Balance Transfers and Bailouts in Credit Networks using Blockchains[*]

Lalitha Muthu Subramanian, Roopa Vishwanathan, Kartick Kolachala
Department of Computer Science, New Mexico State University
{lalitha,roopav,kart1712}@nmsu.edu

Abstract—**In this paper, we propose a technique for rebalancing link weights in decentralized credit networks. Credit networks are peer-to-peer trust-based networks that enable fast and inexpensive cross-currency transactions compared to traditional bank wire transfers. Although researchers have studied security of transactions and privacy of users of such networks, and have invested significant efforts into designing efficient routing algorithms for credit networks, comparatively little work has been done in the area of *replenishing* credit links of users in the network. This is achieved by a process called *rebalancing* that enables a poorly funded user to create incoming as well as outgoing credit links. We propose a system where a user with zero or no link weights can create incoming links with existing, trusted users in the network, in a procedure we call *balance transfer*, followed by creating outgoing links to existing or new users that would like to join the network, a process we call *bailout*. Both these processes together constitute our proposed rebalancing mechanism.**

I. Introduction

Blockchain and cryptocurrencies such as Bitcoin [1] have disrupted the banking industry, enabled new business models, and helped in designing new, efficient financial infrastructure. Blockchains have enabled the growth of IOU (I Owe You) credit networks in recent years. A credit network is a decentralized peer-to-peer lending network, where users lend out financial credit based on social trust, where the payment is routed through multiple intermediate users. Credit networks are usually modeled as a directed graph with vertices (users) and weighted edges/links(ammount of credit). Payments are routed along (and in the direction of) credit links, and once a payment is made from a sender to receiver, the link weights are decremented along the transaction path. There has been growing interest in finding solutions for efficient routing, and enabling private and secure transactions in credit networks [2, 3, 4] but not much work has been done in the area of *rebalancing* link weights of a user that has run out of credit. Rebalancing in credit networks is a significant problem to study, since, if the credit on a given link gets exhausted, i.e., if the weight on a link connecting two nodes drops to zero, no transactions can be done on any path containing those nodes until the link is refunded, a process which involves expensive on-chain transactions. In this paper, we study the problem of rebalancing links in an efficient way, while making minimal use of the blockchain, with the goal of avoiding mining and blockchain write fees. We propose a two-step rebalancing process wherein a node whose link weights are low or close to zero, can create fresh incoming links in a process called *balance transfer*, and then create fresh outgoing links in a process called as *bailout*.

Our Contributions:
1) We propose a two-step approach for rebalancing consisting of *balance transfer* and *bailout*. In the balance transfer step, a poorly connected node, called as *requestor*, will establish incoming links with other nodes in the network by advertising a lower interest rate. In the bailout step, a well-known party such as a bank will infuse capital into the requestor node, by helping it connect to, and establish outgoing links with several other nodes in the network. After the requestor node establishes outgoing connections with other nodes, the bank will leave the network, possibly after collecting a fee from the requestor. At the end of this process, the requestor node will have several incoming and outgoing links, which will enable it to help facilitate several transactions, thus increasing the overall throughput and robustness of the credit network.
2) Since the performance of our balance transfer step is highly dependent on being able to find routes efficiently, we compare and analyze two different routing algorithms for doing balance transfers: Chord [5] and VOUTE [6].

II. Related Work

Credit networks: Fulgor and Rayo [7] were the first to setup a peer-to-peer payment channel network that provides provable security and privacy properties, with Rayo being the first payment network that enforces non-blocking transactions. Both of them assume their users to be honest They both, establish a path between sender and receiver, assuming all users in the path to be honest, and users have at least partial knowledge about network topology. In our system, the entire network topology is not known to the users and we do not assume all the users in the path to be honest. Also, we do not propose any payment operations, instead focus only on rebalancing credit links.

SilentWhispers [3] presents a decentralized credit network (DCN) architecture which consists of subsets of paths between the sender and receiver calculated via several trusted entities

[*]Research supported by NSF award #1800088. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Authorized licensed use limited to: Cornell University Library. Downloaded on September 26,2020 at 20:37:29 UTC from IEEE Xplore. Restrictions apply.

called *landmarks*. PathShuffle [8] presents a path mixing protocol for Ripple network providing complete anonymity. We use the landmarks in our system to assist the requestor node establish outgoing links without placing enormous trust on landmarks. Roos *et al.* [9] used graph embedding for efficient routing with concurrent transactions overcoming some inefficiencies in [3]. Malavolta *et al.* [10] recently proposed a novel linkable ring signature scheme for refund transactions natively in Monero [11]. Panwar *et al.* [4] proposed a DCN system where users can perform path-based transactions that preserves sender, receiver and value privacy but involves a significant number of blockchain writes in the course of a normal, successful transaction. Our system also involves blockchain writes, but in our system, a single blockchain write is done only after the completion and execution of the entire rebalancing protocol.

III. System Design

Credit networks are usually dense networks, e.g., Ripple [12], with several incoming and outgoing links from the nodes. If a node has depleted credit links, then, intuitively, one way for it to rebalance its links would be to extend credit to, and borrow from new users. This could be problematic for several reasons. Ultimately, whether to lend or borrow from a user, we believe, should be a matter of choice. With this design goal in mind, we introduce the two steps of balance transfer and bailout.

Balance transfer: In this step, any user can disconnect from an existing lender and transfer credit links to a new lender node offering a lower rate of interest. We would need to perform efficient routing between lenders and borrowers in the balance transfer step.

Bailout: In the bailout step, a trusted, highly connected party such as a bank, or a credit union *temporarily* lends credit to a node, say, D, so that D can establish outgoing connections. We refer to this trusted party as a *landmark*, or LM. The high-level idea is that LM will use the fact that is is highly connected, and temporarily connect D with several other nodes in the network with whom LM has a direct connection. D will then request each of these nodes if they would like to lend credit to D, thus establishing outgoing links form D to them. Note that any or all nodes can decline D's request, at which point LM will connect D with a fresh set of nodes.

IV. Adversarial Model

In our system, we assume that any adversary can corrupt a single or a set of users in the network. The corrupted user(s) can be either the requester, who raises a rebalancing request, the nodes that respond to the request, or any intermediate node. During the bailout phase, we need a trusted landmark, LM, who is temporarily assists the requestor node. We assume the adversary cannot corrupt the LM node. Each user i has her own signing and verification key pair (sk_i, vk_i) and an encryption, decryption key pair (pk_i, dk_i). Once any user is corrupted, their corresponding signing and verification keys are compromised, the adversary can misreport the credit link between a user and her neighbor, not respond to any request, respond selectively to requests and relay fraudulent balance transfer requests to its neighbors. We assume that an adversary cannot corrupt all users in the network, and thus may know partial network topology, but does not know the entire network.

V. Routing in Balance Transfers

In this section we discuss two different routing protocols we use for balance transfers in credit networks: prefix embedding [13] and Chord [5].

A. Routing using Prefix Embedding

The first part of the balance transfer algorithm is the *Find Route* phase in which the responder node finds a route from itself to the requester. We make use of prefix embedding [13] and VOUTE [6] to establish a route after which rebalancing occurs. Prefix based embeddings are a part of greedy embeddings [13] [14]. They are, in general, created by embedding a spanning tree into a suitable metric space. An ID is assigned to the root node, and the tree consists of several child nodes where each child computes the ID based on the ID of its parent node. Prefix embedding is an adaptation of PIE embedding for unweighted graphs. The idea is that, every node is assigned an ID using a custom metric space such that the node ID is equivalent to the hop distance or the depth of a spanning tree. A child's id is essentially the ID of the parent, and an additional coordinate equal to the index of the child.

B. Routing using Chord

Chord [5] is a routing algorithm built using distributed hash tables for peer-to-peer networks, without any centralized monitoring authority. Chord uses a $\langle key, value\rangle$ pair to map to a specific node across the system. The keys are assigned to nodes using Consistent Hashing [15] across the network. Chord assigns a $\langle key, value\rangle$ pair to each of the nodes in the system, where the key is the identifier using SHA-1 [16] and maps the keys to the nodes that are responsible for them. A peer identifier is chosen by hashing the data key. The length of the identifier is usually large to ensure the probability of keys hashing to the same identifier is negligible. Key k is assigned to the first peer whose identifier is equal to or follows k in identifier place and the first peer, clockwise from k is called the successor peer of k, represented by $successor(k)$. When a peer n joins or leaves the system, the keys that previously belonged to n, is reassigned to $n's$ successor. This enables maintaining consistent hashing in the system. For helping users join and leave the Chord network, we run a *stabilization* protocol at regular intervals that updates the *finger table* stored at each node. The finger table (FT) stored at each node is a table containing the IDs of its successors.

The full version of the paper, which includes details of our routing protocols, our constructions, algorithms, and experimental results is available at [17].

Authorized licensed use limited to: Cornell University Library. Downloaded on September 26,2020 at 20:37:29 UTC from IEEE Xplore. Restrictions apply.

References

[1] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” 2008.

[2] P. Prihodko, S. Zhigulin, M. Sahno, A. Ostrovskiy, and O. Osuntokun, “Flare: An approach to routing in lightning network,” *White Paper*, 2016.

[3] G. Malavolta, P. Moreno-Sanchez, A. Kate, and M. Maffei, “Silentwhispers: Enforcing security and privacy in decentralized credit networks.” in *NDSS*, 2017.

[4] G. Panwar, S. Misra, and R. Vishwanathan, “Blanc: Blockchain-based anonymous and decentralized credit networks,” in *In Ninth ACM Conference on Data and Application Security and Privacy (CODASPY’19)*, 2019.

[5] I. Stoica, R. T. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan, “Chord: a scalable peer-to-peer lookup protocol for internet applications,” *IEEE/ACM Trans. Netw.*, vol. 11, no. 1, pp. 17–32, 2003. [Online]. Available: https://doi.org/10.1109/TNET.2002.808407

[6] S. Roos, M. Beck, and T. Strufe, “Voute-virtual overlays using tree embeddings,” 2016.

[7] G. Malavolta, P. Moreno-Sanchez, A. Kate, M. Maffei, and S. Ravi, “Concurrency and privacy with payment-channel networks,” in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2017, pp. 455–471.

[8] P. Moreno-Sanchez, T. Ruffing, and A. Kate, “Pathshuffle: Credit mixing and anonymous payments for ripple,” *Proceedings on Privacy Enhancing Technologies*, vol. 2017, no. 3, pp. 110–129, 2017.

[9] S. Roos, P. Moreno-Sanchez, A. Kate, and I. Goldberg, “Settling payments fast and private: Efficient decentralized routing for path-based transactions,” *arXiv preprint arXiv:1709.05748*, 2017.

[10] P. Moreno-Sanchez, RandomRun, D. V. Le, S. Noether, B. Goodell, and A. Kate, “DLSAG: non-interactive refund transactions for interoperable payment channels in monero,” *IACR Cryptology ePrint Archive*, vol. 2019, p. 595, 2019. [Online]. Available: https://eprint.iacr.org/2019/595

[11] S. Noether and A. Mackenzie, “Monero research lab.“,” *Ring Confidential Transactions.” Ledger*, vol. 1, pp. 1–18, 2016.

[12] F. Armknecht, G. O. Karame, A. Mandal, F. Youssef, and E. Zenner, “Ripple: Overview and outlook,” in *International Conference on Trust and Trustworthy Computing*. Springer, 2015, pp. 163–180.

[13] S. Roos, L. Wang, T. Strufe, and J. Kangasharju, “Enhancing compact routing in CCN with prefix embedding and topology-aware hashing,” in *Proceedings of the 9th ACM Workshop on Mobility in the Evolving Internet Architecture, MobiArch 2014, Maui, HI, USA, September 11, 2014*, R. L. Aguiar and K. Guo, Eds. ACM, 2014, pp. 49–54. [Online]. Available: https://doi.org/10.1145/2645892.2645900

[14] T. Leighton and A. Moitra, “Some results on greedy embeddings in metric spaces,” *Discrete & Computational Geometry*, vol. 44, no. 3, pp. 686–705, 2010. [Online]. Available: https://doi.org/10.1007/s00454-009-9227-6

[15] A. Dury, “Peer-to-peer computing in distributed hash table models using a consistent hashing extension for access-intensive keys,” in *Agents and Peer-to-Peer Computing, Third International Workshop, AP2PC, 2004, New York, NY, USA, July 19, 2004, Revised and Invited Papers*, ser. Lecture Notes in Computer Science, G. Moro, S. Bergamaschi, and K. Aberer, Eds., vol. 3601. Springer, 2004, pp. 185–192. [Online]. Available: https://doi.org/10.1007/11574781_17

[16] F. P. NIST, “180-1: Secure hash standard,” 1995.

[17] “Balance transfers and bailouts in credit networks using blockchains.” [Online]. Available: https://arxiv.org/submit/3077031/view

Authorized licensed use limited to: Cornell University Library. Downloaded on September 26,2020 at 20:37:29 UTC from IEEE Xplore. Restrictions apply.