A Reinforcement Learning approach to Lightning Network fee policy

Author Name
Affiliation
email@example.com

Abstract

We employ Reinforcement Learning (RL) to learn a profitable fee policy in payment channel networks. This is a well-known problem in the Bitcoin Lightning Network (LN) with no effective solutions. Evidently, suitable mechanisms for profitable fee policy can incentivize users to join the network and significantly enhance its functionality. So far, no such learning-based algorithms have been studied in this domain and current deployments rely only on heuristics. Hence, LN users who apply these methods do not earn sufficient income, negatively impacting the cost of opportunity. In this paper, we proposed LeVIN, a simulator based on real-world LN data. Accordingly, we applied on-policy RL methods to learn an effective fee policy in the real-world configuration of LN. We overcame fundamental challenges of this environment such as partial observability, reward sparsity, non-stationarity, and multi-objectiveness. Our experiments illustrate that RL agents significantly outperform the existing algorithms for the task of fee-setting in payment channel networks.

1 Introduction

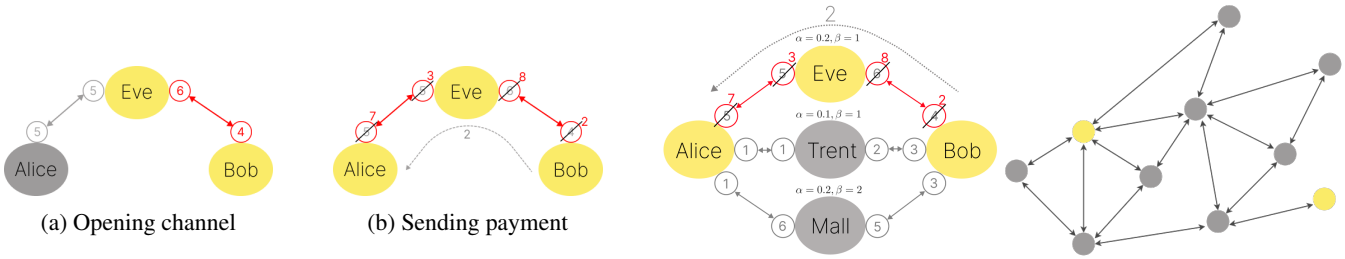
The Bitcoin blockchain's history began with a paper written by the pseudonymous Satoshi Nakamoto [Nakamoto, 2008]. Briefly explained, a blockchain is a sequence of blocks chained together, with each block containing a record of transactions. Currently, Bitcoin holds the highest market capitalization among all of the existing cryptocurrencies, however, it suffers from the scalability issue due to the limited space of each block. The most popular solution for the bitcoin scalability problem is the LN, a decentralized payment channel network capable of processing millions of secure transactions per second. Since the emergence of LN, charging transaction fees has been a crucial component in encouraging users to join the network and provide liquidity to their payment channels. Consequently, the question of which fee policy maximizes profit for each user arises.

RL is a general-purpose formalization of learning through experience. It enables an agent to perform a task by choosing an action and getting a reward signal corresponding to that [Sutton and Barto, 2018]. In the past few years, the advancements in RL were accelerated since the emergence of Deep Reinforcement Learning (DRL), making it possible to use RL in a diverse set of complex applications. However, most RL algorithms were designed under the assumption that the world can be adequately modeled as a stationary Markov Decision Process (MDP), and evidently their performance drop in the face of a sparse reward function. Unfortunately, realistic applications are seldom in this setting. As opposed to the standard assumptions, LN fee-setting problem is partially-observable and suffers from a multi-objective sparse reward function. These challenges prevent most RL methods from coping within such a complex environment, and we believe these are one of the major reasons such methods are yet to be deployed in most potential real-world applications.

To validate our claims, we develop an RL environment based on a realistic LN simulator and performed extensive experiments. To do so, we simplified the fee-setting problem as a Partially observable Markov decision process (POMDP), with fee rates and base fees corresponding to the channels of a selected node as the actions. Due to security reasons, users are unaware of the transaction passing through other nodes, making it a partially observable environment. The reward function might be sparse depending on the topological position of the user in the network and should handle the trade-off between losing liquidity and gaining income, making it a challenging environment to learn. We provide a comparison between our approach and the already existing strategies for the fee-setting problem. Being able to surpass other existing strategies, we hope to take a step forward in using RL for real-world finance problems.

2 Related Work

The LN white paper [Poon and Dryja, 2016], discussed the reason why transaction fee exists in the network and compares the LN fee with the blockchain fee from different aspects. [Di Stasi *et al.*, 2018] proposed a different fee policy that fostered the balancing of channels and improved the network's performance in the long term. [Musacchio and Wu, 2007] deployed a game-theoretic model of network pricing with a focus on the price of anarchy. [Jun Ren *et al.*, 2018] suggested the fee rate to be proportional to the square root of the channel's capacity, in a simple fee structure that only contains the fee rate.



(a) Opening channel (b) Sending payment (c) Routing (d) LN graph

Figure 1: (a) Bob and Eve opening a payment channel. Eve deposits 6 BTC and Bob deposits 4 BTC, resulting in a channel with a capacity of 10 BTC. This transaction is recorded on the Bitcoin blockchain. (b) Bob sends a payment of the amount of 2 BTC to Alice through Eve. After sending the payment, the balance of each channel used in this transaction will be updated. (c) Finding the best route for sending a 2 BTC payment from Bob to Alice. Following the path through Eve, Bob will pay $\alpha \times amount + \beta = 0.2 \times 2 + 1 = 1.4$ BTC fee, and following the path through Mall, Bob will pay $\alpha \times amount + \beta = 0.2 \times 2 + 2 = 2.4$ BTC fee. Note that even though the path through Trent needs less fee, there isn't enough liquidity to route the payment. (d) Any two arbitrary nodes in the LN graph can execute a transaction between each other if there exists a well-funded path between them.

LN was meant to be decentralized, but by the evolution of the network, nodes with higher degrees known as merchants appeared which led to several topological studies on LN including [Seres *et al.*, 2020] and [Martinazzi and Flori, 2020]. Furthermore, [Bartolucci *et al.*, 2020] modeled the growth of LN as a bond percolation process. [Bertucci, 2020] investigated the centralization of LN from the game-theoretic point of view. While there hasn't been a lot of research done on LN fee policy, several papers studied the routing problem in LN and other payment channel networks.([Di Stasi *et al.*, 2018]; [Prihodko *et al.*, 2016a]; [Roos *et al.*, 2017])

Developing user-friendly and reliable software systems helps both researchers and the members of LN to experiment with their ideas. Accordingly, [Conoscenti *et al.*, 2021], same as [Béres *et al.*, 2021] enhanced the software support for experimenting with different fee policies in LN by introducing easy-to-use simulators written in well-known programming languages.

Despite the capable framework of RL and the rapidly growing area of cryptoeconomics, there have been few works in the intersection of these two areas. [Jameel *et al.*, 2020] did a survey on RL in blockchain-enabled IIoT networks and [Dai *et al.*, 2019] combined these two areas for empowering the next-generation wireless networks.

3 Lightning Network Preliminaries

Initially, scalability was not a concern in the Bitcoin blockchain. However, as Bitcoin has grown in popularity, this problem has become a weakness of the entire system, slowing down and raising the cost of transactions. LN is an off-chain protocol layered on top of the Bitcoin blockchain and is mainly proposed for solving the Bitcoin scalability problem. While preserving the security and decentralization properties of Bitcoin, LN allows users to execute faster and cheaper transactions outside of the blockchain.

The LN is a network of payment channels modeled as a graph. In this graph, the vertices represent the users and the edges represent the payment channels. A payment channel is a connection between two members of the network and is a fundamental element of LN. Each channel has a capacity that corresponds to the entire amount of liquidity accessible in that channel. Additionally, each channel peer has its own balance which is a fraction of the channel's capacity. Peers may receive an unlimited amount of payments up to their balance limit.

Two arbitrary users in LN can initiate a transaction with each other if there is a well-funded path between them. A well-funded path is a route consisting of consecutive payment channels where each channel's balance is greater than the amount of the transaction. Each node in this path can charge a small amount of fee in exchange for forwarding the transaction. Furthermore, each peer of these *payment channels* can independently set and change their own fee-setting policy. The final cost of a transaction for the sender will be equal to the sum of the transaction amount and the fees of all nodes along the path.

The fee protocol of LN is the most important element for incentivizing people to lock their money in the network. While the sender is usually trying to minimize the routing cost of its transaction, channels on the route try to maximize their profit by choosing a good fee-setting policy. In the recent implementations of LN, there are two components for a fee, *i.e.*, the fee rate (α) and the base fee (β). In return for forwarding m BTC, a routing node will gain $\alpha \times m + \beta$ BTC as income (Figure 1).

4 Reinforcement Learning Prelimineries

For modeling the problem, we consider a partially-observable Markov decision process (POMDP) defined by the tuple $(O, S, A, P, r, \rho_0, \gamma)$, where S is the state space, A is the action space, $P : S \times A \times S \to [0, 1]$ is the transition probability distribution, $r : S \times A \to R$ is the reward function, $\rho_0 : S \to [0, 1]$ is the initial state distribution, and $\gamma \in [0, 1]$ is the discount factor. While interacting with the environment, an RL agent receives the observation $o(s) \in O$ at state $s \in S$. Then the agent has to take an action from A based on the observed $o(s)$. In most real-world scenarios, $o(s)$ does not necessarily contain all the information about state s which

makes it a partially observable environment. RL agents usually try to find a policy $\pi : S \times A \to [0, 1]$ that maximizes the expected return,

$$J(\pi) = E_{s_0 \sim \rho_0, s_t \sim P(.|s_{t-1}, a_{t-1}), a_t \sim \pi(.|s_t)} \big[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \big],$$

where $s_t \in S$ and $a_t \in A$ denote the state of the environment and the action taken by the agent at time step t, respectively.

5 LN Simulator

There are a few simulators for LN, such as the one introduced by Beres2021Cryptoeconomic, However non of them can be utilized for the fee-setting task. We have enhanced these simulators to be useful for our purpose. For the task of fee-setting, we implemented the **Lightning Experimental Virtual Network (LEViN)**, a realistic simulator that is able to generate random transactions, simulate routing, and calculate metrics of any selected node based on the results of its simulation. We described the simulator in full detail in *Appendix A*.

Data: Primary topological properties of LN, such as nodes, payment channels, channel fee policies, and channel capacities are available in a snapshot of the network [Rohrer *et al.*, 2019]. Any arbitrary snapshot of LN can be fed into LEViN if it adheres to a certain straightforward structure. Additional details of the snapshots and their structure are available in Appendix B.

Balance: Even though the capacities of the channels are accessible through the LN snapshots, the balances of the two peers of a channel are not available to the public. LEViN sets the balances of the two peers, each equal to half of the channel's capacity. It is also possible to set a channel's balances uniformly or even set them to an arbitrary amount. In the Experiments section, we will show that initial balances have a minor impact on a node's optimal income.

Generating transactions: Real data of transactions can not be extracted from the LN snapshots. For the sake of privacy, the publicly available data does not include information about the sender, receiver, and the number of transactions. Thereby, LEViN generates random transactions based on the distribution used in the LN Simulator. The sender of each transaction will be uniformly selected, while the receiver is more likely to be a merchant of the network. Any other arbitrary probability measure can be used to select a merchant. See Appendix A for all the details about generating the transactions.

Routing: Different methods have been proposed for the LN routing problem, namely [Hoenisch and Weber, 2018], [Prihodko *et al.*, 2016b], and [Wang *et al.*, 2019]. For routing transactions, we followed the LN Simulator's assumption of choosing the cheapest possible path. For routing each payment, we found the path with the lowest total fee among all other paths with enough liquidity. Further details can be found in Appendix A.

Localization: LEViN is also able to perform the simulation only for a few nodes close to a selected central node. This property is essential considering the significantly huge number of nodes in the LN and the limited resources available for the simulation. Users can specify the localization size as an input to LEViN. More details about the properties of localization size will be discussed in the Experiments section.

6 RL For The Fee-Setting Task

Assume that we want to achieve optimal income for the node v with k payment channels $c_1, \ldots, c_k$ connected to it. We propose a POMDP for modeling the problem as follows:

In the Experiments section, we will illustrate that only a small number of nodes close to v can have a noticeable effect on its income. Since the changes in the topology of the small neighborhood of v can be effectively slow, we assume the fee-setting problem to be fairly stationary. This allows us to propose a POMDP for modeling the problem as follows:

Observation Space: We define the observation space to be the set of nonnegative $2k$ dimensional vectors. At each time step t, we simulate random transactions in the localized network around v. In the observation vector $s_t = (b_1^t, \ldots, b_k^t, m_1^t, \ldots, m_k^t)$, b_i is the balance of the channel c_i and m_i is the accumulated amount of payments routed through c_i. Note that, LEViN computes these values when the tth round of simulations is completed.

Action Space: We define the action space as the set of all possible fee rates and base fees for the channels of node v. At the time step t, the agent chooses the action vector $a_t = (\alpha_1^t, \ldots, \alpha_k^t, \beta_1^t, \ldots, \beta_k^t)$ in which α_i^t and β_i^t are the fee rate and base fee of channel c_i, respectively. Later on, based on the $(\alpha_i^t)_{i=1}^k$ and $(\beta_i^t)_{i=1}^k$ the simulation will be performed, leveraging the dynamic fee property of LEViN.

Reward Function: Our goal is to maximize the income of node v. With regard to that, the reward function at time step t is defined by the formula

$$r(s_t, a_t) = \sum_{i=1}^{k} \alpha_i^t m_i^t + \beta_i^t n_i^t,$$

in which n_i^t is the number of transactions routed through c_i, provided by LEViN at the end of the time step.

We observed that no transaction will pass through v except for a small fraction of possible fee rates and base fees, resulting in a very sparse reward. For overcoming this problem, we limited the possible fee rates to less than 1000 and possible base fees to less than 10000 MSat. See Appendix C for the experiments supporting the sparsity claim.

The Challenges: There are three main challenges arising when trying to learn the optimal fee policy.

First, this environment is *partially observable* due to security reasons and also the complexity of the state space. The agent does not have any information about the transaction flow of other users. It also does not have any information about the current balances of other payment channels. As a result, the agent is not aware of whether the fee policy used by other nodes is optimal. partial observability will affect the estimated value function in the policy gradient methods.

Non-stationarity is another main issue in the LN fee-setting task. The network is constantly changing, and these changes directly affect the income of the selected node. Therefore, we need to use on-policy algorithms to use new data for each policy update. In the experiments, we will show that off-policy

Name	Index	Capacity (1m*Sat)	# Channels
a	71555	21	6
b	97851	21	6
c	109618	16	7
d	109681	93	42
e	135247	147	6
LN node avg.		25	9

Table 1: Five randomly selected nodes from LN data. a, b, and c are chosen to be similar to the LN average. d is chosen randomly from nodes with a number of channels in the fourth cortile of the LN nodes. e is chosen randomly from nodes with capacity in the fourth cortile of the LN nodes.

RL agents are not able to gain considerable income in comparison to on-policy RL agents. Additionally, in the Experiments section, we will illustrate that only a small number of nodes close to v can have a noticeable effect on its income. Since the changes in the topology of the small neighborhood of v can be effectively slow, we assume the fee-setting problem to be fairly stationary.

The third issue is the sparsity of the reward function. *Appendix C* shows that no transaction will pass through v except for a small fraction of possible fee rates and base fees, resulting in a very sparse reward. For overcoming this problem, we limited the possible fee rates to less than 1000 and possible base fees to less than 10000 MSat. Another key factor in sparsity is the localization parameter which will be discussed further in the experiment section.

7 Experiments

We ran three groups of experiments to address the following questions in order to further show the discussed difficulties of the fee-setting task and the effectiveness of DRL:

1. Evaluating the performance of the state-of-the-art RL algorithms,
2. The effect of sparsity on the performance of trained agents,
3. Whether RL methods can achieve better incomes compared to the other fee-setting baselines.

To answer (1), we compare two on-policy DRL algorithms on five different nodes (Table 1). To address (2), we train different PPO agents with three localization sizes. We show that smaller localization sizes lead to better performances, emphasizing the significance of a node's local topological characteristics in LN. Concerning question (3), our results show that trained agents can perform noticeably better than other available baselines. Running experiments with five different initial balances, we also see that RL agents can perform well regardless of the initializations. Next, we elaborate more on each of these questions in the following subsections.

7.1 Evaluation and Performance Results

Our primary goal in the following experiments is to demonstrate the good behaviors of on-policy DRL agents while interacting with LN fee-setting environment. Being able to gain considerable income requires learning a variety of challenging LN characteristics that were discussed before. These behaviors will get harder to learn as the capacity and number of channels increase, which in turn causes the dimension and complexity of the action space to grow. To show our approach *generalizes well to every node regardless of its topological properties*, we trained RL agents for five different random nodes, three with properties close to the average of LN, and two with highly complex action space (Table 1).

Figure 2 presents the training curves of two different on-policy DRL algorithms. We chose PPO [Schulman *et al.*, 2017] and TRPO [Schulman *et al.*, 2015] as two well-established representative methods, each trained for 1,000K time steps using five different random seeds. We set the localization size to 100 and considered equal initial balances for each channel's peer in all the experiments. We conducted our experiments based on Stable-baselines3 [Raffin *et al.*, 2021] implementations for RL algorithms. We refer interested readers to *Appendix A* for the details of the technical details and the experimental setups.

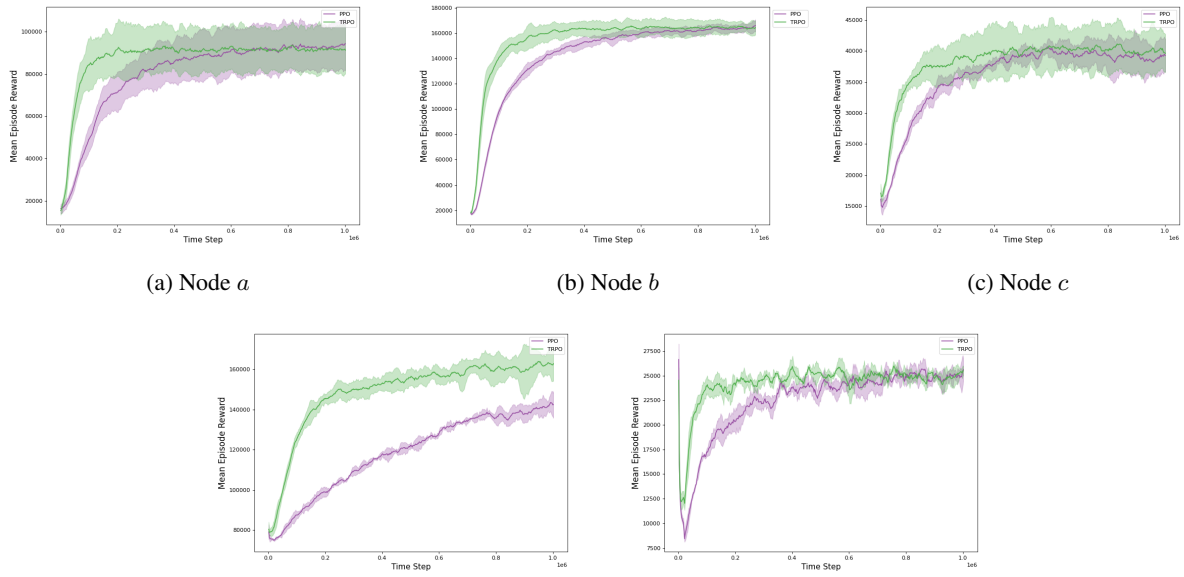
7.2 Sensitivity to Localization Parameter

As previously discussed, LN environments can be sparse based on the number of transactions that flow through the selected node at each time-step. Localization size is one of the key factors in this regard. One can see that as the localization size increases, each transaction has a higher chance of finding a cheaper well-funded route in the network that does not use the selected node, and we expect the reward function to quickly become sparser. One approach to keep up with the sparsity is to simulate many more transactions at each time step. This approach, however, will rapidly increase the complexity of the task at hand and is not guaranteed to fix the reward function's sparsity. *Appendix C* contains more information on the complexity and other studies with localization size.

To evaluate the importance of the localization size parameter, we trained three PPO models on the node b using localization sizes $L = 100, 250, 500$ with five different random seeds. Afterward, we tested the performance of the trained PPO agents on the LN environment with localization size $L = 500$. The results of this group of experiments are summarized in Figure 3. Our experiments show that the PPO model with a smaller localization size is gaining better incomes in comparison to the PPO model with a larger localization size, confirming the effects of the reward sparsity issue. This also indicates that the *local topological properties of the node* are directly affecting its income. Thereby, closer nodes can drastically change one's maximum gain. This highlights the need for a method that is able to simulate LN transactions and predict the maximum income depending on the position of the node. Turning this argument around will allow the users to decide on the nodes they want to establish a payment channel with, in order to optimize their future income.

7.3 Comparison to Heuristic Baselines

Figure 4 illustrates the performance of trained off-policy and on-policy DRL agents in comparison to three famous heuristic fee-setting baselines, namely, static, match peer,



(a) Node a (b) Node b (c) Node c

(d) Node d (e) Node e

Figure 2: Performance of two well-studied on-policy DRL algorithms on three nodes that are close to the LN average and two nodes with complex action spaces (Table 1). The convergence of TRPO and PPO are depicted by the red and purple curves, respectively. In addition to the mean episode rewards, the shaded areas show the standard error of five runs over different random seeds.

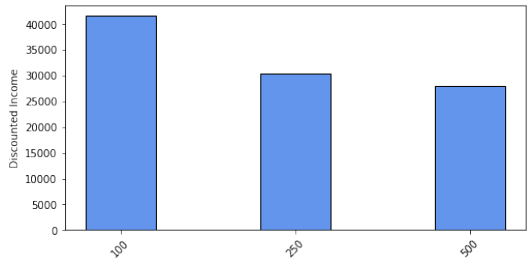


Figure 3: Performance of three trained PPO agents, each trained on $L = 100, 250, 500$. Testing is done with localization size $L = 500$.

and proportional strategies.

Static: In this strategy, the node's fee policy is to choose fixed fee rates and base fees during the whole episode. We used the node's real fee rates and base fees from the LN snapshot to conduct our experiments. As Figure 4 suggests, current fee policies will result in almost negligible income, which is in agreement with the recent observations of the real-world LN. For static fee policies with different base fees and fee rates refer to Appendix C.

Match peer: When using the match peer strategy, the fee policies of the node's payment channels are determined to match the fee policies of the other peer of each channel. Since the topological properties of two peers of a channel are close, one can expect a good fee policy for one to be also suitable for the other. Right now, since the income of most of the users in LN is insignificant, this idea is less likely to work.

Proportional: This strategy is based on the trade-off between channel balance and channel fee rate, *i.e.*,

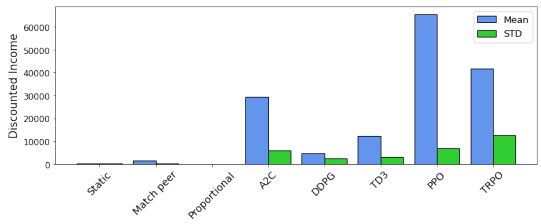
$$\beta = 0, \quad \alpha \sim 1 - \frac{\text{balance}}{\text{capacity}}.$$

The idea behind this strategy is that with a higher fee rate, fewer payments will route through the channel, resulting in more stable balances. On the other hand, when having enough balances, using lower fee rates will result in more payments routing through the channel. We found the proportional strategy to be very sensitive to the initial balances, and it results in zero income for most of the initializations. For further details, refer to Appendix C.

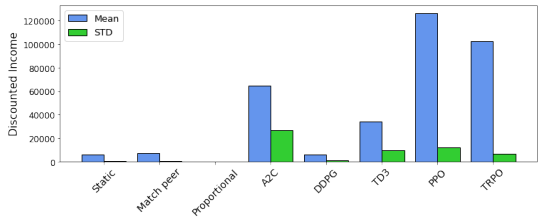
We used five different random initial balances for conducting these experiments. As illustrated in the plots, one can see a significant improvement when using DRL methods. PPO as our best model can gain almost ten times more income compared to the other strategies with low sensitivity to the initial channel balances.

In addition to PPO and TRPO, we conducted experiments for A2C, TD3, and DDPG to further understand the performance of off-policy algorithms in this setup. The differences between the gained income of on-policy and off-policy DRL agents confirm the critical importance of using on-policy methods while facing non-stationarity. The learning curve for off-policy algorithms is available in Appendix C.

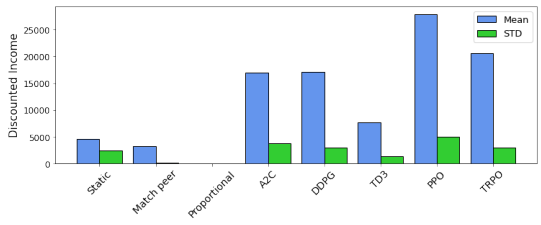
To conclude the experiments, our results on consistent and fast convergence of PPO (Figure 2), its stability despite changing the initial balances (Figure 4), and its superior



(a) Node a



(b) Node b



(c) Node c

Figure 4: Episodic discounted income of three different baselines compared to the trained DRL agents. We used five random initial balances for each strategy.

performance while using smaller localization sizes (Figure 3) provide empirical evidence that DRL algorithms are able to solve the fee-setting problem better than any other currently proposed heuristic methods, which also demonstrates the possibility of using RL in other real-world financial problems.

8 Conclusions & Future Work

In this work, we presented LEViN, a highly optimized simulator for off-chain payment channel networks such as the Bitcoin LN with several novel features targeted toward customizing and evaluating the dynamic fee mechanism. Next, we proposed an RL-based approach built on top of LEViN which learns the difficult but not intractable reference task of finding the optimal fee policy. Furthermore, the performance of several well-established DRL algorithms is studied and compared to the heuristics currently used in practice as baselines. Overall, this work can provide an interface to learn a policy from a snapshot of LN to be used in action. We showed that on-policy RL algorithms are the clear winner with a significantly higher income based on our initial experimental results, but further work is needed to implement and test the derived policies in a real-world scenario.

Several questions can be further investigated following this work, e.g., multi-agent RL can be used to deploy an optimal policy for a group of nodes with varying topological features, contrary to our method which focused on the payment channels of a single node. Moreover, the transaction fee problem in LN can be viewed as the inverse problem of the routing fee under some conditions, which creates new paths for tackling this problem. The software support for LN research can be heavily enhanced by implementing novel simulators for LN that make the simulations as realistic as possible. Besides that, gathering datasets of LN topology and transaction flow which are practical and easy-to-use in simulators can be very valuable for future research and add more functionalities geared towards other hard tasks which arise in layer 2 of blockchain protocols.

Ethical Statement

There are no ethical issues.

Acknowledgments

The preparation of these instructions and the LaTeX and BibTeX files that implement them were supported by Schlumberger Palo Alto Research, AT&T Bell Laboratories, and Morgan Kaufmann Publishers. Preparation of the Microsoft Word file was supported by IJCAI. An early version of this document was created by Shirley Jowell and Peter F. Patel-Schneider. It was subsequently modified by Jennifer Ballentine, Thomas Dean, Bernhard Nebel, Daniel Pagenstecher, Kurt Steinkraus, Toby Walsh, Carles Sierra, Marc Pujol-Gonzalez, Francisco Cruz-Mencia and Edith Elkind.

References

[Bartolucci *et al.*, 2020] Silvia Bartolucci, Fabio Caccioli, and Pierpaolo Vivo. A percolation model for the emergence of the bitcoin lightning network. *Scientific reports*, 10(1):1–14, 2020.

[Béres *et al.*, 2021] Ferenc Béres, István András Seres, and András A Benczúr. A Cryptoeconomic Traffic Analysis of Bitcoin's Lightning Network. *Cryptoeconomic Systems*, 0(1), apr 5 2021. https://cryptoeconomicsystems.pubpub.org/pub/beres-lightning-traffic.

[Bertucci, 2020] Louis Bertucci. Incentives on the lightning network : A blockchain-based payment network. 2020. Proceedings of Paris December 2020 Finance Meeting EUROFIDAI - ESSEC, Available at SSRN: https://ssrn.com/abstract=3540581 or http://dx.doi.org/10.2139/ssrn.3540581.

[Conoscenti *et al.*, 2021] Marco Conoscenti, Antonio Vetrò, and Juan Carlos De Martin. Cloth: A lightning network simulator. *SoftwareX*, 15:100717, 2021.

[Dai *et al.*, 2019] Yueyue Dai, Du Xu, Sabita Maharjan, Zhuang Chen, Qian He, and Yan Zhang. Blockchain and deep reinforcement learning empowered intelligent 5g beyond. *IEEE Network*, 33(3):10–17, 2019.

[Di Stasi *et al.*, 2018] Giovanni Di Stasi, Stefano Avallone, Roberto Canonico, and Giorgio Ventre. 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), pages 1161–1170, 2018.

[Hoenisch and Weber, 2018] Philipp Hoenisch and Ingo Weber. Aodv–based routing for payment channel networks. In *International Conference on Blockchain*, pages 107–124. Springer, 2018.

[Jameel *et al.*, 2020] Furqan Jameel, Uzair Javaid, Wali Ullah Khan, Muhammad Naveed Aman, Haris Pervaiz, and Riku Jäntti. Reinforcement learning in blockchain-enabled iiot networks: A survey of recent advances and open challenges. *Sustainability*, 12(12), 2020.

[Jun Ren *et al.*, 2018] Alvin Heng Jun Ren, Ling Feng, Siew Ann Cheong, and Rick Siow Mong Goh. Optimal fee structure for efficient lightning networks. In *2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS)*, pages 980–985, 2018.

[Martinazzi and Flori, 2020] Stefano Martinazzi and Andrea Flori. The evolving topology of the lightning network: Centralization, efficiency, robustness, synchronization, and anonymity. *Plos one*, 15(1):e0225966, 2020.

[Musacchio and Wu, 2007] John Musacchio and Shuang Wu. The price of anarchy in a network pricing game. In *Proc. of 45th Annual Allerton Conference*, 2007.

[Nakamoto, 2008] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Decentralized Business Review*, page 21260, 2008.

[Poon and Dryja, 2016] Joseph Poon and Thaddeus Dryja. The bitcoin lightning network: Scalable off-chain instant payments, 2016.

[Prihodko *et al.*, 2016a] Pavel Prihodko, S. N. Zhigulin, Mykola Sahno, A B Ostrovskiy, and Olaoluwa Osuntokun. Flare : An approach to routing in lightning network white paper. 2016.

[Prihodko *et al.*, 2016b] Pavel Prihodko, Slava Zhigulin, Mykola Sahno, Aleksei Ostrovskiy, and Olaoluwa Osuntokun. Flare: An approach to routing in lightning network. *White Paper*, page 144, 2016.

[Raffin *et al.*, 2021] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.

[Rohrer *et al.*, 2019] Elias Rohrer, Julian Malliaris, and Florian Tschorsch. Discharged payment channels: Quantifying the lightning network's resilience to topology-based attacks. jun 2019.

[Roos *et al.*, 2017] Stefanie Roos, Pedro Moreno-Sanchez, Aniket Kate, and Ian Goldberg. Settling payments fast and private: Efficient decentralized routing for path-based transactions. *arXiv preprint arXiv:1709.05748*, 2017.

[Schulman *et al.*, 2015] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR, 2015.

[Schulman *et al.*, 2017] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[Seres *et al.*, 2020] István András Seres, László Gulyás, Dániel A Nagy, and Péter Burcsi. Topological analysis of bitcoin's lightning network. In *Mathematical Research for Blockchain Economy*, pages 1–12. Springer, 2020.

[Sutton and Barto, 2018] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[Wang *et al.*, 2019] Peng Wang, Hong Xu, Xin Jin, and Tao Wang. Flash: efficient dynamic routing for offchain networks. In *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies*, pages 370–381, 2019.