ReLEaRN: Reinforcement Learning Enhanced Profitable Rebalancing in Payment Channel Networks

Shruti Mishra*, Mohit Bhuria†, Hitesh Singla‡, Sujata Pal§, Isaac Woungang¶
*†‡§ Department of Computer Science and Engineering, Indian Institute of Technology Ropar, India
Emails: shruti.22csz0011@iitrpr.ac.in*, 2021csb1109@iitrpr.ac.in†, 2021csb1094@iitrpr.ac.in‡, sujata@iitrpr.ac.in§
¶Department of Computer Science, Toronto Metropolitan University, Canada
Email: iwoungan@torontomu.ca¶

***Abstract*—Payment Channel Networks (PCNs) act as a foundational layer in blockchain and deal with the scalability issues. However, a major challenge that continues to hinder the worldwide acceptance of PCNs is the transaction failure rate caused by channel dependency. This depletion occurs due to the dependency of PCNs on network topology. Due to this, effective node placement is crucial for establishing economical channels and strengthening network robustness. On one hand, node attachment is necessary to bring back the PCNs to a balanced state, while on the other hand, the network is prone to getting trapped in a local optimum. To address these challenges, ReLEaRN is proposed as a node placement strategy that deals with the rebalancing problem of PCNs and the local optimum problem using the Soft-Actor critic (SAC) algorithm. We simulated our proposed algorithm on topologies of the Lightning Network and achieved 90% improvement in execution time when compared with ProfitPilot. The execution time is similar to basic heuristic topologies present in the Lightning Network, but the probability of fees collection is improved to 40% compared to these topologies.**

***Index Terms*—Blockchain, Reinforcement Learning, Payment Channel Networks, Soft Actor-Critic, Optimization, Cycle Creation, Network Rebalancing.**

I. Introduction

Digital cryptocurrencies [1] revolutionized digital finance by enabling decentralized and trustless monetary transactions using blockchain [2], [3] technology. Bitcoin [4] and Ethereum [5] are implemented with these decentralized systems. However, these decentralized systems face the challenges of on-chain transactions, such as low throughput and high latency. To address these challenges, Payment channel networks (PCNs) [6] are introduced as a scalability solution. PCNs allow users to perform off-chain transactions, thereby reducing latency and improving scalability.

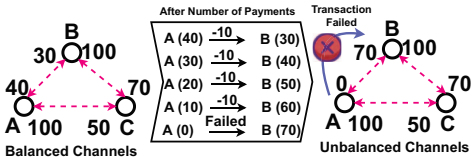


Fig. 1. A → B channel is closed due to imbalance.

These transactions between users are done using a contract known as the Hash Time Lock Contract (HTLC). These contracts are signed by both users to ensure the completion of payment, thus creating a payment channel between those users. Funds are locked by users using these contracts so that transfer of funds can be carried out in both directions.

Fig. 1 shows the funds stored by three users in the network. User A transacts multiple payments towards B. Over time, as transactions flow in a particular direction, the funds of user A become 0 and the channel becomes unbalanced from A → B. This unbalanced channel restricts the network's routing capabilities and lowers transaction success rates.

To overcome this imbalance, rebalancing is carried out by restoring channel liquidity by creating self-transfer and maintaining the operability of the network [7]. The key idea is to keep the channels active that are going to deplete in near future as shown in Fig. 2.

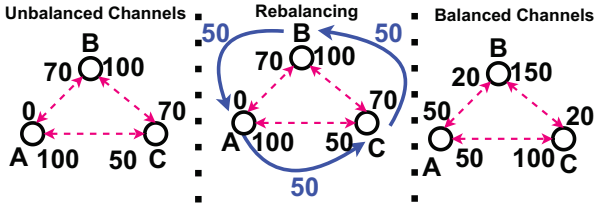


Fig. 2. Self-transfer Rebalancing

The traditional process of rebalancing relies on cycle creation, where multiple nodes do not participate. Profitpilot [8] proposed a node attachment strategy for cost-effective rebalancing and enhancing the robustness of PCNs. However, the strategy is focusing on static node attachment and does not dynamically adhere to real-time changes in the network, which limits responsiveness to fluctuating conditions in PCNs.

ReLEaRN addresses the shortcomings of ProfitPilot and enhances the attachment strategy using reinforcement learning, in a dynamic environment. RL is a branch of Machine Learning (ML) [9], [10] enables an agent to learn sequential

979-8-3503-9281-4/25/$31.00 ©2025 IEEE

decision-making by interacting with the environment. The RL algorithms like SAC [11] demonstrated superior performance in continuous and high-dimensional action spaces.

Motivated by this, our proposed algorithm leverages SAC to optimally attach new nodes and create profitable cycles for rebalancing. Each node acts as an RL agent and observes the current state of the network. The actions are selected by these nodes such as which neighbors to attach to or which cycles to create. These nodes receives rewards based on transaction success, fees, and network rebalancing. The node attachment and cycle creation process is structured as a Markov Decision Process (MDP) which is used to model sequential decision problems in RL. ReLEaRN makes principled, sequential decisions rather than relying on static heuristics. The use of SAC enables ReLEaRN to efficiently learn optimal strategies in continuous and high-dimensional action spaces. The salient features of ReLEaRN are summarized as follows:

- ReLEaRN enables profitable cycles using node attachment mechanism to maximize payment channel balance and minimize time taken to rebalance across different network sizes.
- ReLEaRN operates on a dynamically extracted logical subgraph by reducing the state and action space to improve scalability while preserving rebalancing accuracy.
- ReLEaRN presents rebalancing problem into a Markov Decision Process, enabling stable, offline training and efficient policy optimization without interfering with real-time network operations.
- We evaluate the performance of ReLEaRN over the historical data of real-world PCNs. ReLEaRN restore PCNs to a more balanced state by reducing execution time as compared with state- of-the-art rebalancing algorithms.

The remainder of this article is organized using the following structure. The related works are closely reviewed in Section II. Section III presents the system model and assumptions. Section IV discusses algorithm design. Section V explains the simulation setup and results obtained and finally, Section VI concludes the paper.

II. Related Work

PCNs are becoming more popular and gaining attention as a scalable blockchain. Routing is one of the challenging problem in PCNs and multiple solutions are developed to deal with this challenge. As an illustration, HeRo [6] is proposed as a heuristic based routing algorithm for PCNs while Swift routing [12] is using secret path routing for fee optimization. AMORA [13] is proposed a multi-objective routing algorithm using evolutionary algorithms. However, none of the method deals with depletion of channels while doing transactions, which causes transaction failures in the network.

In response to this, multiple Groundbreaking studies proposed for PCNs rebalancing. Varma and Maguluri [14] proposed a stochastic model for routing and used two-sided queues for rebalancing. DRL-PCR [15] is using RL method to rebalance existing channels. maxREE [16] introduces the idea of replacing the edge for channel rebalancing using edge coloring method. Optimal channels are preserved and congestion at the landmark node is reduced.

Lange et al. [17] discussed multiple attachment strategies such as random, highest degree, k-centre, k-media. These discussed strategies optimizes network robustness and decentralization but routing efficiency and fees like metrics are ignored. Highest degree strategy creates the risk of centralization while k-median and k-centre strategies are computationally intensive. Ersoy et al. [18] make use of greedy algorithm for establishing channels that maximize the node's total expected reward. The authors increases the number of shortest paths passing through the node, which increases potential routing fees. However, greedy strategy chooses one channel at a time thus making early decisions. These early decisions limit future opportunities. They also prove the attachment strategy as NP-hard, so exact solutions are infeasible for large networks. All of these methods rebalance existing skewed network without addressing the underlying limitations of the topology.

ReLEaRN guarantees to improve both the network topology and liquidity distribution using SAC. SAC provides a flexible and effective way to adapt to complex and evolving network dynamics like PCNs. The algorithm is scalable to larger PCNs because reinforcement learning identifies optimal connection strategies. The topology is strategically expanded to enable better future rebalancing.

III. System Model

A. PCNs Model

PCN is modeled as a connected directed graph $G(V, E)$ where V, E represents the set of users and set of channel between those users with amount (amt). PCNs channels are bidirectional such that for each $u, v \in V$ there is an edge such that $(u, v) \in E \Leftrightarrow (v, u) \in E$. The balance matrix $amt \in \mathbb{R}_0^{+(|V|\times|V|)}$ defines the available liquidity along directed edges. $A(u, v) \in \mathbb{R}_0^+$ represents the maximum amount of funds that can be transferred from user u to user v through the channel for each pair of nodes $(u, v) \in E$. The amt is updated after each successful transactions. The intermediary nodes helping in carrying out transactions charge fees f consists of two parts: a base fee $b(f)$ and a proportional fee $Pr(f)$. $Pr(f)$ vary with each transaction depending upon the amount A while $b(f)$ is kept constant for each transaction. The total fee for every transaction tr thus calculated as $f^{tr}_{(u,v)}(amt) = b_{(u,v)}(f) + Pr^{amt}_{(u,v)}(f)$. The notaions used in the paper are summerized in Table I

B. Assumptions

ReLEaRN assumes that new node attaching to the network is aware of the topology used before initiating the attachment strategy. The assumption is aligned with explorer of lightning [19] and Raiden [20] network where all publicly announced nodes, payment channels, their capacities, and associated fee policies are displayed. The SAC agent takes the attachment decisions deterministically and predictably during node attachment, resulting in PCNs changing topology. However, no other external events interfere with the network during

TABLE I
NOTATIONS

Notation	Description
G	Graph for PCN
S	Set of states
A	Set of actions
$P(s_{t+1}\|s_t, a_t)$	State transition probability
$R(s_t, a_t)$	Reward function
$\pi_\theta(a\|s)$	Probability of choosing action a given state s
$J(\pi)$	Expected total utility using policy π
D	Replay buffer
$Q(s, a)$	Expected cumulative reward
$N_{optimal}$	Set of final nodes

the decision-making and attachment process. This assumption is reasonable because substantial network changes require on-chain confirmations, which are relatively slow compared to the ReLEaRN operation.

Node attachment strategy follows the restrictions of valid, non-duplicate, and non-self-loop connections. A continuous action space is managed for the SAC agent operations, attachment preferences based on normalized centrality features of the network. Liquidity improvements, cycle formation, and penalties for invalid actions are considered for reward computations. Training is carried out offline within a deterministic environment using an experience replay buffer, and graph updates are immediately reflected after each attachment. A fixed fee structure is assumed for all new channels and focus on maximizing long-term cumulative rewards under these operational constraints.

IV. Algorithm Design

The PCNs suffer from channel imbalances and decreasing routing efficiency as transactions proceed, especially when payment demands are non-uniform across the network. To balance the PCNs, ReLEaRN proposes a dynamic rebalancing model based on node attachment strategies, utilizing reinforcement learning to enhance network robustness. The proposed algorithm allows a new node to strategically attach itself to existing nodes within the PCN. This attachment uses a learned policy, trained using a Soft Actor-Critic (SAC) agent. The agent dynamically adapts to the network's evolving topology.

ReLEaRN operates independently and locally, allowing the new node to autonomously decide its attachment strategy. The network features such as closeness centrality, betweenness centrality, and node degree are observed by SAC agent. These features are normalized between [0, 1] and output is represented as a continuous action vector representing attachment preferences. The reward function depends upon three factors: *1).* Liquidity improvement, measured by gains in global centrality and reachability, *2).* Cycle formation, rewarding the creation of new triangles and routing opportunities, *3).* Penalties for invalid actions, such as attempting to create duplicate or self-connections.

A. MDP Formulation for ReLEaRN

PCN node attachment and cycle creation problem is transformed into a sequential decision-making process. At the beginning, a PCN $G = (V, E)$ and new nodes n are given. The agent selects an existing node n^* to which n attaches an edge, based on the learned policy at each step. A profitable cycle creation phase is executed after k attachment operations, adjusting the capacities of selected channels. This sequential structure of ReLEaRN is compatible with a Markov Decision Process (MDP) model.

An MDP is defined by a four-tuple $\langle S, A, P, R\rangle$, where S is the set of states, A is the set of actions, $P(s_{t+1}|s_t, a_t)$ is the state transition probability, and $R(s_t, a_t)$ is the immediate reward function. The components of MDP are described as follows:

State (s_t): s_t represents the extracted state features at time step t from the current PCN G. The netowrk properties such as node degrees, betweenness centrality, channel balances, and the remaining attachment budget are captured at each step.

Action (a_t): The action a_t at step t is the selection of a target node $n^* \in V$ to which the new node n will attach an edge. In practice, the SAC agent outputs a probability distribution over candidate nodes, from which an action is sampled.

Reward ($R(s_t, a_t)$): A reward is computed based on three factors: a)improvement in centrality (ΔC) of the node n, b) increase in network transaction throughput (ΔT), c) penalty for cost or adverse impact on network balance (P). The reward is calculated as:

$$R = \beta_1 \Delta C + \beta_2 \Delta T - \beta_3 \Delta P, \tag{1}$$

where $\beta_1, \beta_2, and \beta_3$ represents tunable weights and reflects the trade-off preferences.

State Transition Probability (P): The transitions are deterministic in this environment. The graph is updated after taking an action, by adding the edge (n, n^*), and new features are extracted to form s_{t+1}. The graph changes are depends solely on current state-action pair (s_t, a_t). Since, the evolution process follows deterministic rule, the resulting Markov Decision Process (MDP) is also deterministic.

The deterministic environment is maintained for offline training, using experience replay buffers to stabilize learning. The stochastic policy is optimized by SAC agent. This policy maximizes the cumulative reward under fixed operational constraints, such as a predetermined maximum number of allowed attachments and channel fee structures.

The objective of ReLEaRN is to maximize the cumulative discounted reward over k steps is:

$$arg \max_{\pi} \mathbb{E}\left[\sum_{t=0}^{k-1} \gamma^t R(s_t, a_t)\right], \tag{2}$$

where $\gamma \in (0, 1]$ is the discount factor emphasizing future rewards.

B. ReLEaRN Overall Architecture

Now, lets dive into overall architecture and working of ReLEaRN. ReLEaRN contains three major stages: (i) feature extractor, (ii) SAC-Based node attachment and (ii) profitable cycle creation as shown in Fig. 3.

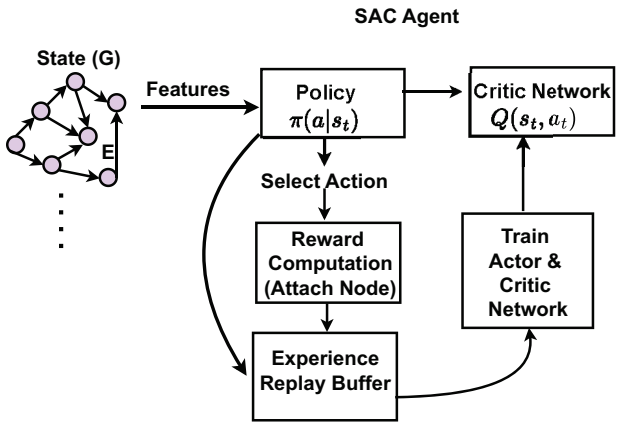


Fig. 3. System Architecture of ReLEaRN

The node attachment problem is treated as an interactive agent-environment process by ReLEaRN. The process begins with extraction of features from PCN topology. The features computed for each node are degree centrality, closeness centrality, and betweenness centrality. The PCN environment provides initial state s_t reflecting the network topology and node properties at each step t to the SAC agent as disussed in Algorithm 1. The policy $\pi_\theta(a|s)$ provides appropriate action vector, and provides attachment priority over candidate nodes.

Algorithm 1 SAC-Based Node Attachment Algorithm

Input: $G = (V, E)$, n, k, α, β_1, β_2, β_3
Output: N_{optimal}, G'

Initialize $N_{\text{optimal}} \leftarrow \emptyset$
Initialize SAC agent with actor $\pi(a|s)$ and critic $Q(s, a)$
while $|N_{\text{optimal}}| < k$ **do**
  $s_t \leftarrow$ Extract features from G
  $a_t \sim \pi(a|s_t)$
  $n^* \leftarrow \arg\max_{n_i} \pi(a|s_t)$
  if Valid attachment (n, n^*) **then**
    Simulate adding (n, n^*) to G
    Compute ΔC, ΔT, P
    Calculate R using Equation 1.
    if $R > 0$ **then**
      Permanently add (n, n^*) to G
      $N_{\text{optimal}} \leftarrow N_{\text{optimal}} \cup \{n^*\}$
      Update SAC with (s_t, a_t, R, s_{t+1}) using Equation 2.
    end if
  else
    Apply penalty P
  end if
end while
Return N_{optimal}, G'

The action a_t provided by policy is deterministically selected for attaching the node. PCN simulates the action by creating a new payment channel between the selected node and existing node. Each time a new channel is created, PCNs topology is updated to reflect the new attachment. The updated PCN is used to extract a new state s_{t+1} for the next decision. The reward $R(s_t, a_t)$ computation is done for each attachment based on improvements in centrality measures, formation of profitable cycles, and penalties for invalid or redundant actions. The experience replay buffer (D) stores the tuple (s_t, a_t, R, s_{t+1}) for SAC training.

Training is provided following SAC framework with the objective of maximizing expected cumulative reward while encouraging policy entropy ($J(\pi)$):

$$J(\pi) = \mathbb{E}_{(s,a)\sim D}\left[Q(s, a) - \alpha \log \pi(a|s)\right], \tag{3}$$

where α is the temperature parameter controlling the trade-off between reward maximization and exploration and $\mathbb{E}_{(s,a)\sim D}$ is the expectation over replay buffer D. $N_{optimal}$ is the final set of nodes that are attached during SAC-based node attachment phase.

In the final stage, ReLEaRN increases routing efficiency through profitable cycle creation using Algorithm 2. The primary focus is to create triangular cycles reduces network imbalance. ReLEaRN searches for each attached neighbor node $n_i \in N_{optimal}$ existing paths connecting n_i back to n intermediate nodes. The profitable cycle is found by adjusting corresponding channel capacities. Channel adjustments are simulated to support the cycle, strengthening the network's ability to process multi-hop payments efficiently.

Algorithm 2 Profitable Cycle Creation

Input: $G' = (V, E)$, n, N_{optimal}
Output: G'

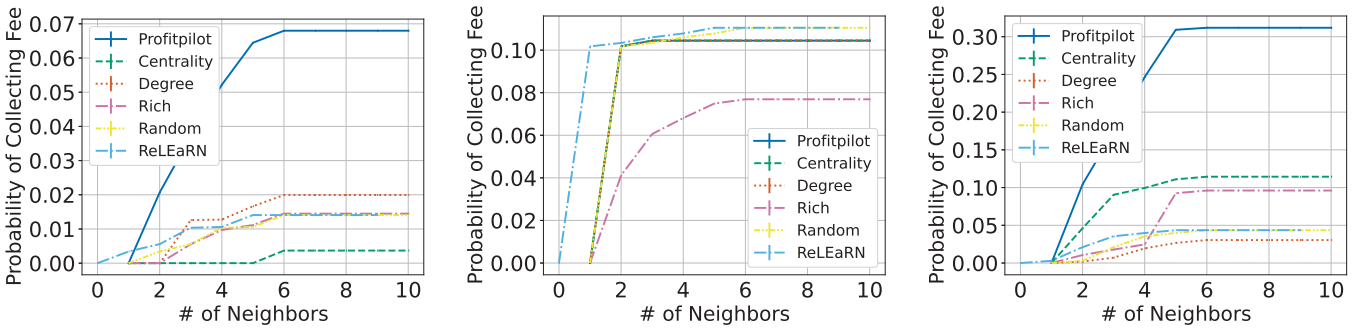
for each $n_i \in N_{\text{optimal}}$ **do**
  Identify potential cycles involving n, n_i, and other nodes
  if Cycle is profitable **then**
    Adjust channel capacities to support the cycle
  end if
end for
Return G'

This three stage architecture improves attachment strategy using SAC, then optimizing cycles through graph-based operations. ReLEaRN not only improves a node's individual connectivity but also systematically enhances the global liquidity and robustness of the entire PCN.

V. Simulations and Results

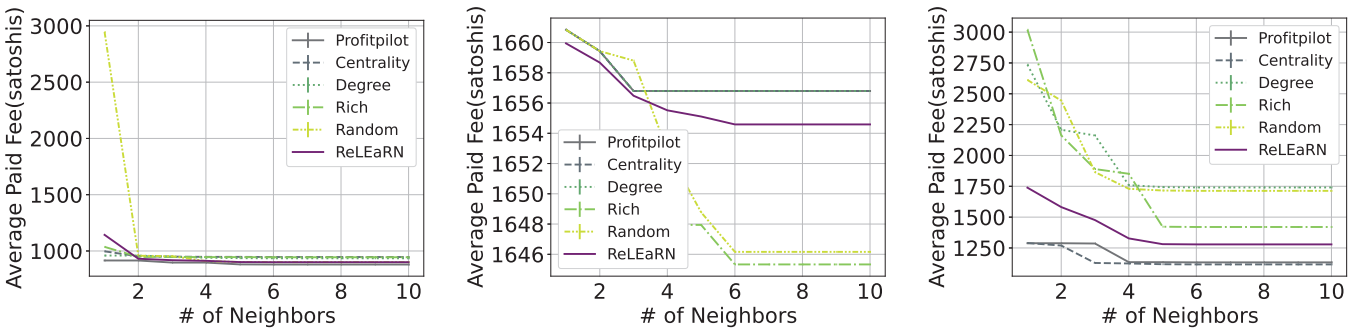
A. Simulation Setup

We implement ReLEaRN and evaluated the simulations on PCNs environment. The evaluations are carried on a desktop with an Intel Core i7-7700 4.20 GHz CPU and 16GB memory. The network is simulated using NetworkX package. We vary number of neighbors of a node to evaluate the performance of the SAC-based node attachment algorithm. The simulations



(a) Barabasi-Albert Topology (b) Lightning Network Topology (c) Watts-Strogatz Topology

Fig. 4. Fee Collection Probabilities Under Evaluated Topologies



(a) Barabasi-Albert Topology (b) Lightning Network Topology (c) Watts-Strogatz Topology

Fig. 5. Average Fees Paid for Evaluated Topologies

are carried out using Python and integrated with reinforcement learning frameworks such as PyTorch and TensorFlow. Key parameters included network size (ranging from 100 to 5000 nodes), maximum channels per node (k), and varying reward weights (β_1, β_2, β_3). We construct the network graph using node and channel announcement messages, declared by nodes publicly on the network. ReLEaRN is evaluated using LN, Barabasi-Albert [21] graph and Watts-Stogratz [22] graph. Barabasi-Albert graph is a scale free topology, Watts-Stogratz is a small-world topology, and Lightning network include scale free and small-world topologies.

ReLEaRN is compared with different attachment methods [23] as follows:

Random: This heuristic method selects nodes uniformly from a uniform distribution following the Erdős-Rényi [24] model.

Centrality: Nodes are selected with probabilities distribution function and selected as a potential neighbors. The distribution is weighted by their betweenness centrality scores.

Rich: Nodes with greater liquidity are filtered and the heuristic method samples uniformly from this group. The nodes are selected according to a probability distribution function proportional to their total capacity in the network.

Degree: The nodes are selected with probabilities proportional to their degree in this method.

ProfitPilot: Nodes are selected based on a based on a probability distribution that is weighted by their betweenness centrality scores. 3-node cycles are created for efficient payment routing and rebalancing while promoting network robustness and counteracting centralization trends.

B. Results

We simulate the PCNs for cycle creation approach using all the above mentioned attachment heuristics. We scale the LN with different number of nodes. The training of the actor-critic network is performed offline. ReLEaRN learns the optimal action by using lightning network topology through offline interactions with the environment. The learning is stable for SAC agent as the MDP formulated is deterministic. After training, node attachment takes place. Fig. 6 shows that execution time for node attachment taken by ReLEaRN is similar to all basic channel management strategies.

However, Fig. 4a shows that the probability of collecting fees by ReLEaRN is higher than other methods in lightning network topology. The fee collection probability is less than Profitpilot in Barabasi-Albert topology and Watts-Strogatz topology as shown in Fig. 4a and 4c because ReLEaRN relies on state exploration and temporal learning. These two topologies exhibits non-uniform degree distributions where our algorithm struggle to generalize quickly. The average fee paid is by the nodes is reduced by our proposed algorithm as shown in Fig. 5 as compared to profitpilot in lightning network topology.

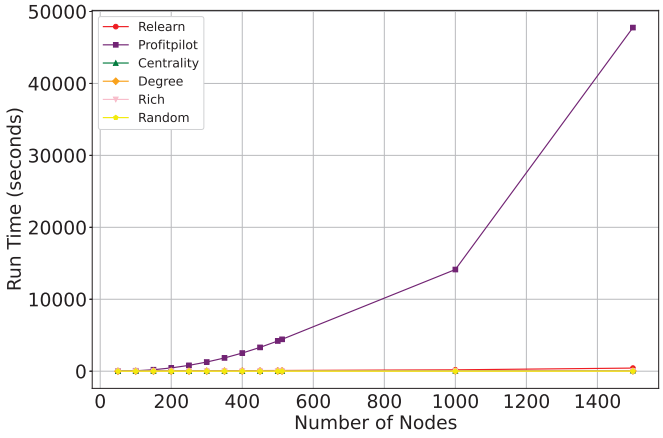


Fig. 6. Execution Time Across Different Network Sizes

VI. Conclusion

In this work, we studied the rebalancing challenge of PCNs. ReLEaRN is proposed as a rebalancing algorithm using a node attachment strategy for new nodes. The attachment strategy introduced a reinforcement learning-based solution for node attachment optimization in PCNs. By leveraging the Soft Actor-Critic (SAC) algorithm and integrating a profitable cycle creation mechanism, the proposed method outperforms traditional algorithms in terms of execution time, probability of collecting fees, and fees paid by the source nodes for transactions. This approach not only enhances network connectivity but also ensures balanced channel capacities. The successful implementation and evaluation of ReLEaRN demonstrate its potential for real-world applications in large-scale PCNs. Future work will explore the integration of multi-agent reinforcement learning to further enhance optimization in dynamic and highly decentralized environments. Additionally, incorporating real-world transaction data and varying network conditions would provide deeper insights into the algorithm's practical applicability and resilience.

References

[1] U. Mukhopadhyay, A. Skjellum, O. Hambolu, J. Oakley, L. Yu, and R. Brooks, "A brief survey of cryptocurrency systems," in *Conf. on Privacy, Security and Trust (PST)*, pp. 745–752, 2016.

[2] V. Agarwal, S. Mishra, and S. Pal, "Towards a sustainable blockchain: A peer-to-peer federated learning based approach," *ACM Transactions on Internet Technology*, vol. 24, no. 4, pp. 1–26, 2024.

[3] B. Jebari, K. Ibrahmi, M. Ghogho, and H. Tembine, "Bivo-a decentralized oracle solution for data authenticity in blockchain-based iot networks," *IEEE Internet of Things Journal*, vol. 12, no. 6, pp. 7259–7276, 2024.

[4] S. Nakamoto, "Bitcoin whitepaper," *White Paper*, pp. 1–15, 2008.

[5] V. Buterin, "Ethereum white paper," *GitHub repository*, pp. 1–46, 2013.

[6] S. Mishra, V. Aggarwal, S. Pal, and V. Agarwal, "Hero: Heuristic-based routing in payment channel networks," *IEEE Networking Letters*, vol. 7, no. 1, pp. 56–60, 2025.

[7] N. Sharma and K. Kapoor, "maxree: Maximizing flow by replacing exhausted edges in lightning network," *IEEE Transactions on Network Science and Engineering*, vol. 12, no. 2, pp. 1011–1025, 2025.

[8] G. F. Camilo, G. A. F. Rebello, L. A. C. de Souza, M. E. M. Campista, and L. H. M. K. Costa, "Profitpilot: Enabling rebalancing in payment channel networks through profitable cycle creation," *IEEE Transactions on Network and Service Management*, vol. 21, no. 3, pp. 3167–3178, 2024.

[9] S. Mishra and S. Pal, "Ear: An energy efficient human activity recognition from wearable devices," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, pp. 1826–1831, 2023.

[10] S. Mishra, S. Pal, and S. Dey, "Blast: Blood clot classification model using transfer learning based convolutional neural network," in *Proc. Int. Conf. on Communication Systems and Networks (COMSNETS)*, pp. 80–85, 2024.

[11] I. Budhiraja, N. Kumar, H. Sharma, M. Elhoseny, Y. Lakys, and J. J. P. C. Rodrigues, "Latency-energy tradeoff in connected autonomous vehicles: A deep reinforcement learning scheme," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 11, pp. 13296–13308, 2023.

[12] N. Sharma, K. Kapoor, and V. Anirudh, "Design and evaluation of swift routing for payment channel network," *Blockchain: Research and Applications*, vol. 5, no. 2, pp. 1–11, 2024.

[13] S. Mishra and S. Pal, "Amora-adaptive multi-objective routing algorithm in payment channel networks," *IEEE Transactions on Network and Service Management*, pp. 1–12, 2025. Last Accessed: August 30, 2025. [Online]. Available: https://doi.org/10.1109/TNSM.2025.3597407.

[14] S. M. Varma and S. T. Maguluri, "Throughput optimal routing in blockchain-based payment systems," *IEEE Transactions on Control of Network Systems*, vol. 8, no. 4, pp. 1859–1868, 2021.

[15] W. Chen, X. Qiu, Z. Cai, B. Tang, L. Du, and Z. Zheng, "Graph neural network-enhanced reinforcement learning for payment channel rebalancing," *IEEE Transactions on Mobile Computing*, vol. 23, no. 6, pp. 7066–7083, 2024.

[16] N. Sharma and K. Kapoor, "maxree: Maximizing flow by replacing exhausted edges in lightning network," *IEEE Transactions on Network Science and Engineering*, vol. 12, no. 2, pp. 1011–1025, 2025.

[17] K. Lange, E. Rohrer, and F. Tschorsch, "On the impact of attachment strategies for payment channel networks," in *Proc. IEEE Int. Conf. on Blockchain and Cryptocurrency (ICBC)*, pp. 1–9, 2021.

[18] O. Ersoy, S. Roos, and Z. Erkin, "How to profit from payments channels," in *Proc. Int. Conf. on Financial Cryptography and Data Security*, pp. 284–303, 2020.

[19] J. Poon and T. Dryja, "The bitcoin lightning network: Scalable off-chain instant payments," *White Paper*, pp. 1–59, 2016.

[20] R. Network, "What is the raiden network," *URL: https://raiden.network/101.html*, 2019. Last Accessed: June 16, 2023.

[21] A. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999.

[22] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[23] R. Pickhardt, "Lightning-network-autopilot." Available: https://github.com/renepickhardt/lightning-network-autopilot, 2019.

[24] P. Erdos and A. Rényi, "On the evolution of random graphs," *Publ. Math. Inst. Hung. Acad. Sci*, vol. 5, no. 1, pp. 17–60, 1960.