Concentrated Stopping Set Design for Coded Merkle Tree: Improving Security Against Data Availability Attacks in Blockchain Systems

Debarnab Mitra, Lev Tauz and Lara Dolecek
Department of Electrical and Computer Engineering, University of California, Los Angeles, USA
email: debarnabucla@ucla.edu, levtauz@ucla.edu, dolecek@ee.ucla.edu

***Abstract*—In certain blockchain systems, light nodes are clients that download only a small portion of the block. Light nodes are vulnerable to a *data availability* (DA) attack where a malicious node makes the light nodes accept an invalid block by hiding the invalid portion of the block from the nodes in the system. A technique based on LDPC codes called Coded Merkle Tree (CMT), proposed by Yu *et al.*, enables light nodes to detect a DA attack by randomly requesting/sampling portions of the block from the malicious node. However, light nodes fail to detect a DA attack with high probability if a malicious node hides a small stopping set of the LDPC code. To improve the probability of detection, in this work, we demonstrate a specialized LDPC code design that focuses on concentrating stopping sets to a small group of variable nodes rather than only eliminating stopping sets. Our design demonstrates a higher probability of detecting DA attacks compared to prior work thus improving the security of the system.**

I. Introduction

A blockchain is an immutable ledger of transaction blocks stored in a distributed manner among its users (nodes). *Full nodes* in a blockchain system store the entire ledger in their memory and operate on it to validate transactions. Modern non-volatile memory technologies such as persistent memories can improve blockchain performance due to their low latency (allowing faster validation) and high reliability, e.g. [1].

Due to the excessive storage and compute requirements of running full nodes [2], blockchain systems also run *Light nodes:* they only store the header corresponding to each block. Light nodes cannot verify transactions and rely on full nodes for fraud notifications. Blockchain systems with a majority of malicious full nodes are susceptible to *data availability (DA)* attacks [2], [3]. In this attack, a malicious full node generates a block with invalid transactions and hides the invalid portion making the honest full nodes unable to validate the block and notify the light nodes of the malicious behaviour. Light nodes can detect a DA attack by randomly sampling a few chunks of the block from the malicious node (who generated the block) and reject the block if all the requested chunks are not returned. To improve the probability of detection, the block is encoded using an erasure code [2]. However, coding allows the malicious node to carry out an *incorrect-coding attack* [2], [3] in which case the full nodes send an *incorrect-coding proof* to the light nodes to reject the block. The size of this proof is proportional to the sparsity of the parity check equations and to keep the storage size at the light nodes small, authors in [3] proposed to use a Low-Density Parity-Check (LDPC) code to encode the block. The coded block is decoded by a peeling decoder which fails if the malicious node hides coded symbols corresponding to a stopping set [3] of the LDPC code. Since the malicious node can hide the smallest stopping set, the best code design strategy to reduce the probability of failure using random sampling is to construct LDPC codes that have large minimum stopping set size. Constructing such LDPC codes is a known hard problem [4]. In this work, we show that the probability of failure can be reduced by an efficient co-design of specialized LDPC codes and the light node sampling strategy. The main contribution of the work is an LDPC code construction called the *entropy-constrained* PEG (EC-PEG) algorithm that concentrates stopping sets to a small set of variable nodes (VNs) and a greedy sampling strategy that samples a large number of stopping sets. We show that our new co-design results in a lower probability of failure compared to prior techniques. All details can be found in [5] and [6].

The authors acknowledge the Guru Krupa Foundation and NSF-BSF grant no. 2008728 to conduct this research work.

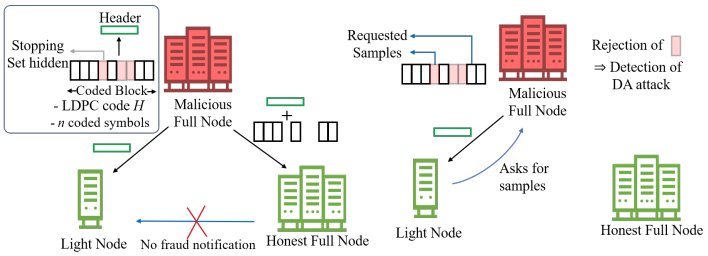

Fig. 1: System Model: A malicious node generates a coded block with n coded symbols using an LDPC code with a binary parity check matrix H (having VNs $\mathcal{V} = \{v_1, \ldots, v_n\}$). A DA attack occurs when a malicious node generates an invalid block and hides coded symbols corresponding to a stopping set of H such that honest full nodes are unable to decode the block fully using a peeling decoder [3]. Light nodes detect a DA attack by sampling coded symbols.

II. System Model

Our system model is shown in Fig. 1. The goal for the light nodes is to detect a DA attack and reject the unavailable block. To accomplish this goal, light nodes sample coded symbols from the malicious node and accept the block if all the requested symbols are returned. A light node fails to detect a DA attack if the samples requested are not hidden. Let $p_f(s, \omega)$ be the probability of failure for some sampling strategy with s samples when the malicious node hides a stopping set of H with ω VNs. In this work, we provide construction of H and a coupled sampling strategy to reduce $p_f(s, \omega)$ compared to prior work. In [3], random sampling with replacement is employed and $p_f(s, \omega) = (1 - \frac{\omega_{\min}}{n})^s$, where $\omega_{\min}$ is the size of the smallest stopping set of H. In this work, we limit ourselves to a malicious node that is unaware of the sampling strategy used by the light nodes and for a given size of the stopping set ω, it hides a randomly chosen stopping set of size ω. We provide co-design techniques to handle stronger adversaries in [6].

III. Design Idea: Stopping Set Concentration

A VN v *touches* a cycle (stopping set) of the LDPC code iff v is part of the cycle (stopping set). Let $\mathcal{G}$ be the Tanner Graph representation of H. A cycle of length g is a g-cycle. Let $ss^\omega = (ss_1^\omega, \ldots, ss_n^\omega)$ and $\zeta^g = (\zeta_1^g, \ldots, \zeta_n^g)$ denote the VN-to-stopping-set of weight ω and VN-to-g-cycle distributions, respectively, where ss_i^ω (ζ_i^g) is the fraction of stopping sets

of weight ω (g-cycles) touched by v_i. We informally say that distribution ss^{ω} (ζ^g) is concentrated if a small set of VNs have a high corresponding stopping set (g-cycle) fraction.

Lemma 1. *Let $\mathcal{SS}_\omega$ be the set of all weight ω stopping sets of H. If the malicious node picks a stopping set randomly from $\mathcal{SS}_\omega$ and hides it, then probability of failure $p_f(s,\omega)$ satisfies $p_f(s,\omega) \geq 1 - \max_{\mathcal{S}\subseteq\mathcal{V},|\mathcal{S}|=s}\tau(\mathcal{S},\omega)$, where $\tau(\mathcal{S},\omega)$ is the fraction of stopping sets of weight ω touched by the set of VNs $\mathcal{S}$. The lower bound in the above equation is achieved when light nodes sample the set $\mathcal{S}_\omega^{opt} = \text{argmax}_{\mathcal{S}\subseteq\mathcal{V},|\mathcal{S}|=s}\tau(\mathcal{S},\omega)$.*

Lemma 1 suggests that the lowest probability of failure is $1-\tau(\mathcal{S}_\omega^{opt},\omega)$. Now, $\tau(\mathcal{S}_\omega^{opt},\omega)$ is large if a majority of stopping sets of weight ω are touched by a small subset of VNs, which is achieved if the distributions ss^{ω} are concentrated. Thus, designing LDPC codes with concentrated ss^{ω} increases the value of $\tau(\mathcal{S}_\omega^{opt},\omega)$ and reduces the probability of failure.

Instead of finding $\mathcal{S}_\omega^{opt}$, in this work, we greedily find a set S_{greedy} of s VNs that touch the largest number of cycles starting from $g_{\min}$(the girth of the code)-cycles. Light nodes sample VNs in S_{greedy} and we have $p_f(s,\omega) = 1-\tau(\mathcal{S}_{greedy},\omega)$.

IV. Entropy-constrained PEG algorithm

In this section, we provide a method to concentrate stopping sets. We do so by concentrating the cycle distributions ζ^g since stopping sets are made up of cycles [7]. Our method utilizes entropy as a measure of concentration. Since distributions that are concentrated have low entropy, we construct LDPC codes using the PEG algorithm [8] by making check node (CN) selections that minimize the entropy of the cycle distributions. The full EC-PEG algorithm is provided below. It differs from the original PEG algorithm [8] in steps 7-15. In the algorithm, we maintain g'-cycle counts $\Lambda_i^{(g')}$ for each VN v_i and $g' < g_c$. In steps 7-13, for each candidate CN c, we find the updated cycle counts (steps 8-10), normalize them (step 11) and calculate the entropy of the averaged normalized cycle counts (step 13). Then, we select the CN with the minimum Entropy[] (step 14).

Algorithm 1 EC-PEG Algorithm

1: **Inputs:** n, m , d_v, g_c, **Outputs:** $\widetilde{\mathcal{G}}$, $g_{\min}$, **Initialize** $\widetilde{\mathcal{G}}$ to n VNs, m CNs and no edges, $T = |\{4,6,\ldots,g_c-2\}|$
2: **Initialize** $\Lambda_i^{(g')} = 0$, for all $g' < g_c$ and $1 \leq i \leq n$
3: **for** $j = 1$ to n **do for** $k = 1$ to d_v **do** $[\mathcal{K}, g]$ = PEG$(\widetilde{\mathcal{G}}, v_j)$
4: **if** $g \geq g_c$ **then**
5: c^{sel} = "min degree CN selection" procedure
6: **else** ▷ *(g-cycles, $g < g_c$ are created)*
7: **for** each c in $\mathcal{K}$ **do**
8: $\lambda_i^{(g',c)} = \Lambda_i^{(g')}$, $g' < g_c$, $1 \leq i \leq n$
9: $\mathcal{L}$ = new g-cycles in $\widetilde{\mathcal{G}}$ due to edge(c, v_j)
10: **for** all v in $\widetilde{\mathcal{G}}$ **do** $\lambda_v^{(g,c)} = \lambda_v^{(g,c)} + |\{\mathcal{O} \in \mathcal{L} \mid v \in \mathcal{O}\}|$
11: $\alpha_i^{(g')} = \frac{\lambda_i^{(g',c)}}{\sum_{i=1}^{n}\lambda_i^{(g',c)}}$, $1 \leq i \leq n$, $g' < g_c$
12: $\alpha_{g_c} = (\sum_{g'<g_c}\frac{\alpha_1^{(g')}}{T}, \sum_{g'<g_c}\frac{\alpha_2^{(g')}}{T}, \ldots, \sum_{g'<g_c}\frac{\alpha_n^{(g')}}{T})$
13: Entropy$[c] = \mathcal{H}(\alpha_{g_c})$
14: $c^{sel} = \text{argmin}_{c\in\mathcal{K}}\text{Entropy}[c]$, $\Lambda_i^g = \lambda_i^{(g,c^{sel})}$ $\forall i$,
15: $\widetilde{\mathcal{G}} = \widetilde{\mathcal{G}} \cup \text{edge}\{c^{sel}, v_j\}$

V. Simulation Results

We show results for $n = 128$, rate $= 0.5$, VN degree $= 4$, $g_c = 10$ in Fig. 2 and 3. Fig. 2 shows the stopping set distributions ss^{ω} for EC-PEG and PEG LDPC codes where the VN indices are arranged in decreasing order of stopping set fractions ss_i^{ω}. Clearly, the EC-PEG algorithm results in concentrated stopping sets since the VNs towards the left on the x-axis have high stopping set fraction. In Fig. 3, the probability of failure $p_f(s,\omega)$ is shown. From the figure, we see that the combination of the greedy sampling and concentrated LDPC codes provided by the EC-PEG algorithm results in a lower probability of failure compared to using random ensembles and random sampling (black curve), as used in [3], as well as the PEG algorithms with random sampling (green and red curves).

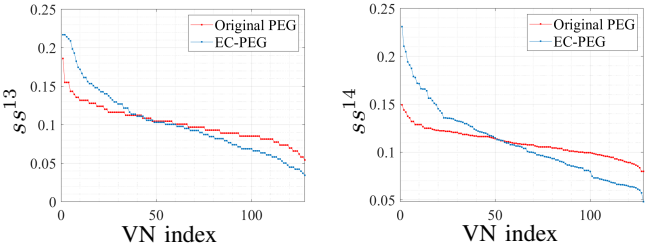

Fig. 2: Stopping set distributions ss^{13} (left panel) and ss^{14} (right panel);

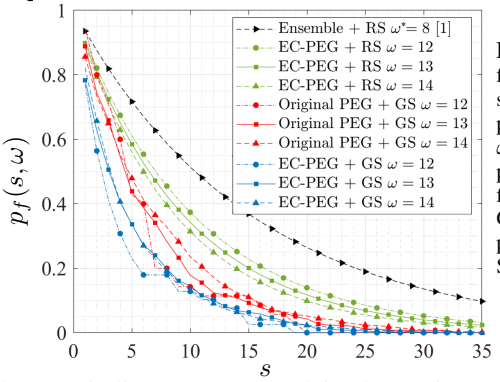

Fig. 3: $p_f(s,\omega)$ for various coding schemes and sampling strategies. ω^*: minimum stopping set size (w.h.p.) for ensemble of [3]. GS: Greedy Sampling RS: Random Sampling

VI. Conclusion, Extensions and Future Work

In this work, we proposed a novel design of LDPC codes for application in blockchains with DA attacks and showed a significant improvement in detecting DA attacks compared to previous schemes. While we have considered an adversary that is unaware of our new co-design, in our full paper [6] we also consider a stronger adversary that does have this knowledge and devise a novel extension of EC-PEG LDPC codes to handle this case. As a future extension, we are currently investigating other code families, such as Polar Codes, for combating DA attacks.

References

[1] Z. E. Lee, *et al.*, "Performance Evaluation of Big Data Processing at the Edge for IoT-Blockchain Applications," *IEEE Global Commun. Conf. (GLOBECOM)*, 2019.

[2] M. Al-Bassam, A. Sonnino, V. Buterin, "Fraud and data availability proofs: Maximising light client security and scaling blockchains with dishonest majorities," *arXiv preprint arXiv:1809.09044*, Sept. 2018.

[3] M. Yu, *et al.*, "Coded merkle tree: Solving data availability attacks in blockchains," *Int. Conf. on Financial Cryptography and Data Secur., Springer, Cham*, Feb. 2020.

[4] X. Jiao, *et al.*, "Eliminating small stopping sets in irregular low-density parity-check codes," *IEEE Commun. Lett.*, vol. 13, no. 6, Jun. 2009.

[5] D. Mitra, L. Tauz, and L. Dolecek, "Concentrated Stopping Set Design for Coded Merkle Tree: Improving Security against Data Availability Attacks in Blockchain Systems," *2020 IEEE Inf. Theory Workshop (ITW)*, Apr. 2021.

[6] D. Mitra, L. Tauz, and L. Dolecek, "Overcoming Data Availability Attacks in Blockchain Systems: LDPC Code Design for Coded Merkle Tree," *arXiv preprint arXiv:2108.13332*, Aug. 2021.

[7] T. Tian, *et al.*, "Construction of irregular LDPC codes with low error floors," *IEEE Int. Conf. on Commun.*, vol. 5, May 2003.

[8] X.Y. Hu, E. Eleftheriou and D.M. Arnold, "Regular and irregular progressive edge-growth tanner graphs," *IEEE Trans. on Inf. Theory*, vol. 51, no. 1, Jan. 2005.