Exchange Pattern Mining in the Bitcoin Transaction Directed Hypergraph

Stephen Ranshous[1], Cliff A Joslyn[2], Sean Kreyling[2], Kathleen Nowak[2], Nagiza F Samatova[1,3], Curtis L West[2], and Samuel Winters[2]

[1] North Carolina State University, Raleigh, NC
smransho@ncsu.edu, samatova@csc.ncsu.edu
[2] Pacific Northwest National Laboratory, Seattle and Richland, WA
{cliff.joslyn, sean.kreyling, katy.nowak, curtis.west, samuel.winters}@pnnl.gov
[3] Oak Ridge National Laboratory, Oak Ridge, TN

Abstract. Bitcoin exchanges operate between digital and fiat currency networks, thus providing an opportunity to connect real-world identities to pseudonymous addresses, an important task for anti-money laundering efforts. We seek to characterize, understand, and identify patterns centered around exchanges in the context of a directed hypergraph model for Bitcoin transactions. We introduce the idea of motifs in directed hypergraphs, considering a particular 2-motif as a potential laundering pattern. We identify distinct statistical properties of exchange addresses related to the acquisition and spending of bitcoin. We then leverage this to build classification models to learn a set of discriminating features, and are able to predict if an address is owned by an exchange with $> 80\%$ accuracy using purely structural features of the graph. Applying this classifier to the 2-motif patterns reveals a preponderance of inter-exchange activity, while not necessarily significant laundering patterns.

Keywords: bitcoin, exchanges, transaction graph, directed hypergraph, motif, classification

1 Introduction

Bitcoin's decentralization makes it difficult to regulate and investigate by law enforcement. This represents a vulnerability in government anti-money laundering (AML) efforts [4]. Conventional AML efforts focus on the Know-Your-Customer (KYC) process, in which banks and other financial services must verify the identity of their customers, monitor transactions, and report suspicious behavior to government entities. As such, government AML and KYC efforts utilize perfect knowledge of identity but incomplete knowledge of financial transactions, which remains in the control of the banks [12, 13]. In contrast, law enforcement generally has no knowledge about bitcoin user identities to use in detecting anomalous behavior, but access to the blockchain grants complete knowledge of transactions. This motivates the desire to detect money laundering through techniques that do not rely on identity information, such as transaction or user patterns. In particular, patterns centered around exchanges are important, as they provide arguably the most important link between Bitcoin and fiat currency networks. Moreover, exchanges are navigating evolving legal precedent to

be AML compliant [1, 2]. In 2015, FinCEN fined Ripple Labs in the first act of civil enforcement against a Virtual Currency Exchange for failing to implement a proper AML program [17].

We model Bitcoin transactions as a directed hypergraph (dirhypergraph), which naturally represents the multi-way relation between addresses and transactions (Section 2). This is distinguished from previous analyses [15, 16] which use graph models with strictly binary edges, whether at the address or tx level. We define motifs in dirhypergraphs as small sub-graph patterns, and introduce a small 2-motif involving exchanges which we call "short thick bands" (STB) as a *potential* laundering pattern. We identify several patterns in the behavior of exchange-owned addresses that differ from non-exchange addresses (Section 3). For example, where regular addresses are likely to be sinks [16], simply accumulating bitcoin, exchange addresses typically keep a near-zero balance. Third, we explore the possibility of applying machine learning techniques to classify latent attributes of addresses (Section 4). In particular, we focus on whether or not it is possible to predict whether a given address is owned by an exchange or not, and the role of both labeled and putative exchanges in STB patterns. Finally, we seek to understand whether the pattern of exchange use in STBs reveals potential laundering activity, but conclude that what can be identified is a preponderance of inter-exchange activity.

2 Bitcoin Transaction Motifs in a Directed Hypergraph

Bitcoin transactions have a natural graphical structure, one form of which is shown on the left of Fig. 1. Vertices are transactions $E_0, \ldots E_3$, while arcs model inputs and outputs labeled by the Bitcoin address $a_1, \ldots, a_6$, weighted by the quantity. We note some common activities such as change making and aggregation. Coinbase transactions are indicated by the vertex SRC in blue, while unspent transaction outputs (UTXOs) are combined into a single sink vertex.

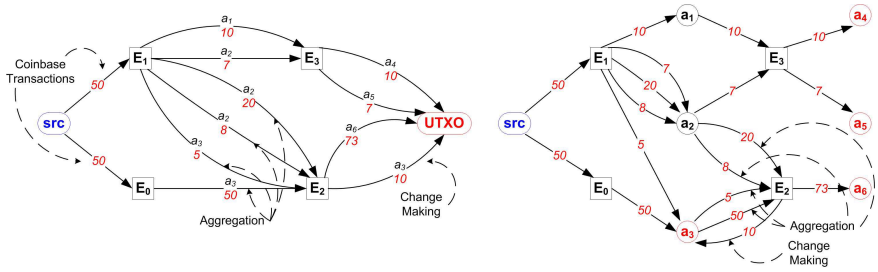


Fig. 1. (Left) Bitcoin transactions as a labeled multigraph. (Right) Bitcoin transactions as a bipartite multigraph.

But in analytical tasks, such as detecting money laundering, it is perhaps more important to focus on addresses (arc labels a_i) than the transactions E_j. And while discouraged in Bitcoin, reuse of addresses is legal and somewhat common. To treat addresses as "first class objects" we create new vertices for each unique address, producing the bipartite graph structure on the right of Fig. 1. Square vertices are transactions E_j, while the circles are addresses a_i (addresses

currently with an UTXO are red). Input arcs from addresses to transactions are now distinguishable from output arcs from transactions to addresses, and address reuse can create looping structures, such as shown in change-making back into a_3 as an output of E_2.

We consolidate by combining quantities on multi-arcs, producing the directed hypergraph [3, 5] in Fig. 2. Dirhypergraphs are characterized as directed bipartite graphs with two kinds of vertices, with connections only between vertices of different types, but possibly multiple inputs from and outputs to each. For us, transaction (square) vertices act as directed hyperarcs. Where an arc in a graph connects a single tail vertex to a single head vertex, hyperarcs connect multiple input (tail) addresses to multiple output (head) addresses (round vertices).

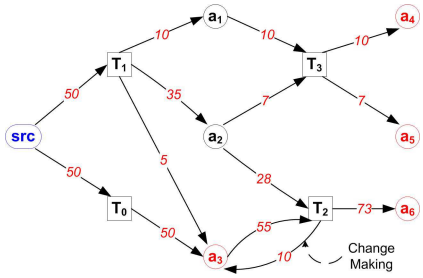


Fig. 2. Bitcoin transactions as a directed hypergraph.

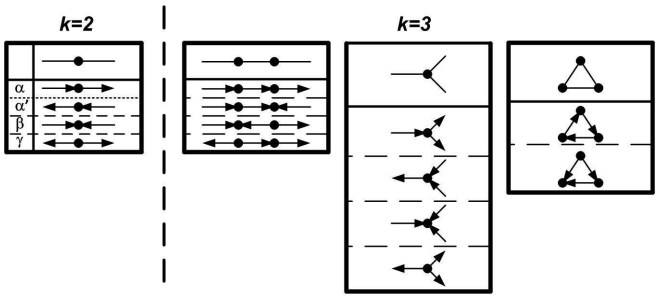


Fig. 3. Graph motifs: (Far Left) The single undirected graph 2-motif (above) with its three directed motifs α, β, and γ (below), including the two isomorphic α, α' patterns. (Right) The three undirected 3-motifs and their directed versions.

Hypergraphs and directed hypergraphs are well known in math and computer science, and can provide significant advantages over regular graphical structures when data are complex, with multiple inputs and outputs as in our case due to address reuse. Identifying subgraphs indicating potential laundering suggests the potential significance of hypergraph motifs. In network analysis, motifs are small subgraphs which are represented with statistical significance [11]. Our research group appears to be the first to consider dirhypergraph motifs, which we generalize directly from graph motifs. Fig. 3 shows all the undirected and directed motifs for two and three edges/arcs. A k-motif is one possible way that k connected (intersecting) edges can be structured, with a range of possible numbers of vertices sitting as tails and heads of the k edges.

The simplest dirhypergraph Bitcoin pattern is the 2-motif,[4] illustrated on the left side of Fig. 4 as a dirhypergraph pattern. As in Fig. 2, the two transactions E_1, E_2 are square vertices, and addresses circles (circle color will be addressed below). Note that any particular address can sit on either the tails (inputs) or heads (outputs) of any transaction, and indeed, more than one transaction.

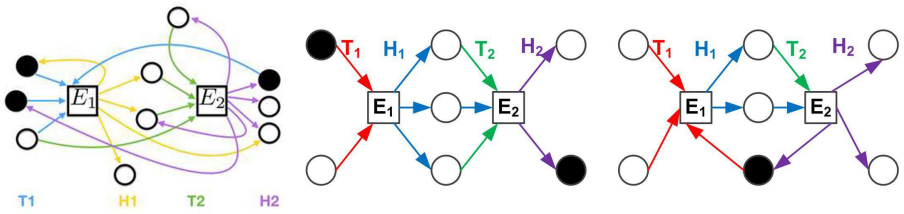


Fig. 4. A 2-motif in a dirhypergraph: Two transactions E_1, E_2 with sets of tail and head vertices T_1, H_1, T_2, H_2 respectively. (Left) Generic. (Center) A linear STB. (Right) A circular STB.

Formalizing 2-motifs, assume a non-empty finite set of vertices $A = \{a_i\}$ and two hyperarcs $E_j = \langle T_j, H_j \rangle, j = 1, 2$, with tails and heads $T_j, H_j \subseteq A$ (nonempty). The set $M = \{E_1, E_2\}$ is a 2-motif if there is at least one pair of tails and heads in each hyperarc which intersect, that is, if $\bigcup_{X,Y \in \{H,T\}} X_1 \cap Y_2 \neq \emptyset$. The potential intersections are detailed in Table 1. For example, an address sits on a γ pattern if it is in both T_1 (the blue vertices in Fig. 4) and T_2 (the green), that is, in the tails of (inputs to) both transactions (e.g. the lowest left address in Fig. 4). The analogous graph pattern γ is shown in Fig. 3, with the two edges pointed outward. Note that in comparison with Fig. 3, here we also allow self-loops identified in the L_1, L_2 patterns.

Pattern	Condition	Description
α	$H_1 \cap T_2 \neq \emptyset$	Forward 2-chain
α'	$T_1 \cap H_2 \neq \emptyset$	Reverse 2-chain
β	$H_1 \cap H_2 \neq \emptyset$	Inward 2-star
γ	$T_1 \cap T_2 \neq \emptyset$	Outward 2-star
L_1	$T_1 \cap H_1 \neq \emptyset$	Self-loop on E_1
L_2	$T_2 \cap H_2 \neq \emptyset$	Self-loop on E_2

Table 1. Participation of addresses in a 2-motif.

The left of Fig. 5 is an abstraction of a 2-motif, where each circular vertex represents one of the patterns in Table 1, standing in for the entire *set* of addresses playing that role.[5] The right is the same abstraction, but now with the counts of the number of addresses in each role for transactions between Jan 12 2015 and April 21 2015.[6]

[4] Terminologically, we can call these hypermotifs or hypergraph motifs, but for simplicity here we will just call them motifs.

[5] We include addresses in *exactly* one intersection, ignoring addresses in *only* a tail or head of one of the transactions, and also addresses in *more than two* intersections.

[6] Note the similarities of the counts for α and α', on the one hand, and L_1 and L_2, on the other, due to isomorphism with respect to the ordering of E_1 and E_2.

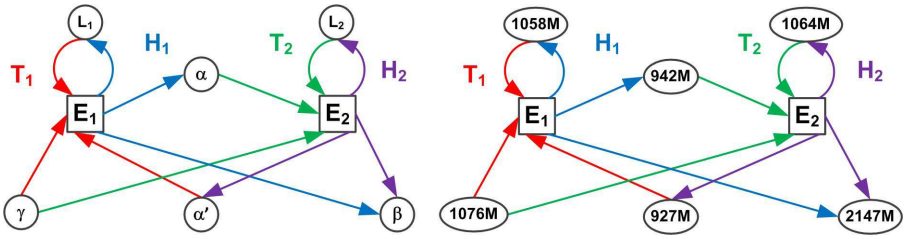


Fig. 5. (Left) Generic 2-motif. (Right) Instantiated with counts for days 2200-2299.

Beyond just identifying dirhypergraph motifs, we are interested in motifs which may or may not involve certain addresses, in our case, exchanges, and their distribution within certain kinds of patterns. In Fig. 4, exchanges are shown as black addresses. A **short thick band (STB)** is then a pattern where a quantity of Bitcoin is purchased from fiat currency, held for a while as Bitcoin, and then converted back to fiat currency. When an STB moves Bitcoin from one exchange address to a different one, we can call it **linear**; and when it returns it to the same exchange address, **circular**. More specifically, STBs are 2-motifs where:

- Two transactions intersect in an α or α' chain only;
- An exchange is included in both an input of the first and an output of the last transaction; but
- No exchange is an intermediate address in the transaction.

STBs could exist for many reasons, including financial speculation, simple user convenience, repeated purchases, remuneration, remittance, or fund management. Our interest is considering the hypothesis that STBs could be used as a potential laundering pattern. Moreover, we recognize that as a laundering pattern, it would not be very extensive. In this work we are begining with the simplest possible such pattern.

To formalize STBs, call a motif "pure" if only one pattern from Table 1 is present (this is *not* the case in the left side of Fig. 4), and otherwise "mixed". Then denote $e(a \in A)$ to mean that a is an exchange, and $e(X \subseteq A)$ to mean that X has an exchange: $\exists a \in X, e(a)$. We then can define an STB as follows.

Definition 1 (STB). *A 2-hypermotif $M = \{E_1, E_2\}$ is a* **linear STB** *if one and only one of the following holds:*

1. *It is a pure α 2-motif with $e(T_1) \wedge e(H_2) \wedge \neg e(H_1 \cap T_2)$; or*
2. *It is a pure α' 2-motif with $e(H_1) \wedge e(T_2) \wedge \neg e(T_1 \cap H_2)$.*

M is a **circular STB** *if one and only one of the following holds:*

1. *It is a mixed linear α and α' 2-motif with $e(T_1 \cap H_2) \wedge \neg e(H_1 \cap T_2)$; or*
2. *It is a mixed linear α' and α 2-motif with $e(H_1 \cap T_2) \wedge \neg e(T_1 \cap H_2)$.*

Note that no STB can have a self-loop. But because of the α, α' isomorphism noted above, it is sufficient to assume that a linear STB is a pure α pattern, and a circular STB is a mixed α, α' pattern, with

$$e(T_1) \wedge e(H_2) \wedge \neg e(H_1 \cap T_2), \quad e(T_1 \cap H_2) \wedge \neg e(H_1 \cap T_2)$$

respectively. Fig. 4 shows a linear (center) and circular (right) STB.

3 Descriptive Statistics

Given the nature of exchanges, and their primary function of converting between bitcoin and other currencies (including fiat and other alt coins), we question whether exchange addresses exhibit a different type of behavior from address owned by regular users of the network.

We downloaded the Bitcoin blockchain data using the Bitcoin Core Client,[7] and built a custom parser to convert the raw data into the dirhypergraph structure described in Section 2. We used data from the first transaction in the network up to April 22 2015, encompassing 72.7M unique addresses (vertices), involved in at least one of 66.3M transactions (hyperarcs) in our dirhypergraph. Addresses known to be exchanges were drawn from the WalletExplorer listing,[8] call these "labeled". Some exchanges are associated with several wallets ("current", "output", "old"). The full list we use is shown in Appendix A. While labeled addresses are presumed to be actual exchanges, the number of exchange addresses which are not listed as such is hard to judge for many reasons. At least, the WalletExplorer listings began on April 23 2011, while we know that exchanges have been around since 2010. Additionally, Mt. Gox, a substantial contributor over that time, was not included. There are still 2.44M labeled addresses (3.36 % of the total addresses), and 6.76M transactions involving an exchange (10.2 % of total transactions). Daily activity is summarized in Fig. 6, with addresses involved in several transactions in a single day counted once.

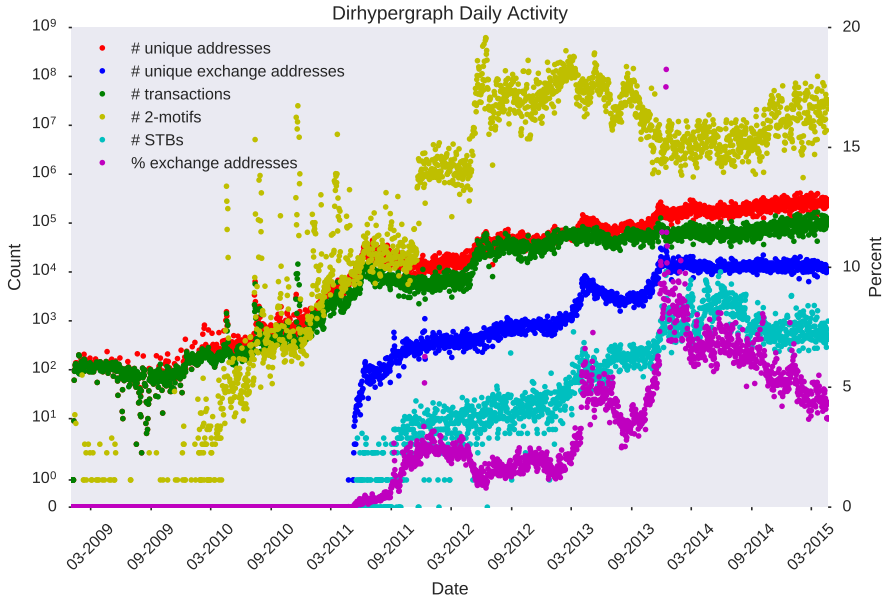


Fig. 6. Daily activity in each dirhypergraph.

[7] https://bitcoin.org/en/download

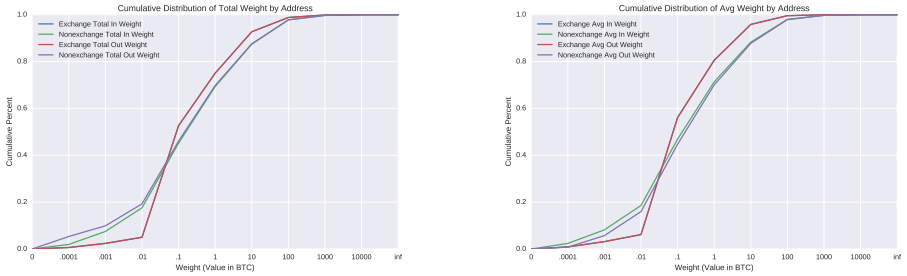
[8] https://www.walletexplorer.com, accessed January 16 2016.

Since our dirhypergraph presents as a bipartite graph of addresses and transactions, the in-degree of an address is actually the number of transactions on which an address serves as an output, and *vice versa* for out-degree. While the in- and out-degree distributions for both address types follow a power law, with the distributions having no significant difference using a 2-sample KS test, Table 2 shows several interesting aspects. Where 4.7% of unlabeled addresses are sinks – simply accumulating bitcoin and never redistributing it, resulting in an out-degree of zero – this drops to $< 0.1\%$ for labeled addresses. Also labeled addresses are more likely to have equal and positive in- and out-degrees. This behavior for labeled addresses is consistent with the use and function of exchanges, and for unlabeled addresses it is consistent with previous results [16], although to a much lesser extent. We attribute the decline in the proportion of sink addresses to the general growth of Bitcoin, but more importantly the sustained *trading phase* [6] it has been in, dwarfing activity in the *initial phase*.

Query	Labeled		Unlabeled	
In-degree > 0, out-degree $= 0$	2,123	0.087%	3,297,725	4.696%
In-degree > 0, out-degree > 0	2,435,472	99.91%	66,914,170	95.29%
In-degree $=$ out-degree > 0	2,356,530	96.67%	64,305,459	91.58%
In-weight > 0, out-weight $= 0$	2,123	0.087%	3,297,195	4.696%
In-weight > 0, out-weight > 0	2,435,472	99.91%	66,914,166	95.29%
In-weight $=$ out-weight > 0	2,421,944	99.35%	65,658,855	93.51%

Table 2. Comparing labeled and unlabeled address' degrees and weights.

Fig. 7 shows the cumulative distribution of weights in bitcoin (BTC), total on the left and average on the right. Unlabeled addresses have a much better separation between the in- and out-weight, suggesting that nonexchange addresses tend to keep a positive balance of bitcoin while exchanges keep zero, or near-zero, balances. The majority of exchange addresses have both an average and total transaction weight between 0.01 and 0.1 BTC, shown by the large jump in the figure. Roughly 50% of labeled addresses sit in this bucket, compared to about 30% of unlabeled addresses.

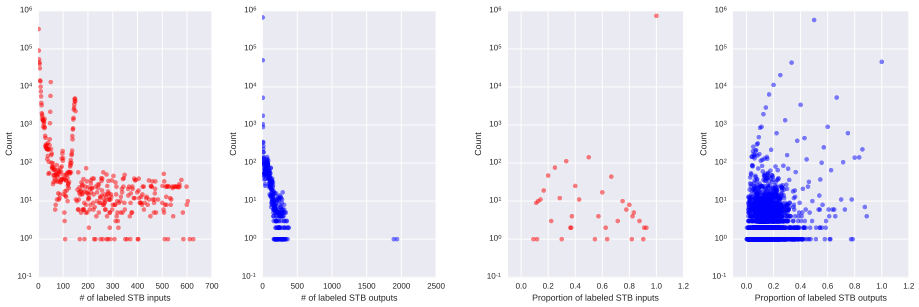


(a) (b)

Fig. 7. Cumulative percent of addresses with total (left) or average (right) input and output weights.

We next examine 2-motifs, STBs, and how exchanges are involved in them. If there are m transactions in a day, then there are at most $\binom{m}{2}$ possible 2-motifs. Of

our 40.0B 2-motifs, 10.3B are pure linear α or α' patterns, just 42.4M of which involve exchanges. 741K of those are STBs, including 727K linear and 13.4K circular. The volume of 2-motifs precludes the opportunity to do a comparison between STBs and non-STB 2-motifs, so instead we focus on just STBs.



(a) (b)

Fig. 8. Counts (left) and proportions (right) of labeled addresses on the inputs and outputs of STBs.

The number and proportion of addresses that are labeled as exchanges on the inputs and outputs is shown in Figure 8.[9] The number of labeled input addresses ranges from 1 to 635, while output addresses range from 1 to 1937. However, when the two clear outlier STBs are removed, the max number shrinks to 376.

According to the well known heuristic in Bitcoin to group all addresses that are inputs into the same transaction as being owned by the same entity [10, 14, 16], the expected proportion of labeled inputs should be 1, because if a single address is labeled, then all others are labeled as a consequence. Orders of magnitude more STBs do in fact have all inputs labeled, but many do not. This could be because the WalletExplorer data is incomplete, because it uses a different method for aggregating exchange wallets, or because they should in fact not be labeled. Yet, in every STB with multiple labeled input addresses, every address is owned by the same exchange – that is, every input is from a single exchange label. Interestingly, this *is not* the case for outputs of STBs. Of the 739.8K STBs, 63.4K (8.57%) have multiple exchange labels in the outputs ("multi-out STBs"), 48.1K (75.9%) of which have exactly two labels. Multiple exchanges on outputs of a single transaction could be due to mining pool payouts, but on outputs of STBs it is more likely indicative of inter-exchange activity.

4 Classifying and Labeling Exchange Addresses

While exchange addresses comprise a small percent of the Bitcoin network, they are of growing importance outside of the Bitcoin world, as they provide potentially the only avenue for connecting real-world people with pseudonymous

[9] Recall that this is at the address level, and each exchange has a set of addresses they own. The frequency of each exchange (e.g. BTC-e.com) in STBs is shown in Figure 10 in the Appendix, and is highly correlated with the number of addresses each exchange has, see Figure 11.

Bitcoin addresses. Being able to identify exchanges in the network is then a critical task, as it enables one to connect transactions or addresses of interest to the point at which they enter or exit the Bitcoin network.

We can leverage the different characteristics of exchange and non-exchange addresses to construct a machine learning model to classify an address as an exchange or not. In these experiments we use data from September 29 2011, roughly 100 days after exchanges first appear in our data, until April 22 2015. For every address we extract a set of features that numerically characterizes it, e.g. out-degree, total in-weight. Addresses are then assigned a class label corresponding to whether or not they are labeled as an exchange. The goal of the model is to learn a set of features and weights that can accurately discriminate between the two classes. Given the immense class imbalance (far fewer exchange addresses), we randomly sample an equal number of labeled and unlabeled addresses for training and testing the model. To account for the random sampling, 10 independent trials are run, and average results are reported.

Five different classifiers' results are summarized in Table 3.[10] AdaBoost and random forests perform the best, and are far superior to the remaining three, both yielding an F1 score of over 0.99. In the case of random forests, on average only 2,587 out of 972,866 test addresses were incorrectly predicted, falsely classifying 1,190 non-exchanges as exchanges, and 1,397 exchanges as non-exchanges. Moreover, the incredibly low variance of these models indicates they are much more robust than the others, performing well across all random samples.

Model	F1	Recall	Precision
Random Forest	0.9973 +- (0.0001)	0.9976 +- (0.0001)	0.9971 +- (0.0001)
AdaBoost	0.9944 +- (0.0001)	0.9974 +- (0.0001)	0.9915 +- (0.0003)
Linear SVM	0.8291 +- (0.0833)	0.8396 +- (0.1514)	0.8573 +- (0.1209)
Perceptron	0.2075 +- (0.3029)	0.3034 +- (0.4557)	0.2210 +- (0.2053)
Logistic Regression	0.0014 +- (0.0001)	0.0007 +- (0.0001)	0.2755 +- (0.0304)

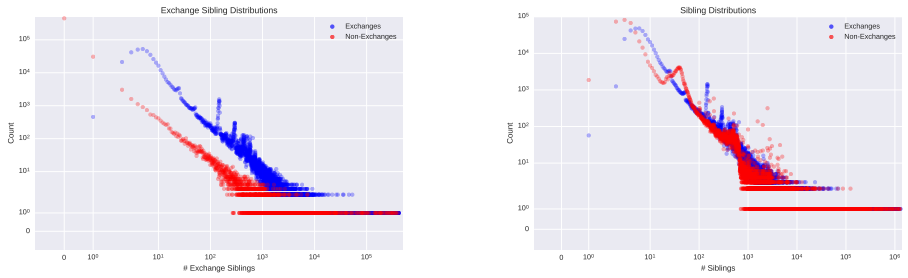
Table 3. Results for exchange address classification. All results shown are $mean+-std$ over the 10 runs.

Equally important, or perhaps even more important, than achieving such a high accuracy is understanding what it is about exchange addresses that facilitates the result. One way to quantify this is looking at the "feature importance" values that are calculated by the classifier. The top 5 features and their importance in the random forest model are: (1) # sibling exchanges (0.613); (2) # successor exchanges (0.184); (3) # predecessor exchanges (0.072); (4) # siblings total (0.044); (5) total out-weight (0.015).[11] It is not surprising that, by far, the most important feature is the number of exchange siblings. Figure 9a shows the substantial difference in the distributions for exchange siblings. Again, according

[10] All experiments were run using Python 2.7 and the scikit-learn and numpy packages.

[11] For an address a, siblings are addresses that have been a co-input or co-output, successors are addresses that have been an output when a was an input, and predecessors are addresses that were an input when a was an output.

to the common address group heuristic, if you are siblings with numerous exchange addresses then it is likely you are also an exchange address. Moreover, it is likely that you are an exchange address owned by the same exchange that your siblings are (c.f. Section 3). From a network science perspective, homophily [9] tells us that vertices of one type tend to interact with vertices of the same type.



(a) (b)

Fig. 9. Distribution of how many exchange addresses siblings (left) or total siblings (right) each address has. These distributions were drawn from a random sample of 100K exchanges and 100K non-exchanges.

Incorporating features related to the exchange labels clearly produces high quality results. However, it may restrict the capability of our model, failing to generalize well to new data which is not labeled, or handling incomplete labeling as we have in our dataset. To test this a second set of experiments is performed, identical to those described above except all features related to exchange labels are removed. Table 4 summarizes the new results.

Model	F1	Recall	Precision
Random Forest	0.8200 +- (0.0004)	0.8218 +- (0.0006)	0.8183 +- (0.0004)
AdaBoost	0.7941 +- (0.0012)	0.8264 +- (0.0033)	0.7643 +- (0.0018)
Linear SVM	0.3052 +- (0.2619)	0.3488 +- (0.3775)	0.5179 +- (0.1710)
Perceptron	0.1349 +- (0.2683)	0.1998 +- (0.3989)	0.1849 +- (0.1886)
Logistic Regression	0.0014 +- (0.0001)	0.0007 +- (0.0001)	0.3056 +- (0.0266)

Table 4. Results for exchange vertex classification when features related to exchanges are removed. All results shown are $mean + -std$ over the 10 runs.

Removal of the exchange features has an obvious negative impact on the accuracy of the classifiers. Random forest F1 score drops to 0.82 (a decrease of about .18), with the average number of incorrectly classified addresses increasing from 2587 (0.266%) to 175956 (18.086%). Similar to the runs that included exchange features, the average variance for the random forest classifier was very low. With the removal of exchange related features, structural features rose in importance. The new top 5 features and weights are: (1) # siblings (0.255); (2) total out-weight (0.232); (3) total in-weight (0.217); (4) # successors (0.092); (5) # predecessors (0.082). The former fourth and fifth ranked features are now the top two, and in conjunction with the total in-weight represent the majority of the discriminatory power. The top three now have a very equal share of importance,

indicating that the model relies on information from each of them instead of a single dominating feature. Figure 9 shows the distribution of the number of siblings for both exchange and non-exchange addresses. The distributions for less than 100 siblings are easily separable, but become much more intertwined when considering addresses with 100 or more siblings.

As we note in Section 3, the list of exchange addresses we have is incomplete. However, it is impossible to know exactly how incomplete the list is – whether we have 10% of the exchange addresses or 90%. A natural next question, then, is to try to classify all of the unlabeled addresses using our models constructed in the previous experiments.

All unlabeled addresses not used in training the classifiers were run through both random forest models and predicted as an exchange address or not (Table 5). The two classifiers yielded drastically different results. Using exchange label features, a mere 0.28% of the unlabeled addresses were labeled as exchanges. Conversely, 18.17% of the addresses were labeled as exchanges using the purely structural features. If instead of omitting the training addresses from the results we include them, the percent predicted raise to 3.98% and 21.87%. As 3.36% of the addresses are labeled from our ground truth data, this result is expected.

It is likely that 18% is a much better estimate for the true exchange address percent than 0.28%. The absence of Mt. Gox (among others) from our label data, and its historical dominance in Bitcoin, indicates that we are missing a large number of exchange addresses. Moreover, the extremely high accuracy combined with the extremely low prediction of unlabeled addresses of the first model suggests that the first classifier overfit the training data, exploiting the label features and becoming too reliant. Structural features, which we have perfect knowledge of for all addresses, are much more reliable and generalizable.

	With Label Features	Without Label Features
All addresses	3.98%	21.87%
Unlabeled addresses	0.28%	18.17%
Middle addresses 1-out STBs	1.34%	48.35%
Middle addresses multi-out STBs	0.68%	52.09%

Table 5. Percent of addresses classified as exchanges.

Our initial proposition of STBs as a laundering pattern stems from a user activity view of the network: a user receives bitcoin from an exchange, then converts it back into fiat currency, with the hope of obfuscating any money trail. From this perspective, addresses in the middle of an STB – which by definition cannot be labeled as an exchange – should be *less likely* to be predicted as an exchange than a randomly chosen unlabeled address. But (see Table 5) addresses in the middle of an STB are 2-3x *more likely* to be classified as an exchange than an a random unlabeled address. This directly contradicts our hypothesis, and instead is highly suggestive of lots of inter-exchange activity taking place. Self-churn [10] i.e. change-making is likely why an exchange address would be in the middle of what would otherwise be an STB. For example, an exchange E_1 sends bitcoin to one of its customers, making change for itself with the excess bitcoin

in the transaction, and then another exchange E_2 buys bitcoin from E_1, creating a 2-motif with exchanges on the input, middle, and output.

5 Conclusions and Future Work

In this work we make a first attempt at statistical and machine learning approaches that may be of interest in identifying laundering patterns, latent attribute classification, and discriminatory analysis. Directed hypergraphs are a sound model for transactions, and exchanges exhibit several patterns that are distinct from general address behaviors, as also shown in previous work. Our machine learning models are capable of labeling addresses as being owned by exchanges or not with very high accuracy, even when restricted to purely structural features. STBs are proposed as a potential laundering pattern, and shown to have a high degree of filtering when compared to the number of general 2-motifs in the network. Finally, we showed that middle vertices in STBs are much more likely to be classified as an exchange, indicating that there is a large amount of inter-exchange activity taking place.

Obvious areas of improvement include a much better label set, including Mt. Gox and generally being of higher fidelity. Similarly, an obvious area of expansion is to move beyond 2-motifs to 3-motifs, and consider triangular and other patterns involving three transactions. We have began the mathematical exploration of the 3-motif in directed hypergraphs, and it is somewhat complicated combinatorially, but manageable.

While we use learning to label an address as an exchange or not, the general tasks of latent attribute learning and discriminatory feature analysis impose no such constraint. A variety of customary labels may be of interest [10] – "mining pool", "wallet", "exchange", "vendor", "gambling" – in addition to your own personal labels – "suspicious", "country X". It is also not necessary to constrain the analysis to a single label at a time, but instead use multi-class classification models to predict from a set of labels. Moreover, instead of using a supervised learning approach where we assume our label sets are complete, we could explore methods such as PU Learning [7, 8], which account for imperfect data.

In addition to expanding the possible labels, the structure of interest could be expanded as well. Instead of looking at single addresses, transactions, or chains of transactions that form a higher level pattern, could be considered. For example, Möser et al. [12] show that some mixing services leave a distinct transactional pattern as a result of their mixing algorithms. Models for identifying similar patterns could be constructed using the hand curated transactions found in [12].

Acknowledgements

This material is based on work supported in part by the Department of Energy National Nuclear Security Administration under Award Number(s) DE-NA0002576. It is also supported in part under the Laboratory Directed Research and Development Program at the Pacific Northwest National Laboratory, a multi-program national laboratory operated by Battelle for the U.S. Department of Energy.

References

1. Anti-money laundering programs for money services businesses, 31 C.F.R. § 1022.210
2. Compliance and exemptions, and summons authority, 31 U.S.C. § 5318
3. Ausiello, G., Fanciosa, P.G., Frigioni, D.: Directed hypergraphs: Problems, algorithmic results, and a novel decremental approach. In: ICTCS 2001, LNCS, vol. 2202, pp. 312–328 (2001)
4. FATF: Virtual currencies key definitions and potential AML/CFT risks. Tech. rep. (2014)
5. Gallo, G., Longo, G., Pallottino, S.: Directed hypergraphs and applications. Discrete Applied Mathematics 42, 177–201 (1993)
6. Kondor, D., Pósfai, M., Csabai, I., Vattay, G.: Do the rich get richer? an empirical analysis of the bitcoin transaction network. PloS one 9(2), e86197 (2014)
7. Li, X.L., Liu, B.: Learning from positive and unlabeled examples with different data distributions. In: European Conference on Machine Learning. pp. 218–229. Springer (2005)
8. Liu, B., Lee, W.S., Yu, P.S., Li, X.: Partially supervised classification of text documents. In: ICML. vol. 2, pp. 387–394. Citeseer (2002)
9. McPherson, M., Smith-Lovin, L., Cook, J.M.: Birds of a feather: Homophily in social networks. Annual review of sociology pp. 415–444 (2001)
10. Meiklejohn, S., Pomarole, M., Jordan, G., Levchenko, K., McCoy, D., Voelker, G.M., Savage, S.: A fistful of bitcoins: characterizing payments among men with no names. In: Proceedings of the 2013 conference on Internet measurement conference. pp. 127–140. ACM (2013)
11. Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskil, D., Alon, U.: Network motifs: Simlpe building blocks of complex networks. Science 298, 824–827 (2002)
12. Möser, M., Böhme, R., Breuker, D.: An inquiry into money launder tools in the bitcoin ecosystem. In: eCrime Researchers Summit. pp. 6–24. Springer (2013)
13. Möser, M., Böhme, R., Breuker, D.: Towards risk scoring of bitcoin transactions. In: International Conference on Financial Cryptography and Data Security. pp. 16–32. Springer (2014)
14. Ober, M., Katzenbeisser, S., Hamacher, K.: Structure and anonymity of the bitcoin transaction graph. Future internet 5(2), 237–250 (2013)
15. Reid, F., Harrigan, M.: An analysis of anonymity in the bitcoin system. In: Security and privacy in social networks, pp. 197–223. Springer (2013)
16. Ron, D., Shamir, A.: Quantitative analysis of the full bitcoin transaction graph. In: International Conference on Financial Cryptography and Data Security. pp. 6–24. Springer (2013)
17. U.S. Department of the Treasury, FinCEN: FinCEN fines ripple labs inc. in first civil enforcement action against a virtual currency exchanger (May 2015), https://www.fincen.gov/news/news-releases/fincen-fines-ripple-labs-inc-first-civil-enforcement-action-against-virtual

A Exchanges Used

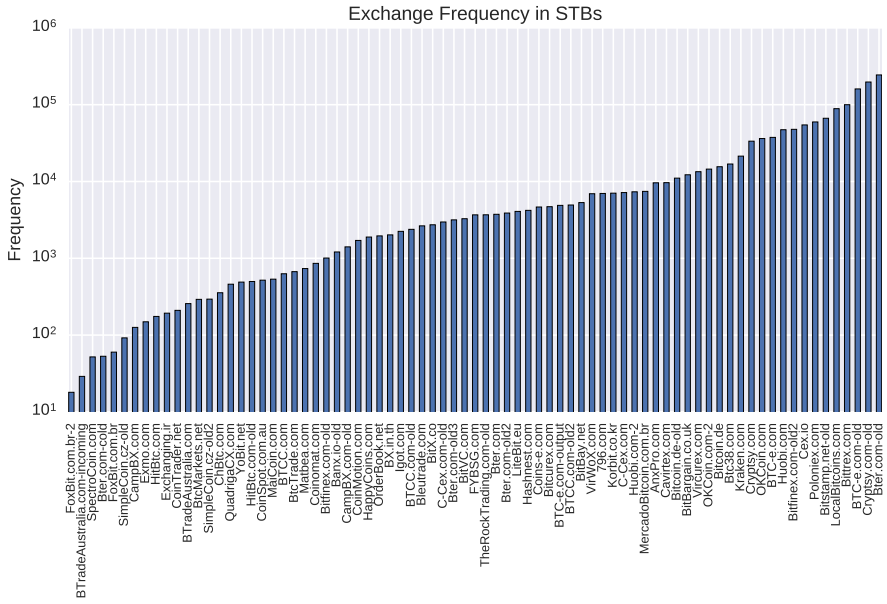


Fig. 10. Frequency of exchanges in STBs.

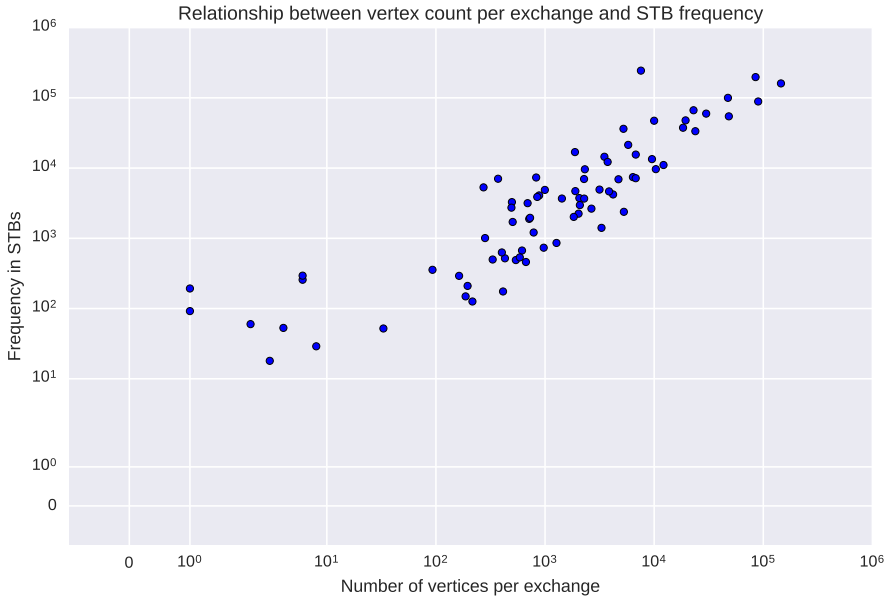


Fig. 11. Relationship between the number of vertices owned by each exchange and the frequency of that exchange in STBs.

B Features Used

An address' feature matrix is composed of the following features, extracted from each day of the data and then aggregated. Features prefixed with "*" are those removed in the second experiment, where exchange label based features are removed.

1. *total_bitcoin_received* – How much BTC the address received from transaction outputs over the full time window.
2. *total_bitcoin_spent* – How much BTC the address spent as transaction inputs over the full time window.
3. *bitcoin_balance* – Total bitcoin received minus total bitcoin spent.
4. *num_predecessors* – How many unique addresses have been an input to transactions where this address was an output.
5. *num_transaction_outputs* – How many times this address has been used in a transaction output.
6. *num_successors* – How many unique addresses have been an output in transactions where this address was an input.
7. *num_transaction_inputs* – How many times this address has been used as a transaction input.
8. *num_siblings* – How many unique addresses have been co-inputs or co-outputs with this address.

9. *$num\_predecessor\_exchanges$ – How many unique exchange addresses have been an input to transactions where this address was an output.
10. *$num\_successor\_exchanges$ – How many unique exchange addresses have been an output in transactions where this address was an input.
11. *$num\_sibling\_exchanges$ – How many unique exchange addresses have been co-inputs or co-outputs with this address.
12. $num\_gamma\_patterns$ – How many times this address is part of a γ pattern.
13. $num\_beta\_patterns$ – How many times this address is part of a β pattern.
14. $num\_L1\_patterns$ – How many times this address is part of an L_1 pattern.
15. $num\_L2\_patterns$ – How many times this address is part of an L_2 pattern.
16. $num\_alpha\_patterns$ – How many times this address is part of a α pattern.
17. $num\_alphaprime\_patterns$ – How many times this address is part of a α' pattern.
18. $reciprocity$ – How many of this addresses successors are also predecessors.
19. $anti\_reciprocity$ – How many of this addresses predecessors are also successors.

We focused on local features that are fast to compute. Examples of more expensive but potentially very useful features are explained in [13], e.g. peeling chains or coinbase transactions.