020)
10. Tang, W., Wang, W., Fanti, G., Oh, S.: Privacy-utility tradeoffs in routing cryptocurrency over payment channel networks. Proceedings of the ACM on Measurement and Analysis of Computing Systems **4**(2), 1–39 (2020)
11. Thakur, S., Breslin, J.G.: A balanced routing algorithm for blockchain offline channels using flocking. Advances in Intelligent Systems and Computing **1010**, 79–86 (2020)
12. Waugh, F., Holz, R.: An empirical study of availability and reliability properties of the bitcoin lightning network. arXiv preprint arXiv:2006.14358 (2020)