IEEE ICBC'24 1570981989

# Shared Send Mixers Untangling in Bitcoin Clustering Heuristics Adjustment

***Abstract*—An address in the Bitcoin blockchain serves as an identifier for spending cryptocurrency. The blockchain itself does not contain information about the actual users who control the assets. Users have the ability to create multiple addresses, leading to the challenge of grouping addresses, also known as clustering, in order to analyze Bitcoin users. The grouping problem is a crucial initial step in the analysis of Bitcoin users. The grouping solution involves using heuristics based on usage patterns, with a focus on Common Spending (CS) and One-Time Change (OTC) in the current research. Additionally, anonymization techniques such as Shared Send Mixers (SSM) are considered in this paper as they prevent or at least complicate analysis. It is possible to untangle SSM, and based on the number of untanglings per transaction and their size, the transaction can be classified into a certain complexity class. Both heuristics and untangling were applied to the Bitcoin transaction history in our study. Our findings revealed that OTC misuse may occur in 25 to 70 percent of cases, depending on the specific algorithm used. Furthermore, CS and OTC were found to generate 4 to 0.04 percent of new cases when applied to subtransactions of separable SSM. Additionally, we demonstrated that SSM address grouping respects untangling complexity classes. In the future, we plan to adapt our workflow to other Bitcoin-like blockchains and modify our untangling algorithm to cover more SSM transactions.**

***Index Terms*—Bitcoin, blockchain, common spending, one-time-change, shared send mixer, anonymity, knowledge discovery**

## I. Introduction

The Bitcoin blockchain address serves as an identifier for cryptocurrency transactions [1]. From the privacy perspective, the best practice is to use each address only be used twice: once to receive cryptocurrency and once to spend it, leading users to have multiple addresses. To analyze Bitcoin user data, addresses can be grouped by user, reducing the problem size and providing more information about individual users [2].

The Bitcoin blockchain does not contain any data on users, but the transaction model of Bitcoin motivates cryptocurrency usage patterns. In Bitcoin, each transaction generates unspent transaction outputs (UTXOs), and these UTXOs are then spent as inputs in subsequent transactions. The exception is the first transaction in each block known as a coinbase transaction, which has no inputs but only outputs as a miner reward.

The UTXO model makes the circulation of Bitcoin similar to the circulation of banknotes, but Bitcoin allows for arbitrary nominals. Just like with banknotes, it is possible to combine several small UTXOs into a larger one, a pattern known as Common Spending (CS). Additionally, when making a purchase, there may be a need for change back, which is known as One-Time Change (OTC). Both CS and OTC are of interest in our paper.

CS and OTC may contain errors due to Shared Send Mixers (SSM) [3], [4], which were used in custodial wallets until 2017 to reduce transaction fees and are still used as an anonymization technique. If SSM only allows for subtransaction splitting–untangling [5], it may indicate misuse of OTC. Additionally, CS and OTC can be applied to subtransactions after untangling, if the original transaction was impossible.

The goal of this paper is to examine the impact of SSM on CS and OTC heuristics. To achieve this, CS, three popular versions of OTC, and SSM untangling were applied to Bitcoin data up to August 2022. A ground truth OTC dataset was also collected from [6]. For OTC-flagged transactions, their untanglability was checked. For all untangled transactions, CS and OTC were applied.

The main contributions of the paper to the Bitcoin transaction analysis are

1) Provide evidence of OTC misuse through SSM untangling.
2) Present new use cases for CS and OTC based on SSM untangling.
3) Prove that address grouping does not increase SSM complexity.

The rest of the paper is organized as follows. Section II dives into the related work. The previous work notations and formal definitions are summarized in Section III, with transaction examples being the only novel part. Section IV details on relation between heuristics and untangling, and proves that address grouping respects untangling complexity classes. We analyze Bitcoin transaction history from both heuristics and untangling points of view in Section V. Finally, Section VI concludes the paper.

## II. Related Work

Address clustering is a crucial step in Bitcoin transaction history analysis, as it allows for a reduction in dataset size and an increase in available information about individual samples. The Unspent Transaction Output (UTXO) model provides general usage patterns that allow for the grouping of addresses by owners. However, with the evolution of the Bitcoin transaction model and key management software, such as Pay to Script Hash [7], Segregated Witness [8] and Hierarchical Deterministic Wallets [9], these usage patterns also change over time. Nevertheless, the main principles remain the same, and we will review them in the current section.

In [2], common spending (CS) and one-time change (OTC) heuristics were formulated, along with ways to evade them. CS occurs when a user combines multiple UTXOs to send them to

a single output, often when the payment amount exceeds the value of each UTXO. OTC occurs when the amount to spend does not match the available UTXOs, resulting in the user receiving change back. While no effective evading techniques were provided at the time of the study, practical transactions with multiple senders were mentioned as a promising direction, which has since been implemented by Bitcoin developers.

Another paper by Reid et al. [10] provides a comprehensive check on Bitcoin users, including the aforementioned heuristics and vulnerabilities at the TCP/IP layer. While many of the issues beyond heuristics are outdated due to transaction propagation modifications and the spread of TOR browser, the authors emphasize their research motivation not to de-anonymize individual users, but rather to demonstrate the inherent limits of anonymity when using Bitcoin.

In their paper [11], the authors considered current trends in Bitcoin usage and introduced an OTC heuristic in its modern form. They focused on improving its low accuracy by refining it with a new address requirement and the absence of intersection between input and output address sets. The numerical evaluation demonstrated that the modifications reduced false positives by $13\%$ for the one-time change heuristic.

The paper [12] developed its own version of the OTC heuristic. Notably, they excluded transactions with two inputs and two outputs, as they are most likely SSM. The authors emphasized the importance of considering heuristics not only individually but also in terms of their impact on clustering. They highlighted that heuristics provide evidence to group addresses, while offchain data can provide evidence not to group certain addresses together. By considering all the evidence for and against grouping with a probabilistic model, researchers can obtain more reliable clustering results.

In their paper, the authors [13] created ransomware cluster-specific filters. Their inflow filters identify potential ransom payments from real victims and estimate ransomware revenue. One filter identifies inflows consistent with known ransom payment patterns, while another filter includes inflows that send bitcoins from an exchange cluster to a ransomware cluster. These filters serve as additional heuristics for clustering and can provide additional information for CS and OTC heuristics given offchain data.

Möser and Narayanan's study on OTC versions, as presented in [6], is the most recent and comprehensive. The study examines different versions that consider output value in relation to the inputs, address type, amount rounding, and other consistent fingerprints. The authors also introduced a change address dataset labeled based on the UTXO spending followed by the change, serving as a ground truth for evaluation. Additionally, they provided an evaluation of heuristics versions and a machine learning model for identifying change addresses.

In their work, [14], the authors focused on improving OTC for Bitcoin clustering. They introduced seven heuristics to combine into OTC and utilized the coverage ratio, which measures the fraction of transactions marked as OTC by one algorithm among the OTC transactions identified by another algorithm, to compare the combinations of heuristics.

The demand for multi-user transactions from a security perspective arose with the early adoption of Bitcoin, as discussed above for [11]. The merging of transactions into one, known as CoinJoin [15] , and more generally as shared send mixers (SSM), has been used to tangle transaction history and reduce fees per transfer. These features made SSM a part of custodial wallets before 2017, and Europol claims they are still in use [16], [17].

The concept of SSM untangling was formulated in [3]. The paper [5] presented the problem mathematically, analyzed its complexity class, and computed statistics for a selected month. However, the authors did not provide pseudo code or source code for the algorithm. In [18] SSM untangling, if the result is unique, was used as part of a Bitcoin tracing design. The authors provided a link to the source code on Google Disc, but as of late 2023, it is inaccessible. Larionov and Yanovich [19] provided an untangling algorithm with source code and demonstrated SSM spread throughout the entire Bitcoin transaction history.

As clustering heuristics such as CS and OTC are prone to errors, the common practice is to skip SSM transactions during clustering. Möser and Böhme [20] numerically analyzed JoinMarket, a CoinJoin-based marketplace for more anonymous transfers, and demonstrated its viability both theoretically and on empirical data. the The researchers [21] analyzed the privacy model of the marketplace and presented a custom heuristic to identify such transactions among others. This heuristic was further utilized in [22] to avoid SSM in OTC. So skiped tangled SSM the authors in above-mentioned [18].

In addition to errors in heuristic applications, untangled SSM can reveal new information for subtransactions. Therefore, the focus of our current research is to examine the impact of SSM untangling on Bitcoin address clustering via CS and OTC heuristics.

## III. Clustering View on Transactions

This section provides an introduction to the notations and existing results. In this paper, we have chosen to use the notations from [19] as they are suitable for addressing the untangling requirements, which involve a higher level of notation complexity. Additionally, we have also incorporated the notations commonly used for clustering heuristics.

### A. Untangling

A Bitcoin address serves as a unique identifier for a UTXO payment instruction. Typically, this instruction includes public key(s) information and a payment script template chosen from a limited set, such as P2PK, P2PKH, P2SH, and their Segregated Witness versions [8]. It is important to understand that while the address provides information, it is primarily treated as an identifier. In each transaction, an address and a non-negative amount are used to represent the input and output. To differentiate between them, we use lowercase letters for amounts, uppercase letters for addresses, and calligraphic capital letters for multisets of amount and address pairs. For instance, $(c_k, C_k)$ denotes an input or output with the

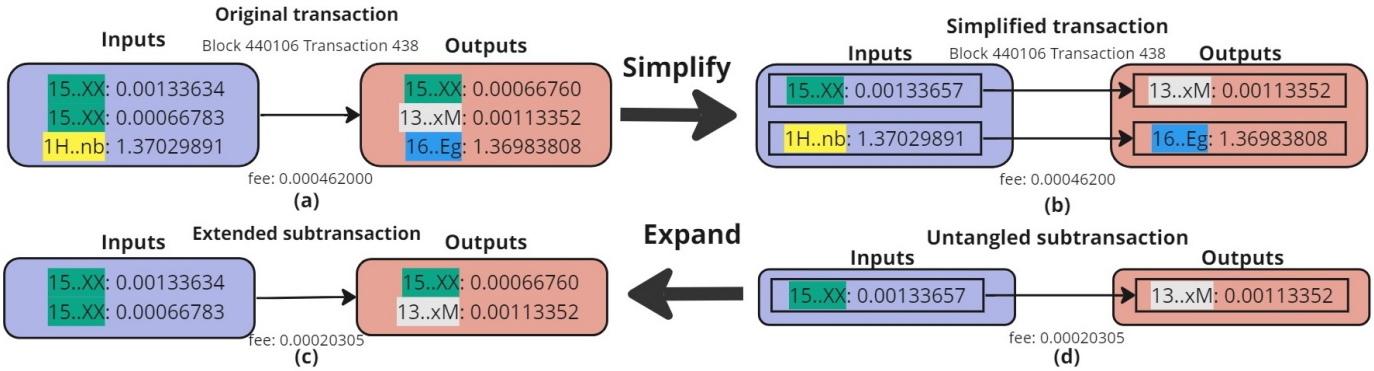


Fig. 1. Transaction 438 from block 440106. Left-to-right, top-to-bottom: (a) $t$, (b) $\texttt{Simplify}(t)$ with untangling (*separable*), (c) subtransaction $\tau = \texttt{Expand}(t, \tau')$ (see Section IV), and (d) subtransaction $\tau'$. The addresses are highlighted by different colors

address $C_k$ and the amount $\mathcal{C} = \cup_{k=1}^{K}\{(c_k, C_k)\}$ represents a collection of such pairs.

Coinbase transactions are not relevant to the scope of this paper as they do not involve inputs, are not shared send mixers, and do not appear after the untangling process.

A Bitcoin transaction is represented as an ordered triple $t = (\mathcal{A}, \mathcal{B}, c)$, where

- $\mathcal{A}$ is a multiset of transaction inputs, where each input $(a_n, A_n) \in \mathcal{A}$ consists of an ordered pair with the address $A_n$ and the value of the input $a_n \geq 0$.
- $\mathcal{B}$ is a multiset of transaction outputs, where each output $(b_m, B_m) \in \mathcal{B}$ consists of an ordered pair with the address $B_m$ and the value of the output $b_m \geq 0$.
- $c = \sum\limits_{(a_n,\cdot)\in\mathcal{A}} a_n - \sum\limits_{(b_m,\cdot)\in\mathcal{B}} b_m \geq 0$ represents the transaction fee.

For any arbitrary multiset of transaction inputs or outputs $\mathcal{C}$, $\texttt{Addr}(\mathcal{C}) = \cup_{(\cdot,C)\in\mathcal{C}}\{C\}$ denotes the multiset of addresses in $\mathcal{C}$, and $\texttt{Sum}(\mathcal{C}) = \sum_{(c,\cdot)\in\mathcal{C}} c$.

An address can be utilized multiple times within the set of inputs and/or outputs. To streamline this process during untangling, a `Simplify` function is used to consolidate multiple usages into one. `Simplify` processes the multisets of inputs and outputs in the following manner (see Figure 1-a and 1-b)

1) The inputs are grouped by addresses and the sums of each group are calculated. The same is done for the outputs.
2) For each address $C$ that exists both as an input and an output, the smaller amount between the input and output is substituted for both.
3) Any pairs with zero amounts are discarded.

I.e., $\texttt{Simplify}: t \mapsto t'$, where

$$
\begin{aligned}
&t = (\mathcal{A}, \mathcal{B}, c),\ t' = (\mathcal{A}', \mathcal{B}', c'),\\
&\mathcal{A}' = \{(-\texttt{Bal}(t, A), A) | A \in \texttt{Addr}(\mathcal{A}) \wedge \texttt{Bal}(t, A) < 0\},\\
&\mathcal{B}' = \{(\texttt{Bal}(t, B), B) | B \in \texttt{Addr}(\mathcal{B}) \wedge \texttt{Bal}(t, B) > 0\},\\
&c' = c,
\end{aligned}
$$

and $\texttt{Bal}$ is the balance function

$$\texttt{Bal}(t, C) \equiv \sum_{(b,C)\in\mathcal{B}} b - \sum_{(a,C)\in\mathcal{A}} a.$$

The untangling deals with simplified transactions. We omit any special notations for the simplified transactions then possible.

Let $|\cdot|$ be the cardinality operator. Denote $N \equiv |\mathcal{A}|$ and $M \equiv |\mathcal{B}|$.

A transaction $t = (\mathcal{A}, \mathcal{B}, c)$ can be represented by a bipartite graph $G(t)$, which shows the flow of coins among participants:

- The first set of vertices represents $\mathcal{A}$, and the second set represents $\mathcal{B}$.
- Each vertex $C$ has a weight equal to the corresponding amount $c$.
- Two vertices $A$ and $B$ are connected by an edge if the entity controlling the input $A$ authorized spending to $B$.

I.e., $G(t) = (V, E),\ V = \mathcal{A} \cup \mathcal{B},\ E \subseteq \mathcal{A} \times \mathcal{B}$. And

**Assumption 1.** *Intended expenses of each input subset in the transaction do not exceed their actual expenses:*

$$\forall \mathcal{B}' \subset \mathcal{B}\colon \sum_{\substack{(a,A)\in\mathcal{A}\colon\ \exists(b,B)\in\mathcal{B}'\\ \wedge((a,A),(b,B))\in E}} a \geq \sum_{(b,B)\in\mathcal{B}} b.$$

Public information about a transaction includes all the details about the vertices in the transaction graph. However, it does not provide any information about the edges between these vertices. Therefore, the objective of shared send analysis is to infer or reconstruct the missing edges based on the available vertices and potentially additional data.

Hereafter, we will represent transactions as a set of input addresses displayed in purple boxes on the left, and a set of output addresses displayed in pink boxes on the right. An input box is connected to an output box if the coins flow from the input to the output. To avoid confusion, we only show the first and last two symbols of the addresses. In Bitcoin, calculations are done using integer arithmetic in satoshis, where 1 BTC is equal to $10^8$ satoshis. We display amounts in BTC with eight decimal places for alignment and

scale. The fee for a transaction is listed at the bottom of the figure. When referring to a committed transaction, we use its position in the blockchain, which includes the block height and index.

**Definition 1.** *A (simplified) transaction $t = (\mathcal{A}, \mathcal{B}, c)$ is called a shared send transaction iff (if and only if) it has multiple inputs and multiple outputs: $N \equiv |\mathcal{A}| > 1$ and $M \equiv |\mathcal{B}| > 1$. If $N \equiv |\mathcal{A}| \leq 1$ and $M \equiv |\mathcal{B}| = 1$, the transaction is not a shared send, and we call it regular.*

The recovery of edges, denoted as $G(t)$, can have multiple solutions. It is always possible to construct a complete graph. In the case of a shared spend mixing transaction, we can assign a connected component for each participating entity. The objective of untangling is to construct all the potential graphs.

**Definition 2.** *Given a transaction $t = (\mathcal{A}, \mathcal{B}, c)$, a pair of sets $\mathcal{A}' \subset \mathcal{A}$ and $\mathcal{B}' \subset \mathcal{B}$, with at least one non-empty set, is called connectable iff the following condition holds:*

$$\mathtt{Sum}(\mathcal{B}') + c \geq \mathtt{Sum}(\mathcal{A}') \geq \mathtt{Sum}(\mathcal{B}').$$

That is, connectivity means that bitcoins from the addresses in $\mathcal{A}'$ were spent to the addresses in $\mathcal{B}'$, and from $\mathcal{A} \setminus \mathcal{A}'$ to $\mathcal{B} \setminus \mathcal{B}'$.

A collection of sets $\{X_k\}_{k=1}^K$ is called a partition of a set $X$ if the sets in the collection are pairwise disjoint, and their union equals $X$. We denote a partition as $X = \sqcup_{k=1}^K X_k$.

**Definition 3.** *Consider a transaction $t = (\mathcal{A}, \mathcal{B}, c)$. A pair of $K$-element partitions of input and output sets, i.e.*

$$\mathcal{A} = \sqcup_{k=1}^K \mathcal{A}_k, \quad \mathcal{B} = \sqcup_{k=1}^K \mathcal{B}_k,$$

*is called acceptable iff each pair $(\mathcal{A}_k, \mathcal{B}_k)$ is connectable.*

We will denote partitions in the form

$$\mathcal{P} = \{(\mathcal{A}_1, \mathcal{B}_1), \ldots, (\mathcal{A}_K, \mathcal{B}_K)\},$$

and say that sets $\mathcal{A}_k$ and $\mathcal{B}_k$ in the partition correspond to each other.

As merging any pair of partition's connectable pairs into one results in a valid partition, the number of possible partitions may be huge. The untangling considers only minimal partitions to reduce the number of untanglings to consider and to make a solution feasible.

**Definition 4.** *A connectable pair $(\mathcal{A}, \mathcal{B})$ is called minimal iff the sets in it does not allow smaller partitions: $\forall \mathcal{A}_1, \mathcal{A}_2, \mathcal{B}_1, \mathcal{B}_2 \colon \mathcal{A} = \mathcal{A}_1 \sqcup \mathcal{A}_2$ and $\mathcal{B} = \mathcal{B}_1 \sqcup \mathcal{B}_2 \rightarrow$ at least one pair $(\mathcal{A}_1, \mathcal{B}_1)$ or $(\mathcal{A}_2, \mathcal{B}_2)$ is not connectable.*

**Definition 5.** *An acceptable partition $\mathcal{P} = \{(\mathcal{A}_k, \mathcal{B}_k)\}_{k=1}^K$ is called minimal iff it cannot be further subdivided into an acceptable partition, i.e., $(\mathcal{A}_k, \mathcal{B}_k)$ is a minimal connectable pair for each $k = 1, \ldots, K$.*

Using the concept of minimal partitions, the untangling problem is

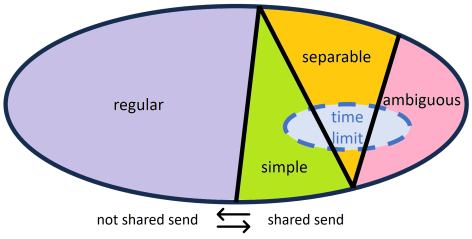


Fig. 2. Bitcoin transaction types in relation to SSM

**Definition 6.** *The minimal untangling problem is, based on the transaction $t = (\mathcal{A}, \mathcal{B}, c)$, to produce every minimal partition of the transaction graph $G(t)$.*

By accepting such a restriction, we implicitly assume that a non-mixing transaction is unlikely to look like a mixing transaction. Furthermore, the value flows are more likely to correspond to a minimal acceptable partition than to a non-minimal partition extensible to this minimal partition, as additional measures need to be taken to make value flows dividable.

**Definition 7.** *Depending on the result of the minimal untangling formulation of the shared send analysis problem, a shared send transaction $t = (\mathcal{A}, \mathcal{B}, c)$ is called:*

- *Simple iff the minimal partition is $\mathcal{P} = \{(\mathcal{A}, \mathcal{B})\}$. I.e., we can not find non trivial disconnected components.*
- *Separable iff the unique minimal partition $\mathcal{P}$ exists and $|\mathcal{P}| > 1$.*
- *Ambiguous iff there are at least two different minimal partitions.*

The white paper [5] presented a necessary condition for identifying *ambiguous* transactions and demonstrated that detecting *ambiguous* shared send transactions is an NP-complete problem. The NP-completeness implies that it is unlikely to find an algorithm that is faster than brute force. However, executing the brute force method for all input and output subsets may be infeasible for certain transactions due to the large search space. To address this issue, untangling algorithms have been developed to terminate execution when a computational time budget is reached, resulting in a semi-formal class of transactions called time limit . The classification of transactions as time limit depends on the capabilities of the specific solver being used. For a solver with unlimited computational resources, all shared send transactions would fall into one of the classes: *simple*, *separable*, or *ambiguous*. In the paper [19], an untangling algorithm is provided. In our paper, we do not contribute to the untangling process and instead utilize the code from the aforementioned paper. Therefore, our classification of time limit transactions aligns with that of the referenced paper, with minor deviations due to differences in hardware configurations.

Hence, a Bitcoin transaction is *regular*, *simple*, *separable*, *ambiguous*, or *time limit* (see Figure 2). *Time limit* is a denial of the transaction classification into *simple*, *separable*

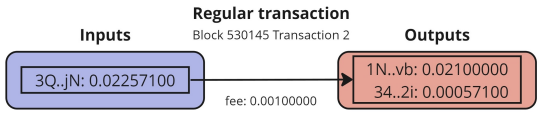


Fig. 3. Transaction 2 from block 530145 is *regular* as the number of inputs $N = 1$

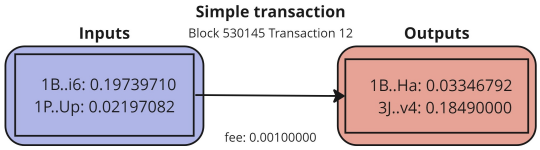


Fig. 4. Transaction 12 from block 530145 is *simple* as the input to get enough funds for the output $(0.02197082, \texttt{1P..Up})$ is not sufficient for either output

or *ambiguous* due to the computational limitations. *Regular*, *simple*, *separable*, and *time limit* are types of *shared send transactions*. Transaction examples are on Figures 3, 4, 5, and 6.

### B. Heuristics

Clustering heuristics are used to identify addresses that are likely owned by a single entity (user) based on their bitcoin usage patterns. We can denote the user of an address $C$ as $u(C)$, where $u(C)$ is an identifier, such as an integer number. The two common heuristics used to address the issue of untangling are common spending (CS) and one-time change (OTC) [2].

CS occurs when a user combines multiple UTXOs to send them to a single output, often when the payment amount exceeds the value of each UTXO (see Figure 7). I.e, if in an (unsimplified) transaction $t = (\mathcal{A}, \mathcal{B}, c)$ the number of outputs $|\mathcal{B}| = 1$, then

$$\forall A, A' \in \texttt{Addr}(\mathcal{A}) : \texttt{user}(A) = \texttt{user}(A').$$

OTC occurs when the amount to spend does not match the available UTXOs, resulting in the user receiving change back (see Figure 8). Let the set of the new address of the transaction $t = (\mathcal{A}, \mathcal{B}, c)$ be denoted as $\texttt{new}(\mathcal{B})$, where $C \in \texttt{new}(t)$ means $C$ has never occurred in the transaction log before $t$. Since each input spends UTXO, a new occurrence is possible only among outputs $\texttt{new}(t) \subseteq \texttt{Addr}(\mathcal{B})$. While the CS heuristic is straightforward, OTC needs to identify a change address, which allows for numerous interpretations. We do not aim to provide a comprehensive study on all possible versions of OTC, but rather to show the potential synergy between untangling and heuristics. Therefore, we will only consider a few alternatives.

**Definition 8** (OTC-1 [2])**.** *Consider a transaction* $t = (\mathcal{A}, \mathcal{B}, c)$*. If*

- $|\texttt{Addr}(\mathcal{B})| = 2$,
- $|\texttt{new}(t)| = 1$,

*then the only new output* $(b, B) \in \texttt{new}(t)$ *is a change address.*

More than two output addresses indicate a deviation from the pattern of a single payment with a change.

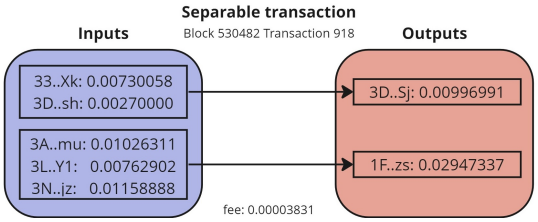


Fig. 5. Transaction 918 from block 530482 is *separable*

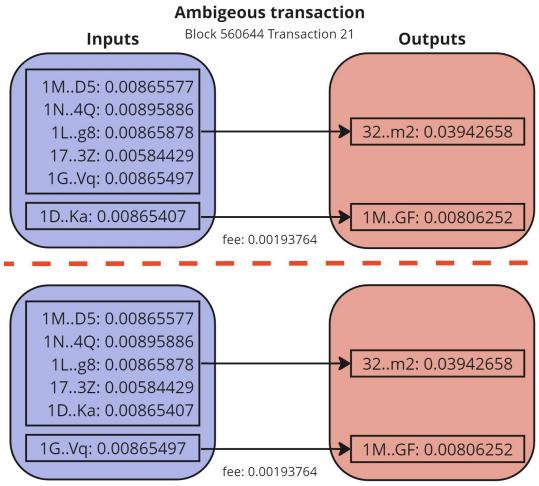


Fig. 6. Transaction 21 from block 560644 is *ambiguous* with two different untanglings separated by a dashed red line

**Definition 9** (OTC-2 [12])**.** *Consider a transaction* $t = (\mathcal{A}, \mathcal{B}, c)$*. If*

- $|\texttt{Addr}(\mathcal{B})| = 2$,
- $|\texttt{new}(t)| = 1$*. Let us denote it as* $B_1$*, resulting in i.e.,* $\mathcal{B} = \{(b_1, B_1), (b_2, B_2)\}$,
- $\texttt{Addr}(\mathcal{A}) \neq 2$,
- $B_2 \notin \texttt{Addr}(\mathcal{A})$,
- $b_1$ *has at least* 4 *decimals,*
- $B_2$ *has not been a change address in previous transaction log,*

*then the only new output* $(b_1, B_1) \in \texttt{new}(t)$ *is a change address.*

OTC-2 excludes $\texttt{Addr}(\mathcal{A}) = 2$ as it is likely SSM. If $B_2 \in \texttt{Addr}(\mathcal{A})$, then $B_2$ is a self-change address, and there is no need for a new change address. The fewer decimals there are in the amount, the more aesthetically pleasing it appears to humans. It is also unlikely that the user will pay attention to the change address, as it is usually generated automatically. The BTC exchange rate to traditional currencies may affect this condition, but we do not address it in the current paper. Additionally, the usage of $B_2$ as a change address in the previous log may indicate its continued usage as a change address.

**Definition 10** (OTC-3 [6])**.** *Consider a transaction* $t = (\mathcal{A}, \mathcal{B}, c)$*. If*

- $|\texttt{Addr}(\mathcal{B})| = 2$,

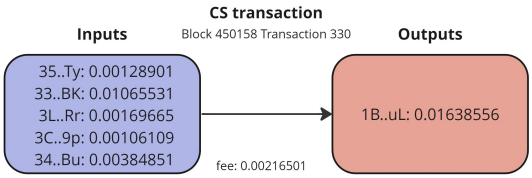


Fig. 7. Transaction 330 from block 450158 is CS

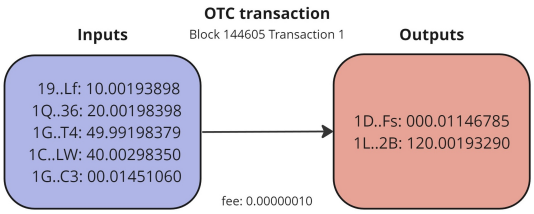


Fig. 8. Transaction 1 from block 144605 is OTC according to the definition 8

- $|\texttt{new}(t)| = 1$. *Let us denote it as* $B_1$*, resulting in i.e.,* $\mathcal{B} = \{(b_1, B_1), (b_2, B_2)\}$,
- $\forall i, j : 1 \le i < j \le N$ *the change candidate amount* $b_1 < a_i + a_j$,

*then the only new output* $(b_1, B_1) \in \texttt{new}(t)$ *is a change address.*

If there exist $i$ and $j$ such that $1 \le i < j \le N$ and the change candidate amount $b_1 \ge a_i + a_j$, then it is not advisable to include it as change in the transaction. This is because the sender can eliminate both inputs from the transaction, thereby reducing the change amount. The paper [6] provides additional conditions and their combinations, such as the number of decimals in the change address and coinciding address types. Our goal is to consider various combinations to determine if the combination of untangling and clustering heuristics is worth exploring. We also aim to ensure that the algorithms remain operational on isolated transactions whenever possible, in order to assess the feasibility of the methods' synergy, even at the early stage of the clustering analysis.

## IV. Untangling Meets Heuristics

Both untangling and heuristics can complement each other and provide valuable insights. Untangling helps identify the transaction type, providing additional information based on its nature. However, for *simple* transactions, untangling does not offer any extra information. On the other hand, *separable* transactions can be divided into subtransactions, allowing for further analysis. By applying CS and OTC heuristics to these subtransactions individually, we can obtain new data regarding user wallets.

However, when dealing with *ambiguous* transactions, where it is unclear how to analyze them, using the OTC heuristic may lead to misuse. This is because the inputs of different subtransactions within an *ambiguous* transaction are assumed to be owned by different users.

If the clustering goes first, it groups addresses by users and may effect on the untangling. Denote `Group` function as a generalization of the `Simplify` function, which groups inputs and outputs not by addresses, but by their owners given a clustering `user`. I.e., $\texttt{Group} : t \mapsto t'$, where

$$
\begin{aligned}
&t = (\mathcal{A}, \mathcal{B}, c),\ t' = (\mathcal{A}', \mathcal{B}', c'),\\
&\mathcal{A}' = \{(-\texttt{Bal}(t, U), U) |\\
&\qquad U \in \texttt{user}(t) \cap \texttt{Addr}(\mathcal{A}) \wedge \texttt{Bal}(t, U) < 0\},\\
&\mathcal{B}' = \{(\texttt{Bal}(t, U), U) |\\
&\qquad U \in \texttt{user}(t) \cap \texttt{Addr}(\mathcal{B}) \wedge \texttt{Bal}(t, U) > 0\},\\
&c' = c,
\end{aligned}
$$

and `Bal` is the balance function for users is

$$
\texttt{Bal}(t, U) \equiv \sum_{(b,C)\in\mathcal{B}\wedge\texttt{user}(C)=U} b - \sum_{(a,C)\in\mathcal{A}\wedge\texttt{user}(C)=U} a.
$$

The `Group` operation can only make transaction simpler.

**Proposition 1.** *Let*

$$\texttt{regular} \prec \texttt{simple} \prec \texttt{separable} \prec \texttt{ambiguous}$$

*be the complexity order of SSM with respect to untangling.* `Group` *given any* `user` *can not make a transaction more complex in terms of* $\prec$ *order.*

*Proof.* Each partition of $\texttt{Group}(t)$ is obtained by applying the `Group` function to the partition of $\texttt{Simplify}(t)$. In other words, if we group user identifiers into addresses in a way that ensures all addresses of a user are in the same connectable pair, we will obtain a partition of $\texttt{Simplify}(t)$. It is important to note that each minimal partition of $\texttt{Simplify}(t)$ can be mapped to a minimal (although not necessarily unique) partition of $\texttt{Group}(t)$ using the function `Group`. Therefore, the set of minimal partitions of $\texttt{Group}(t)$ is a subset of the minimal partitions of $\texttt{Simplify}(t)$ that are mapped by `Group`. Consequently, $\texttt{Group}(t)$ cannot have more untanglings than $\texttt{Simplify}(t)$. ■

Moreover,

**Proposition 2.** *All the simplification cases along*

$$\texttt{regular} \prec \texttt{simple} \prec \texttt{separable} \prec \texttt{ambiguous}$$

*are possible.*

*Proof.* Consider an example. Let $t = (\mathcal{A}, \mathcal{B}, 0)$, where $\mathcal{A} = \{(1, A_1), (1, A_2), (3, A_3)\}$ and $\mathcal{B} = \{(1, B_1), (1, B_2)(3, B_2)\}$. The transaction $t$ is *ambiguous* as both $A_1$ and $A_2$ connects to $B_1$. The $\texttt{user}_1$, which groups $A_1$ and $A_2$, makes the transaction $t$ *separable*. The $\texttt{user}_2$, which groups $B_1$ and $B_3$, makes the *separable* transaction *simple*. The $\texttt{user}_3$, which groups $B_1$, $B_2$ and $B_3$, makes the *simple* transaction *regular*.

Now, starting from either the initial transaction $t$ or one of the intermediate transactions, and applying either a trivial user grouping or a composition of a subset $\{\texttt{user}_1, \texttt{user}_2, \texttt{user}_3\}$, we can achieve any complexity of `Group` given any greater or equal complexity of `Simplify`. ■

These observations motivate us not to untangle after the clustering, but rather check if the clustering and untangling may contradict and compliment each other.

This preprocessing step `Simplify` is not used for heuristics. To utilize the untangling results by heuristics, we propose `Expand` function as an inverse for `Simplify` on subtransactions. `Expand` takes unsimplified transaction $t$ and its subtransaction $\tau'$ as inputs and outputs expanded subtransaction $\tau$. The subtransaction $\tau$ has the same input and output addresses as $\tau'$, and their entries are the same as in $t$ (see Figure 1). I.e., $\texttt{Expand}: (t, \tau') \mapsto \tau$, where

$$t = (\mathcal{A}, \mathcal{B}, c),\ \tau = (\mathcal{A}^\tau, \mathcal{B}^\tau, c^\tau),\ \tau = (\mathcal{A}^{\tau'}, \mathcal{B}^{\tau'}, c^{\tau'}),$$
$$\mathcal{A}^\tau = \{(a, A) \in \mathcal{A} | A \in \texttt{Addr}(\mathcal{A}^{\tau'}) \cup \texttt{Addr}(\mathcal{B}^{\tau'})\},$$
$$\mathcal{B}^\tau = \{(b, B) \in \mathcal{B} | B \in \texttt{Addr}(\mathcal{A}^{\tau'}) \cup \texttt{Addr}(\mathcal{B}^{\tau'})\},$$
$$c^\tau = c^{\tau'}.$$

## V. Numerical Experiments

We conducted an analysis using data from Bitcoin Core's implementation of node and a ground truth of OTC transactions from [6]. The OTC dataset covers transactions up to block $637,090$ (June 2020) and is referred to as OTC-GT. The code for the SSM untangling method can be found in [19], and the untangling results are summarized up to the first $747,936$ blocks (August 2022), although transaction-wise untangling data is not provided. To validate the results, we replicated the untangling process up to block $747,936$ (754 million transactions) and observed the consistency with the original paper's findings. We did not consider any further blocks beyond this point. We denoted the heuristics as CS, OTC-1, OTC-2, OTC-3, and OTC-GT. The workflow is as follows

1) Identify heuristic transactions among all available transactions.
2) Classify all historical transactions using the SSM algorithm within a fixed time limit.
3) Identify heuristic transactions within *separable* SSM.
4) Visualize the intersection of SSM and heuristics.

Once the paper is accepted, we will make the source code for our research openly available on GitHub.

### A. Heuristics

CS requires a single output and non-zero inputs (excluding coinbase transactions), whereas OTCs necessitate two outputs. As a result, CS and OTCs are mutually exclusive. CS does not demand any supplementary information and can be computed on a transaction-by-transaction basis with a single pass over the transaction log.

All OTC-1, OTC-2, and OTC-3 require checking if the potential change address is new. This can be done either in one transaction log run by saving all the first occurrences of the addresses or in two runs with less memory by finding change candidates in the first run and checking if they are first during the second run. For OTC-3, it is also necessary to check if a not-a-change address in a current transaction was not a change address previously. This check can be handled during either a single-run or a double-run processing. OTC-GT contains transaction hashes ordered historically and change UTXO indexes, making it easy to use and find intersections with a single bi-index run over the transaction log and the ground truth dataset.

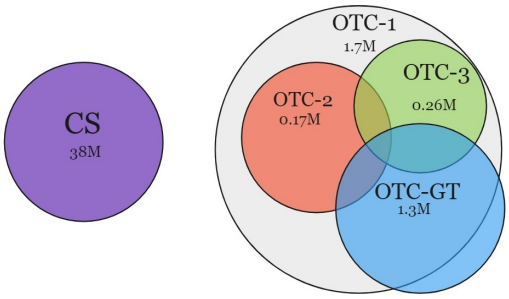


Fig. 9. The intersections between heuristics

The Euler diagram depicted in Figure 9 illustrates the relationships between different categories. CS stands out as the sole heuristic that deals exclusively with *regular* transactions and does not intersect with OTC. OTC-1, on the other hand, is the least restrictive category and encompasses the other OTCs.

### B. Untangling

SSM untangling is performed in parallel and on a transaction basis. Following the approach described in the original paper [19], we imposed a time limit of one second and achieved consistent results comparable to those presented in Section V-B . While we acknowledge that the specific transactions within the time limit may vary due to hardware differences, the overall proportions rounded to the tens of percents remain unchanged.

### C. Dependency between Heuristics and Untangling

CSs are considered *regular* from an SSM perspective, meaning there is no need to untangle any transactions. On the other hand, OTCs are inherently shared send mixers, which can lead to transactions that are *simple*, *separable*, or *ambiguous*. If an OTC transaction is *simple*, it does not contradict the heuristics. However, if it is *separable* or *ambiguous*, it may suggest misuse of the OTC. Nonetheless, the heuristics can still be applied to the individual parts of *separable* transactions. It is important to verify if the change address is new and if the non-change address was not previously a change address. Applying OTC to separated subtransactions can result in new change addresses, thereby compromising the original OTC transaction's property of being an OTC transaction. We have also taken into account such cases in our analysis.

The Sankey diagram illustrating the heuristics and untangling relations can be seen in Figure 10. It is evident that all of the OTC heuristics contradict the untangling process, with OTC-3's $70\%$ cases to OTC-GT's less than $30\%$. With the exception of OTC-GT, where we solely relies on the dataset without applying any algorithm, all of the heuristics

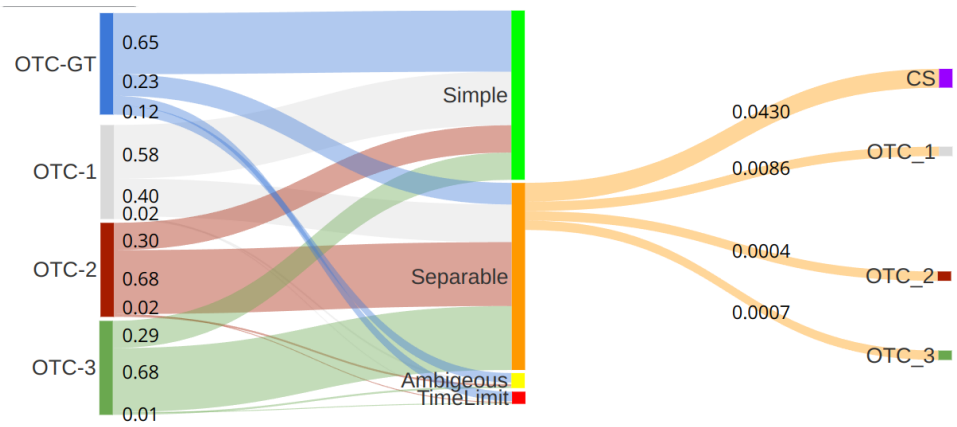


Fig. 10. The Sankey diagram of the heuristics and untangling relations. Left-to-right it demonstrates the fractions of heuristics transactions by SSM class, and the number of *separable* transaction resulted in heuristics subtransactions

are applicable to the *separable* transaction. `Separable` SSM produces $4\%$ of new CSs and tenths of percents of new OTCs.

## VI. Conclusions and Future Work

The analysis of Bitcoin anonymity is valuable for both cryptocurrency users seeking to understand their privacy limitations and for investigators developing analysis tools. Bitcoin address grouping by users is a crucial step in this analysis, although it is still under active research. Bitcoin usage pattern-driven heuristics are used for grouping. One-Time Change (OTC) is a commonly used heuristic, but it is controversial due to its tendency to produce errors. Different researchers often adjust OTC algorithms based on their own observations and experiences. In our study, we evaluated three OTC detection algorithms (OTC-1, OTC-2, OTC-3) and compared them to a dataset claimed to be the ground truth (OTC-GT). Another commonly used heuristic is Common Spending (CS), which occurs when a transaction has a single output. Although CS is less controversial than OTC, it still has the potential to lead to errors.

One technique for anonymizing Bitcoin transactions is the Shared Send Mixer (SSM), which combines multiple transactions from different users into one, creating a complex network of cryptocurrency flows. Mixers are related to grouping heuristics, particularly when they are *separable* and *ambiguous*. The first type allows a transaction to be split into multiple subtransactions in only one way, while the second type allows for two or more different splits.

The type of mixing transaction can be affected by address grouping. For example, if all the inputs and outputs are owned by a single user, there is nothing to untangle. In this paper, we have proven that the complexity of untangling transactions is ordered with respect to grouping (Propositions 1 and 2). In other words, after grouping, the complexity class of untangling a transaction cannot increase.

If a transaction is labeled as OTC and is either *separable* or *ambiguous*, it may indicate misuse of OTC. We identified from $35\%$ to $70\%$ such cases among OTC-1, OTC-2, OTC-3, and OTC-GT. Additionally, if a transaction is *separable*, heuristics can be applied to its subtransactions. We found millions of *separable* transactions split into two to eight subtransactions, resulting in thousands of new CS, OTC-1, OTC-2, and OTC-3 cases respectively. Our findings not only highlight the importance of grouping heuristics to address potential errors but also demonstrate the potential for uncovering new grouping data by applying heuristics to untangled subtransactions.

The Bitcoin untangling problem is known to be non-polynomial complete, making it difficult to classify some transactions within a time limit. While there is no universal untangling algorithm that can handle all possible transactions, it may be possible to automatically classify practical transactions with some algorithm modifications due to their simpler structure. With over 10 million timed-out transactions currently, untangling a significant portion of them could provide a better understanding of the Bitcoin flow and is the direction we are considering for future work.

Other blockchains, including Litecoin, Bitcoin Cash, and several Bitcoin forks, also utilize the UTXO transaction model. This means that the heuristics and untangling methodology mentioned earlier can be applied to these blockchains as well. However, some modifications are necessary for Cardano and Zcash. Cardano has an extended UTXO model that allows for complex smart contracts with tokens and other logic, in addition to native tokens. To gain further insights into user patterns on Cardano, additional information would be required. Zcash, on the other hand, supports both Bitcoin-like transactions and shielded zero-knowledge transactions. However, the time-consuming nature of shielded transactions has resulted in a poor user experience, leading to their low popularity. Considering both the untangling and heuristics techniques for Litecoin and Bitcoin Cash, as well as the challenging adaptation required for Cardano and Zcash, are areas that could be explored in future research.

## References


[1] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," *www.bitcoin.org*, pp. 1–9, 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[2] E. Androulaki, G. O. Karame, M. Roeschlin, T. Scherer, and S. Capkun, "Evaluating User Privacy in Bitcoin," in *Lecture Notes in Computer Science*. Heidelberg: Springer, 2013, vol. 7859, pp. 34–51.

[3] K. Atlas, "Weak Privacy Guarantees for SharedCoin Mixing Service," Tech. Rep., 2014. [Online]. Available: https://www.coinjoinsudoku.com/advisory/

[4] A. Wahrstatter, J. Gomes, S. Khan, and D. Svetinovic, "Improving Cryptocurrency Crime Detection: CoinJoin Community Detection Approach," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–11, 2023.

[5] Y. Yanovich, P. Mischenko, and A. Ostrovskiy, "Shared Send Untangling in Bitcoin," *bitfury.com*, vol. 2016, pp. 1–25, 2016. [Online]. Available: https://bitfury.com/content/downloads/bitfury_whitepaper_shared_send_untangling_in_bitcoin_8_24_2016.pdf

[6] M. Möser and A. Narayanan, "Resurrecting Address Clustering in Bitcoin," 2022, pp. 386–403.

[7] G. Andresen, "Pay to Script Hash," 2012. [Online]. Available: https://github.com/bitcoin/bips/blob/master/bip-0016.mediawiki

[8] E. Lombrozo, J. Lau, and P. Wuille, "Segregated Witness (Consensus layer)," 2015. [Online]. Available: https://github.com/bitcoin/bips/blob/master/bip-0141.mediawiki

[9] P. Wuille, "(bip-0032) Hierarchical Deterministic Wallets," 2012. [Online]. Available: https://github.com/bitcoin/bips/blob/master/bip-0032.mediawiki

[10] F. Reid and M. Harrigan, "An Analysis of Anonymity in the Bitcoin System," in *Security and Privacy in Social Networks*. New York, NY: Springer New York, 2013, pp. 197–223. [Online]. Available: https://link.springer.com/10.1007/978-1-4614-4139-7_10

[11] S. Meiklejohn, M. Pomarole, G. Jordan, K. Levchenko, D. McCoy, G. M. Voelker, and S. Savage, "A fistful of bitcoins: Characterizing payments among men with no names," *Proceedings of the ACM SIGCOMM Internet Measurement Conference, IMC*, pp. 127–139, 2013. [Online]. Available: https://dl.acm.org/doi/10.1145/2504730.2504747

[12] D. Ermilov, M. Panov, and Y. Yanovich, "Automatic Bitcoin Address Clustering," in *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 12 2017, pp. 461–466. [Online]. Available: http://ieeexplore.ieee.org/document/8260674/

[13] D. Y. Huang, M. M. Aliapoulios, V. G. Li, L. Invernizzi, E. Bursztein, K. McRoberts, J. Levin, K. Levchenko, A. C. Snoeren, and D. McCoy, "Tracking Ransomware End-to-end," in *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 5 2018, pp. 618–631.

[14] F. Liu, Z. Li, K. Jia, P. Xiang, A. Zhou, J. Qi, and Z. Li, "Bitcoin Address Clustering Based on Change Address Improvement," *IEEE Transactions on Computational Social Systems*, pp. 1–12, 2023.

[15] G. Maxwell, "CoinJoin: Bitcoin privacy for the real world," 2013. [Online]. Available: https://bitcointalk.org/?topic=279249

[16] European Union Agency for Law Enforcement Cooperation, "Internet organised crime threat assessment (IOCTA) 2020," Tech. Rep., 2020. [Online]. Available: https://www.europol.europa.eu/sites/default/files/documents/internet_organised_crime_threat_assessment_iocta_2020.pdf

[17] ——, "Internet Organised Crime Threat Assessment (IOCTA) 2021," Tech. Rep., 2021. [Online]. Available: https://www.europol.europa.eu/publications-events/main-reports/internet-organised-crime-threat-assessment-iocta-2021

[18] D. Eck, A. Torek, S. Cutchin, and G. G. Dagher, "Diffusion: Analysis of Many-to-Many Transactions in Bitcoin," in *2021 IEEE International Conference on Blockchain (Blockchain)*. IEEE, 12 2021, pp. 388–393. [Online]. Available: https://ieeexplore.ieee.org/document/9680541/

[19] N. Larionov and Y. Yanovich, "Bitcoin Shared Send Transactions Untangling in Numbers," *IEEE Access*, vol. 11, pp. 71 063–71 072, 2023.

[20] M. Möser and R. Böhme, "Join Me on a Market for Anonymity," *Workshop on the Economics of Information Security (WEIS)*, 2016.

[21] S. Goldfeder, H. Kalodner, D. Reisman, and A. Narayanan, "When the cookie meets the blockchain: Privacy risks of web payments via cryptocurrencies," *Proceedings on Privacy Enhancing Technologies*, vol. 2018, no. 4, 2018.

[22] H. Kalodner, M. Möser, K. Lee, S. Goldfeder, M. Plattner, A. Chator, and A. Narayanan, "BlockSci: Design and applications of a blockchain analysis platform," in *29th USENIX Security Symposium (USENIX Security 20)*. USENIX Association, 2020, pp. 2721–2738. [Online]. Available: https://www.usenix.org/conference/usenixsecurity20/presentation/kalodner