AntiBenford Subgraphs: Unsupervised Anomaly Detection in Financial Networks

Tianyi Chen
ctony@bu.edu
Boston University
Boston, MA, USA

Charalampos Tsourakakis
ctsourak@bu.edu
Boston University
Boston, MA, USA

ABSTRACT
Benford’s law describes the distribution of the first digit of numbers appearing in a wide variety of numerical data, including tax records, and election outcomes, and has been used to raise “red flags” about potential anomalies in the data such as tax evasion. In this work, we ask the following novel question:

Given a large transaction or financial graph, how do we find a set of nodes that perform many transactions among each other that also deviate significantly from Benford’s law?

We propose the AntiBenford subgraph framework that is founded on well-established statistical principles. Furthermore, we design an efficient algorithm that finds AntiBenford subgraphs in near-linear time on real data. We evaluate our framework on both real and synthetic data against a variety of competitors. We show empirically that our proposed framework enables the detection of anomalous subgraphs in cryptocurrency transaction networks that go undetected by state-of-the-art graph-based anomaly detection methods. Our empirical findings show that our AntiBenford framework is able to mine anomalous subgraphs, and provide novel insights into financial transaction data.

The code and the datasets are available at https://github.com/tsourakakis-lab/antibenford-subgraphs.

CCS CONCEPTS
• Mathematics of computing → Graph algorithms; • Computing methodologies → Anomaly detection.

KEYWORDS
anomaly detection, financial networks, dense subgraph

1 INTRODUCTION
Anomaly detection is a broad term describing methods that detect patterns that deviate from normal behavior [17]. The development of an anomaly detection method is application-dependent, and requires an understanding of what constitutes “normal”. For example, consider the following question of interest to revenue services: how do we detect tax evasion and fraud? What information can we use in order to raise “red flags” about certain accounts that will undergo additional control? Benford’s law, aka Newcomb-Benford law [10, 53], states that the first digit of numbers appearing in natural datasets, such as addresses, areas, stock quotations, town populations, is not uniformly distributed. Specifically, the fraction of numbers starting with digit d is not the uniform 1/9 (d = 1, . . . , 9), but instead follows the monotone decreasing function of d, i.e., log_{10}(1 + 1/d). This distribution is shown in Figure 1(a). Benford’s law has been used numerous times successfully in practice to raise red flags that eventually detect fraud, see for instance [16, 21, 58, 61, 63, 70].

In this work we are interested in developing novel unsupervised methods for detecting anomalous subgraphs in financial and transaction networks. Despite the large toolbox of unsupervised anomaly detection methods [31, 71], the task of designing novel, well-performing unsupervised methods remains a great challenge. Furthermore, the case of transaction networks is special for (at least) two major reasons. First, transaction networks are knowledge graphs, with a lot of important information available for the nodes and edges; e.g., we know that if a node has a PO box only, or resides in what is called high risk country, the conditional probability of being part in an illicit scheme is higher than some node that has a standard bank account in a low risk country. These special characteristics can play a key role in detecting anomalies, but have not been taken account largely by the research community. One of the key reasons behind this fact is the scarcity of real-world banking data, as well as the natural hesitation of financial institutions to publish more on their anomaly detection systems, despite the existence of dedicated venues (e.g., Journal of Money Laundering Control). Despite the scarcity of publicly available bank transaction datasets due to privacy reasons, the wide-spread use of cryptocurrencies has enabled the collection of interesting financial datasets. For instance, in a Bitcoin transaction network, nodes correspond to transactions, and edges to Bitcoins flow between transactions [79].

Finding subgraphs that correspond to illicit behavior such as malware, money laundering, ransomware, and Ponzi schemes is a challenging task. Towards this goal, we propose a novel graph based anomaly detection method that leverages two components: (i) Benford’s law, and (ii) dense subgraph discovery, a major area of graph mining. The latter has been used widely in a variety of tasks ranging from finance to biology, and also anomaly detection,
In this work we propose the \textbf{AntiBenford} subgraph anomaly detection problem, stated informally as follows:

\textbf{Problem 1. (Informal) Anti-Benford subgraphs.} Given a weighted network $G(V, A, w)$, \( w : A \to \mathbb{R}^+ \) where each arc \( e = (u, v) \in A \) corresponds to a transaction, and \( w(u, v) \) is the amount of money \( u \) sends to \( v \), find a subset of nodes \( S \subseteq V \) such that (i) it induces on average many edges, and (ii) whose weights’ first digit empirical histogram deviates from Benford’s law significantly, in a statistical sense.

Intuitively, Antibenford subgraphs are dense subgraphs whose edges have an empirical distribution significantly different from Benford’s law. The edge density constraint (i) biases the output towards a set of nodes that in some way are likely to know each other, or in the context of cryptocurrency financial networks, they may even correspond to wallets held by a single person or a small group of people who coordinate their actions. The second constraint (ii) biases the output to include edges whose weights “violate” Benford’s law in the following sense; if the edges’ first digits are independent and identically distributed (iid) random variables (RVs) according to Benford’s law, then their distribution is a multinomial. There exists a variety of \textit{goodness-of-fit} tests for testing such hypotheses, e.g., [34, 66]. As we show experimentally in Section 4, combining constraints (i), and (ii) results in finding interesting, and statistically significant anomalous subgraphs. It is worth mentioning that we empirically verify that Ethereum blockchain transactions follow closely Benford’s law, see Section 4.3 for details.

Our contributions in this work include the following.

- \textbf{AntiBenford subgraphs.} We propose a novel unsupervised framework for detecting anomalies in financial networks. This generic task is known to be a challenging real-world problem of great importance.

- \textbf{Mathematical and algorithmic analysis.} We propose a specific formulation for problem 1, and design an algorithm that is algorithmically founded on dense subgraph discovery. Furthermore, we provide mathematical insights into the behavior of our objective using a probabilistic analysis.

- \textbf{Experimental validation.} Our method is able to uncover anomalous subgraphs in various settings. For example, Figure 1(b) shows an AntiBenford subgraph found in the Ethereum network spanning a period of a month (Jan. 2018). The subgraph has an apparent tripartite structure, where the money flows from one layer to the next. Furthermore, the distribution of the first digit among this set of transactions deviates significantly from Benford’s law, see Figure 4. The structure resembles a smurf-like\(^1\) structure with respect to the layering, a subgraph that is known to appear in money laundering schemes [44, 67]. These findings are representative of what our proposed method can achieve in complex, networked data. Despite the remarkable success of prior graph-based anomaly detection methods, our findings verify that our framework provides information that is otherwise inaccessible.

\textbf{Reproducibility.} Our code is publicly available at https://github.com/tsourakakis-lab/antibenford-subgraphs. The paper is organized in the usual way, i.e., related work (Section 2), proposed method (Section 3), experiments (Section 4), and conclusions (Section 5).

\textbf{Audience.} We believe that our contributions are especially valuable to practitioners in the banking sector, where the detection of fraud and money laundering are of paramount importance. Designing unsupervised anomaly detection systems is an extremely challenging task, and our proposed framework is based on well-founded principles, and enjoys great performance on real-world transaction graphs.

\(^1\)In banking, smurfing refers to splitting of a large financial transaction into multiple smaller transactions.
2 RELATED WORK

Graph Based Anomaly Detection is an intensively active area of graph mining [55], that attracts interest from a diversity of industrial and scientific applications. Such applications include security applications [12, 19], anti-money laundering [45], spammers [3, 25, 35], and fraud detection [22]. We indicatively report some works in this area; the interested reader should confer the extensive survey on graph-based anomaly detection [4]. k-cores have also been frequently used to detect anomalies in large-scale networks [28, 65]. Many frameworks tend to use both graph topology and node/edge attributes to identify anomalies. Noble and Cook [55] studied anomaly detection on graphs with categorical features by searching for graph sub-structures that occur infrequently. The evaluation is based on the Minimum Description Length Principle (MDL). Davis et al. [20] suggested an extension that is able to treat both structural data and numeric attributes. OddBall is a network. Graph anomaly detection has been applied to financial near-cliques, see also [37] for further applications.

Finally, EigenSpokes [60] uses the correlations between eigenvectors to detect near-bipartite cliques and near-cliques, see also [37] for further applications.

Dense subgraph discovery is a major topic in graph mining [29]. Detecting dense subgraphs has a wide range of applications, ranging from anomaly detection [25, 35, 36, 45], to bioinformatics [26, 62]. A wide variety of mathematical formulations has been proposed in the literature. The archetypical dense subgraph is a clique. However, the maximum clique problem is not only NP-hard, but also strongly inapproximable, see [33]. Many other formulations that relax the clique requirement, and instead require that a large fraction of all pairs of nodes are connected are also NP-hard, and typically hard to approximation. For instance, finding an optimal quasi-clique [39, 74] is also NP-hard [73], but scalable solutions that work well on real-world datasets have been developed in recent years [42, 51].

In sheer contrast to the quasi-clique formulations with respect to the computational complexity, stands the densest subgraph problem (DSP) that is solvable in polynomial time on graphs with non-negative weights [27, 30, 40]. Given a graph \(G(V, E, w)\), \(w \colon E \to \mathbb{R}^+\) the densest subgraph problem (DSP) aims to find a subset of nodes \(S \subseteq V\) that maximizes the degree density \(\rho(S) = \frac{\sum_{(i, j) \in E(S)} w(i, j)}{|S|^2}\). Here, \(w(S)\) denotes the total edge weight induced by \(S\), i.e., \(w(S) = \sum_{(i, j) \in E(S)} w(i, j)\). When the graph is unweighted, then the numerator \(w(S)\) simply equals the number of induced edges \(e(S) = |E(S)|\), and the degree density is simply half the average degree of the subgraph \(G(S)\) induced by \(S\). When there exist negative weights, the DSP becomes NP-hard [76]. We emphasize that the degree density objective is better suited for undirected, non-bipartite graphs; we focus on this objective in this paper. More suitable objectives for other types of graphs have also been proposed in the literature, see [18, 38, 40, 47].

The DSP can be solved exactly using (parametric) maximum flows [27, 30]. Since maximum flow computations are computationally expensive [56] and theoretical advances in approximate max flow computations have not been yet translated to scalable software implementations [48, 50, 57], practitioners usually favor a \(\frac{1}{2}\)-approximation algorithm that uses linear-space and run in linear time for unweighted graphs. Specifically, Charikar [18] showed that the greedy algorithm proposed by Asashiro et al. [7] is a \(\frac{1}{2}\)-approximation algorithm. We refer to this algorithm as the Greedy Peeling algorithm. The algorithm removes in each iteration, the node with the smallest degree. This process creates a nested sequence of sets of nodes \(V = S_n \supseteq S_{n-1} \supseteq S_{n-2} \supseteq \ldots \supseteq S_1 \supseteq \emptyset\). The algorithm outputs the graph \(G(S_j)\) that maximizes the degree density \(\rho(S_j)\) among \(j = 1, \ldots, n\). Recently, Boob et al. proposed a new algorithm [14] that combines near-optimal solutions with time efficiency. We emphasize that the DSP has no restrictions on the size of the output. When one imposes cardinality constraints on the size of \(|S|\) the problem becomes NP-hard. For instance, solving the DSP with the requirement that the output set should have exactly \(k\) nodes is known as the densest-k-subgraph problem and the state-of-the-art approximation algorithm is due to Bhaskara et al. [11], and provides a \(O(n^{1/4 + \epsilon})\) approximation in \(O(n^{1/\epsilon})\) time. Other versions where \(|S| \geq k, |S| \leq k\) have also been considered in the literature see [6]. The DSP has been studied under various other computational models in addition to the RAM model, including dynamic setting [13, 23, 64], the streaming model [9, 13, 24, 49], and the MapReduce model [8, 9]. In recent years, notable extensions of
the DSP have been proposed. Tsourakakis proposed the k-clique DSP [72, 75] that can be solved fast on massive graphs [51, 68], and Tatti and Gionis [69] introduced the locally-dense graph decomposition.

**Tests of fit.** Parametric hypothesis testing is an extensive area of statistics [78]. We are interested in tests of fit for the multinomial distribution, as Benford’s law can be thought of as a 9-sided loaded die that determines the first digit of a transaction weight. More generally, we wish to test the hypothesis

\[ H_0 : F(x) = F_0(x), \]

where \( F_0(x) \) is some particular cumulative distribution function. This problem is known in general as a goodness-of-fit problem \([34, 66]\). In the case where we have k classes, and we assume \( F_0 \) corresponds to the multinomial distribution \((p_1, \ldots, p_k)\), where \( n_i \) is the number of observations from class \( i \) for \( i = 1, \ldots, k \), and the total number of observations is \( n = \sum_{i=1}^{k} n_i \) then the likelihood function is given by

\[ L(n_1, \ldots, n_k \mid p_1, \ldots, p_k) = \prod_{i=1}^{k} \frac{n_i!}{n!}, \]

It is well known that the likelihood of alternative hypothesis \( H_1 : F(x) = F_1(x) \) is maximized when we consider \( F_1 \) to be the multinomial distribution with parameters given by the ML estimators of its parameters, namely \(( \hat{p}_1 = \frac{n_1}{n}, \ldots, \hat{p}_k = \frac{n_k}{n} \). The likelihood ratio (LR) statistic for testing \( H_0 \) against \( H_1 \) is equal to the ratio of the two likelihood functions, and is equal to \( \ell = n^n \prod_{i=1}^{k} \left( \frac{p_i}{n} \right)^{n_i} \).

Hypothesis \( H_0 \) is rejected when the LR statistic is small enough. It is known that as \( n \to +\infty \), the function \(-2\log \ell \) is asymptotically distributed in the \( \chi^2 \) form with \( k-1 \) degrees of freedom. Karl Pearson proposed the \( \chi^2 \) statistic \( \chi^2 = \sum_{i=1}^{k} \frac{(n_i - np_i)^2}{np_i} \). Asymptotically the LR test, and the \( \chi^2 \) statistic have the same distribution, but it is known that for small values of \( n \) they can yield different results.

**Theoretical Preliminaries.** In Section 3 we use the following Chernoff bound \([52]\).

**Theorem 1** (Chernoff bound). Consider a set of mutually independent binary random variables \( X_1, \ldots, X_n \). Let \( X = \sum_{i=1}^{n} X_i \) be the sum of these random variables. Then, for \( 0 < \epsilon < 1 \) we have

\[ \Pr \left[ |X - E[X]| \geq \epsilon E[X] \right] \leq 2e^{-\epsilon^2 E[X]/3}. \]

### 3 ANTI-BENFORD SUBGRAPHS

Given a weighted transaction network \( G(V = [n], E, w) \), we view the set of its edges \( E \) as iid samples from Benford’s distribution, and we use the standard \( \chi^2 \)-statistic to test whether it follows such distribution \([34]\). We refer to this hypothesis as \( H_0 \).

The \( \chi^2 \)-statistic of a subgraph \( G[S] \) induced by \( S \subseteq V \) is defined as

\[ \chi^2(S) = \frac{\sum_{d=1}^{k} (X_{S,d} - \mathbb{E}(X_{S,d}))^2}{\mathbb{E}(X_{S,d})}, \]

where \( X_{S,d} \) is the number of edges in subgraph induced by \( S \) whose weight’s first digit is \( d = 1, \ldots, k \). The \( \chi^2 \)-statistic of the whole graph \( G \) is defined as \( \chi^2(V) \).

It is a well-known fact this statistic follows the \( \chi^2 \) distribution with \( k-1 = 8 \) degrees of freedom. We define the average \( \chi^2 \)-statistic of the graph as \( \chi^2(V) = \frac{\chi^2(V)}{|V|} \), and more generally the average \( \chi^2 \)-statistic of any induced subgraph \( G[S] \) as \( \chi^2(S) = \frac{\chi^2(S)}{|S|} \), where \( \chi^2(S) \) is the \( \chi^2 \) statistic of the weights induced by \( S \).

**Definition 1** (AntiBenford subgraph). An AntiBenford subgraph is a subset of nodes \( S \subseteq V \) such that \( \chi^2(S) \gg \frac{|S|}{|V|} \).

The above definition is founded on our probabilistic analysis, see Corollary 1. Assuming the null hypothesis \( H_0 \) is true, the average \( \chi^2 \)-score of any subgraph is a diminishing fraction of its order:

\[ \mathbb{E} \left[ \chi^2(S) \right] = \frac{\mathbb{E} \left[ \chi^2(S) \right]}{|S|} = \frac{1}{|S|} \sum_{d=1}^{k} \mathbb{E} \left[ X_{S,d} \right] \]

\[ = \frac{1}{|S|} \sum_{d=1}^{9} (p_d(1-p_d)) e(S) = \frac{9}{|S|} \sum_{d=1}^{9} p_d e(S) = \frac{8}{|S|}. \]

However, even if the null hypothesis \( H_0 \) is true, we do expect to observe subgraphs that deviate from the mean. Roughly speaking, our mathematical analysis shows that the key quantity to look at is the degree density, i.e., half of the average degree. Notice that a claim of the form ”there does not exist a large subgraph whose average \( \chi^2 \)-score is much larger than its degree density”, is far from obvious; the standard probabilistic approach of bounding the failure probability of a given subgraph by \( \frac{1}{n^2} \) for any constant \( C \), and a union bound over all possible subgraphs does not yield any meaningful results, since we have an exponential number of bad events.

Our key theoretical result shows that under hypothesis \( H_0 \) there exist no AntiBenford subgraphs with average degree \( \Omega(\log n) \).

**Theorem 2.** Let \( 0 < \epsilon < 1 \) be an accuracy parameter, and let \( \delta \) def min\( d \), \( p_d = p_0 = 0.048 \) be the lowest digit probability in Benford’s law. Suppose that the null hypothesis \( H_0 \) is true, and all edge weights are iid samples from Benford’s distribution. Then, with high probability for all subgraphs \( S \subseteq V \) with average degree at least \( C \log n \) where \( C = \frac{36}{\delta^2} \) is a constant depending on \( \delta, \epsilon \), the number of edges \( X_{S,d} \) that start with digit \( d \) in \( G[S] \) is strongly concentrated around its expectation, for all \( d = 1, \ldots, 9 \).

**Proof.** Consider a subgraph induced by \( S \subseteq V \), and a digit \( d \). Let \( e(S), X_{S,d} \) be the number of induced edges, and the number of edges whose weight starts with digit \( d \) respectively. Notice that \( \mathbb{E} \left[ X_{S,d} \right] = p_d e(S) \). By the Chernoff bound (see Theorem 1) we obtain that for any \( 0 < \epsilon < 1 \) the following inequality holds

\[ \Pr \left[ |X_{S,d} - p_d e(S)| \geq \epsilon p_d e(S) \right] \leq 2e^{-\epsilon^2 p_d e(S)/2} \leq 2e^{-6|S|\log n} \leq n^{-6|S|}. \]
We used the fact that \( \mathbb{E} \left[ X_{S,d} \right] = \frac{p_d e(S)}{12} \log n \), since the average degree satisfies \( \frac{\Delta(G)}{|S|} \geq \frac{\Delta}{\log n} \). Now, consider the following double union bound over all digits, and subgraphs with average degree \( \frac{\Delta(G)}{|S|} = \Omega(\log n) \) as described above:

\[
\Pr \left[ \exists d \in \{1, \ldots, 9\}, S \subseteq V : X_{S,d} \not\in (1 \pm \epsilon) \mathbb{E} \left[ X_{S,d} \right] \right] \leq \sum_{d=1}^{9} \left( \frac{n}{k} \right)^{-6k} \leq \sum_{k=2}^{n} \left( \frac{en}{k} \right)^{k} n^{-6k} = o(1)
\]

Therefore, we conclude that with high probability \( 1 - o(1) \), the number of edges with first digit \( d \) in subgraphs with large enough average degree is strongly concentrated around the expectation for all possible digits \( d \). □

This implies that we expect for all sufficiently large subgraphs to have a strong concentration of \( X_{S,d} \) around the true expectation. This fact in turn implies that the average \( \chi^2 \) score of a set \( S \) satisfying the conditions of Theorem 2 is within a factor of \( e^2/2 \) of its average degree. To see why, notice that with high probability the following holds:

\[
\psi(S) = \frac{1}{|S|} \sum_{d=1}^{9} \left( X_{S,d} - \mathbb{E}(X_{S,d}) \right)^2 \leq \frac{1}{|S|} \sum_{d=1}^{9} e^2 p_d e(S) = \frac{e^2}{2} \frac{\mathbb{E}(S)}{|S|}.
\]

Since \( e < 1 \), we state this informally as the next corollary:

**Corollary 1.** Under the null hypothesis \( H_0 \), we expect to observe subgraphs \( S \subseteq V, |S| = \Omega(\log n) \) with \( \psi(S) \approx \frac{\epsilon(S)}{|S|} \). Furthermore, \( \max_{X_{S,d}} \psi(S) \leq \rho^* \), where \( \rho^* \) is the density of the densest subgraph in \( G \).

**Algorithm.** Our approach is outlined in pseudocode 1. For each node \( u \), we measure how the transactions it is involved into, deviate from Benford’s distribution by computing the \( \chi^2 \) score. We define the anomaly score \( s(u) \) of node \( u \) as

\[
s(u) = \sum_{d=1}^{9} \left( \frac{X_{S,d} - \mathbb{E}(X_{S,d})}{\mathbb{E}(X_{S,d})} \right)^2.
\]

where \( X_{S,d}^u \) is the number of transactions incident to \( u \) whose first digit is \( d \). Once we have computed all scores \( \{s(u)\}_{u \in V} \) we reweight each edge by a function of the scores. Specifically, we find that the following function \( f(u,v) = \sqrt{s(u) \cdot s(v)} \) works well in practice. Then we solve the densest subgraph problem, namely we maximize the following objective \( \max_{S \subseteq V} \frac{\sum u \in E(S) f(u,v)}{|S|} \). Our choice of reweighting biases the densest anomalous subgraph to contain nodes that are all involved in transactions that deviate from Benford’s; consider a node \( u \) with high degree towards an anomalous set of nodes, but assume that overall the transactions that involve \( u \) agree with Benford’s, so \( s(u) \) is low. Then, our reweighting scheme will avoid including such a node in an optimal solution, in contrast to other possible AntiBenford formalizations of the form \( \max_{S \subseteq V} \frac{\epsilon(S) + \sum u \in S s(u)}{|S|} \). Such formulations are known to be solvable in polynomial time for non-negative scores, see [30, 35]. Notice that one can use a linear program, a max-flow approach, or a greedy approximation algorithm for finding the DSP (line 6) in the algorithm. Let \( \alpha \) and \( T_{\text{dsp}}(n, m) = \Omega(n + m) \) be the approximation guarantee (\( \alpha = 1 \) for exact) and the run time of the algorithm DSP used in line 6. Then, we obtain the following straight-forward proposition:

**Proposition 1.** We can find an \( \alpha \)-approximate AntiBenford subgraph in \( O(n + m + T_{\text{dsp}}(n, m)) = O(\Omega(n + m)) \) time in the standard RAM model of computation.

**Algorithm 1: AntiBenford subgraphs**

**Input:** \( G = (V, E, \omega) \)

**Output:** Set of nodes \( S \subseteq V \)

1. for \( u \in V \) do
   2. \( E_u = \{ w(u,v) : (u,v) \in E \} \);
   3. \( s(v) = \sum_{d=1}^{9} \left( \frac{X_{S,d} - \mathbb{E}(X_{S,d})}{\mathbb{E}(X_{S,d})} \right)^2 \);
   4. for \( (u,v) \in E \) do
      5. \( W_{uv} = \sqrt{s(u) \cdot s(v)} ;\)
   6. \( S = \text{DSP}(G' = (V, E, W)) ;\)
   7. if not \( \psi(S) \gg \psi(V) \) then
      8. \( */ No statistically significant anomalous subgraph found */ \)
      9. return \( S \)

In our experiments, we favor for the scalable greedy algorithm. Finally, in practice we may want to extract more than one AntiBenford subgraph. In order to do so, we run Algorithm 1 repeatedly, and each time we remove the previous output set of vertices from the graph, thus finding node-disjoint AntiBenford subgraphs.

4 EXPERIMENTS

4.1 Experimental Setup

**Datasets and competitors.** We use Ethereum token transfer networks together with synthetic data to evaluate our proposed framework. As competitors we use EigenSpoke [60], HoloScope [46], and FlowScope [45]. The real-world datasets are summarized in Table 1.

- **Ethereum blockchain.** We use the Ethereum token transfer dataset supported and shared by Google BigQuery [1]. The original data contains for each token transaction its source and sink addresses, value, timestamp and block information. We present our anomaly detection results from two snapshots, spanning the periods of January 2018 and January 2019. We filter out all transactions with values less than 1 unit. We also construct five cumulative Ethereum token transfer networks spanning from one week to the whole year of 2019 to analyze the running time of our method, see the second part of Table 1.

**Machine Specs and code.** The experiments are performed on a single machine, with Intel i7-10850H CPU @ 2.70GHz and 32GB of main memory. Our graph analysis code is written in Python3, and we use the C++ implementation of DSP algorithms from [14].
for efficiency. We use the spartan2\(^2\) library for all competitors. The code and datasets are available at https://github.com/tsourakakis-lab/antibenford-subgraphs.

### 4.2 Synthetic data

The goal of our experiments on synthetic data is two-fold. First, we want to show that popular competitors cannot detect the planted anomalies, and secondly show the limitations of AntiBenford subgraphs.

Towards this purpose, we generate clustered networks, with planted anomalies with respect to the first digit distribution in order to compare how well various methods can detect them. Since the original HoloScope implementation assumes that the input has a bipartite structure, we generate a version of the stochastic block-model as follows. We have two types of nodes (users vs. objects), and nine clusters. Each cluster is a bipartite clique, and clusters are interconnected with edge probability \( p = 0.1 \), namely for each pair of user \( u \), and object \( i \) from two separate clusters we generate an edge independently with probability \( p \). Then we sample first digits on the edges (since we only care about the first digit, this suffices) according to Benford’s distribution for all the edges except for the induced edges for three clusters. Those three bicliques receive digits 1, 2, 3 with probability 1 respectively. We set the size of each normal cluster to 80 nodes, and we range the size of the anomalous bicliques in \([20, 50, 80, 100]\).

Figure 2 shows the results of the three methods. Eigenspoke and Holoscope cannot detect the planted anomalous subgraphs. For example, Holoscope outputs the whole graph. It is worth emphasizing that the increase in the F1 of Holoscope is due to the fact that the size of the anomalous clusters increases, and not due to any changes in the performance of the algorithm. Our method achieves perfect precision and recall when the size of the anomalous clusters is greater than 20, by finding the top-3 outputs of Algorithm 1, we find the top-5 AntiBenford subgraphs and show their \( \chi^2 \) and \( \psi \) scores in Table 2, together with the statistics for the whole network that closely follows Benford’s law. We observe a sheer contrast of the value of \( \psi \) between the global network and the suspicious subgraphs we found. In Table 2 we also mark highly suspicious subgraphs whose \( \psi \) values are significantly higher than their densities \( |E|/|S| \), according to our rigorous analysis, see Corollary 1. Figure 3 shows the empirical distribution of the quantity \( s(u)/\text{deg}(u) \), i.e. the \( \chi^2 \) score of node \( u \) averaged by its degree. The histogram of this quantity is skewed, and follows (roughly) a power law with slope \( \alpha = 2.6 \) on the whole network of ETH-Jan-18 [5]. We obtain this fitting using the Python package powerlaw [5]. The middle and right figures in Figure 3 show the histograms for two top AntiBenford subgraphs that deviate from the histogram of the global graph on the left.

Figure 4 describes the first digit distribution of the subgraph we found from the Ethereum token transfer network in January 2018. Transaction values of this subgraph strongly violate Benford’s law as half of them start with 5. Figure 1(b) visualizes the topology which is close to a tripartite directed graph. According to previous studies

---

\(^2\)https://github.com/BGT-M/spartan2

---

| Dataset | Metrics | 1\(^{st}\) | 2\(^{nd}\) | 3\(^{rd}\) | 4\(^{th}\) | 5\(^{th}\) | global |
|---------|---------|---------|---------|---------|---------|---------|---------|
| ETH-Jan-18 | \( \chi^2 \) | 3429 | 2587 | 379 | 2.59e5 | 698 | 1.6e5 |
| | \( \psi \) | 14.4 | 7.39 | 11.14 | 35.75 | 3.96 | 0.09 |
| | \(|E|/|S|\) | 18.8 | 13.09 | 3.82 | 1.96 | 1.99 | 2.43 |
| ETH-Jan-19 | \( \chi^2 \) | 131818 | 3963 | 1205 | 10677 | 9372 | 2.92e5 |
| | \( \psi \) | 1588 | 10.88 | 9.06 | 9.58 | 9.49 | 4.26e-3 |
| | \(|E|/|S|\) | 214 | 15.5 | 4.77 | 14.15 | 3.96 | 2.79 |
Figure 3: Left: Histogram of the quantity $s(u) / \deg(u)$ of nodes in (a) ETH-Jan-18, which roughly fits power law with $\alpha = 2.6$. Middle, right: Histogram of $s(u) / \deg(u)$ of nodes in two AntiBenford subgraphs.

Figure 4: First digit distribution of Benford’s law, transaction values of ETH-Jan-18 and ETH-Jan-19 networks, and the suspicious subgraphs found.

Table 3: Suspicious transactions started from one sender, through multiple middle accounts, and ended at a receiver at the same date, graph structure visualized in Figure 1(a). We only show the last eight characters of address hash. In and out date correspond to the day Ethereums are transferred into and out of middle accounts respectively.

| Sender address | Receiver address | Amount (ETH) per middle account | In Date (yy/mm/dd) | Out Date (yy/mm/dd) |
|----------------|------------------|---------------------------------|--------------------|---------------------|
| beb58d36       | 9b06578f         | 9667                            | 18/01/18           | 18/02/06            |
| 07c15792       | 59c42026         | 2160                            | 17/09/15           | 17/09/24            |
| 9a974c3b       | 59c42026         | 33565                           | 17/10/13           | 17/10/20            |
| c0h31d0        | 89705733         | 17200                           | 18/02/03           | 18/02/17            |

[41, 45, 67], this follows a common money laundering network pattern where money flows quickly (i.e., in a matter of a couple of weeks) from few sources to few targets via middle nodes. We can even visually recognize the sources, middle nodes, and final recipients. Over 140,000 Ethereums are transferred from top to bottom in one month in this subgraph. Interestingly we verify that all middle nodes send out all the money they receive, which is a property known to hold in money laundering as the zero out property, see also [45, 67]. A sample of such transactions are shown in Table 3. We observe that a certain amount of money is sent to several middle accounts on the same day by one sender, and is altogether sent out to one receiver after at most a couple of weeks, and sometimes even earlier than that.

On the Ethereum Jan. ’19 network, our method outputs a clearly anomalous subgraph. The distribution of its digits significantly deviates from Benford’s law, which, as we mentioned before, fits well the empirical global first digit distribution. Most of the transactions start with digits 3 and 4. Table 2 summarizes the statistics of the top-5 anomalous subgraphs found using Algorithm 1, and Figure 4 shows the resulting digit distributions, contrasted with the global digit distributions that follow closely Benford’s law. Figure 5 shows that the subgraph can be partitioned into three groups of nodes. There is a central node, depicted as red, a set of middle nodes marked as orange, and a group of accounts marked blue that form a more complicated network, but still exhibit a DAG-like structure, with top-down edge flow. Each big arrow between groups represents complete edge connections between two groups in a certain direction. The central node in one month sends out more than 1.1 million Ethereum Tokens to other accounts inside this subgraph, and receives more than 1.2 Ethereum Tokens from them with some external incomes through the blue bottom accounts.

Flowscope - a framework tailored for anti-money laundering - is unable to output any of these subgraphs. Even worse, its outputs do not appear suspicious at all, and consist of a path of length 2, with large weights on the edges. Furthermore, we observe that suspicious structures in real-world financial networks do not always exhibit a perfect $k$-partite structure, and thus are not easily detected by Flowscope.

**Scalability.** We apply AntiBenford and other competitors on a set of five Ethereum networks that span transactions from one week to one year. We report the running time in Figure 6. Due to the linear complexity of greedy peeling, Algorithm 1 can efficiently find anomalies even in the largest graph we have.

**5 CONCLUSION**

In this work we contribute a novel, statistically founded formulation for finding anomalies in transaction networks; it is known that unsupervised anomaly detection in financial networks is a challenging, and important problem world-wide. We show that
well-founded definition of AntiBenford [22] W. Eberle and L. Holder. Discovering structural anomalies in graph-based data. [26] E. Fratkin, B. T. Naughton, D. L. Brutlag, and S. Batzoglou. Motifcut: regulatory motifs finding with maximum density subgraphs. [25] D. Eswaran, C. Faloutsos, S. Guha, and N. Mishra. Spotlight: Detecting anomalies benchmarking in graphs and other methods on cumulative snapshots of Ethereum transaction networks. Figure 6: Running time comparison of AntiBenford subgraphs and other methods on cumulative snapshots of Ethereum transaction networks.

Figure 5: Suspicious subgraph found from Ethereum token transfer network of Jan 2019. Center node circulates more than 0.7 million USD from Oct 2018 to Mar 2019.

ANTI-BENFORD subgraphs exhibit characteristics that are known in the literature to appear in illicit transactions, such as smurf-like structures showing that criminals are likely to operate abnormally both in terms of creating special subgraph structures, and splitting the money carefully. We provide an extension of Benford’s law on networks, and we show that the analysis of such a law presents technical challenges due to the fact that there exists an exponential number of “bad events”. Based on our analysis we provide a well-founded definition of AntiBenford subgraphs. We show empirically that the resulting AntiBenford subgraphs are able to find anomalies in Ethereum networks that are otherwise unobtainable from state-of-the-art graph-based anomaly detection methods. An interesting direction is the design of algorithms that allow for overlapping anomalous subgraphs, and extending the experimental setup to include other measures of statistical deviation.

REFERENCES
[1] Complete live historical ethereum blockchain data (bigquery). https://www.kaggle.com/bigquery/ethereum-blockchain.
[2] S. Agarwal, S. Dutta, and A. Bhattacharya. Chisel: Graph similarity search using chi-squared statistics in large probabilistic graphs. Proc. VLDB Endow., 13(10):1654–1668, June 2020.
[3] L. Akoglu, M. McGlohon, and C. Faloutsos. Oddball: Spotting anomalies in weighted graphs. In Pacific-Asia conference on knowledge discovery and data mining, pages 410–421. Springer, 2010.
[4] L. Akoglu, H. Tong, and D. Koutra. Graph based anomaly detection and description: a survey. Data mining and knowledge discovery, 29(3):626–688, 2015.
[5] J. Aslott, E. Bullmore, and D. Flenz. powerlaw: a python package for analysis of heavy-tailed distributions. PloS one, 9(1):e85777, 2014.
[6] R. Andersen and K. Chellapilla. Finding dense subgraphs with size bounds. In International Workshop on Algorithms and Models for the Web-Graph, pages 25–37. Springer, 2009.
[7] Y. Asahiro, K. Iwama, H. Tamaki, and T. Tokuyama. Greedily finding a dense subgraph. Journal of Algorithms, 34(2):203–221, 2000.
[8] B. Bahmani, A. Goel, and K. Munagala. Efficient primal-dual graph algorithms for maxreduce. In International Workshop on Algorithms and Models for the Web-Graph, pages 59–78. Springer, 2014.
[9] B. Bahmani, R. Kumar, and S. Vassilvitskii. Densest subgraph in streaming and mapreduce. Proceedings of the VLDB Endowment, 5(3):445–465, 2012.
[10] F. Benford. The law of anomalous numbers. Proceedings of the American philosophical society, pages 551–562, 1938.
[11] A. Bhaskara, M. Charikar, E. Chlamtac, U. Feige, and A. Vijayaraghavan. Detecting high log-densities: an o(n1/4) approximation for densest k-subgraph. In Proceedings of the forty-second ACM symposium on Theory of computing, pages 201–210, 2010.
[12] S. Bhatia, B. Hoot, M. Yoon, K. Shin, and C. Faloutsos. MidaS: microcluster-based detector of anomalies in edge streams. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, pages 3242–3249, 2020.
[13] S. Bhattacharya, M. Henzinger, D. Nanongkai, and C. Tsourakakis. Space-and time-efficient algorithm for maintaining dense subgraphs on one-pass dynamic streams. In Proceedings of the forty-seventh annual ACM symposium on Theory of computing, pages 173–182. ACM, 2015.
[14] D. Boob, Y. Gao, R. Peng, S. Sawlani, C. Tsourakakis, D. Wang, and J. Wang. Flowless: extracting densest subgraphs without flow computations. In Proceedings of The Web Conference 2020, pages 573–583, 2020.
[15] F. Braun, O. Caem, E. N. Smirnov, S. Kelk, and B. Lechiot. Improving card fraud detection through suspicious pattern discovery. In S. Benerhat, K. Tabia, and M. Ali, editors, Advances in Artificial Intelligence: From Theory to Practice, pages 181–190, Cham, 2017. Springer International Publishing.
[16] A. Cersioli, L. Barabesi, A. Ceresa, M. Menegatti, and D. Perrotta. Newcomb–benford law and the detection of frauds in international trade. Proceedings of the National Academy of Sciences, 116(1):106–115, 2019.
[17] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. ACM computing surveys (CSUR), 41(3):1–58, 2009.
[18] M. Charikar. Greedy approximation algorithms for finding dense components in a graph. In International Workshop on Approximation Algorithms for Combinatorial Optimization, pages 84–95. Springer, 2000.
[19] Y. Chen, Y. Nadji, A. Kountouras, F. Monrose, R. Perdisci, M. Antonakakis, and N. Vasiloglou. Practical attacks against graph-based clustering. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, pages 1125–1142, 2017.
[20] M. Davis, W. Liu, P. Miller, and G. Redpath. Detecting anomalies in graphs with numeric labels. In Proceedings of the 20th ACM International Conference on Information and Knowledge Management. CIKM ’11, page 1197–1202, New York, NY, USA, 2011. Association for Computing Machinery.
[21] P. D. Drake and M. J. Nigrini. Computer assisted analytical procedures using benford’s law. Journal of Accounting Education, 18(2):127–146, 2000.
[22] W. Eberle and L. Holder. Discovering structural anomalies in graph-based data. In Seventh IEEE international conference on data mining workshops (ICDMW 2007), pages 393–398. IEEE, 2007.
[23] A. Epasto, S. Lattanzi, and M. Sozio. Efficient densest subgraph computation in evolving graphs. In Proceedings of the 24th International Conference on World Wide Web, pages 300–310. International World Wide Web Conferences Steering Committee, 2015.
[24] H. Esfandiari, M. Hajiaghayi, and D. P. Woodruff. Applications of uniform sampling: Densest subgraph and beyond. arXiv preprint arXiv:1506.04505, 2015.
[25] D. Eswaran, C. Faloutsos, S. Guha, and N. Mishra. Spotlight: Detecting anomalies in streaming graphs. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pages 1378–1386, 2018.
[26] E. Fratkin, B. T. Naughton, D. L. Brultag, and S. Batzoglou. Motifcut: regulatory motifs finding with maximum density subgraphs. Bioinformatics, 22(14):e150–e157, 2006.
