Analysis of CryptoNote Transaction Graphs using the Dulmage-Mendelsohn Decomposition

Abstract. Transactions in CryptoNote blockchains use linkable ring signatures to prevent double spending. Each transaction ring is associated with a key image, which is a collision-resistant one-way function of the spent output's secret key. Several techniques have been proposed to trace CryptoNote transactions, i.e. identify the actual output associated with a key image, by using the transaction history. In this paper, we show that the Dulmage-Mendelsohn (DM) decomposition of bipartite graphs can be used to trace CryptoNote transactions. The DM decomposition technique is optimal in the sense that it eliminates every output-key image association which is incompatible with the transaction history.
We used the Monero transaction history for performance comparison. For pre-RingCT outputs in Monero, the DM decomposition technique performs better than existing techniques. For RingCT outputs in Monero, the DM decomposition technique has the same performance as existing techniques, with only five out of approximately 29 million outputs being identified as spent. To study the effect of hard forks on Monero RingCT output traceability, we used information from four Monero hard forks and found that the DM decomposition technique is able to trace only 62,809 out of approximately 26 million RingCT transaction rings. Our results are further evidence supporting the claim that Monero RingCT transactions are mostly immune to traceability attacks.

Keywords: Cryptocurrency · CryptoNote · Monero · Traceability.

1 Introduction

Coins in CryptoNote blockchains are associated with stealth addresses, which are also called one-time addresses or transaction outputs [15]. We will use the term output for brevity. Each output is uniquely identified by a public key, which is a point on an elliptic curve. To spend from an output, the spender needs to know the corresponding secret key. In a transaction, the spender creates a *ring* of outputs which is a set containing the output being spent and some other outputs sampled from the CryptoNote blockchain (these are called decoy outputs or *mixins*). The spender generates a linkable ring signature over the ring of outputs using the secret key of the output being spent. This signature only reveals that the signer knows the secret key corresponding to one of the ring outputs, without revealing the identity of the actual output being spent. To prevent double spending from an output, the linkable ring signature reveals the *key image* of the output being spent. The key image of an output is a collision-resistant one-way function of the secret key. For example, in Monero the public

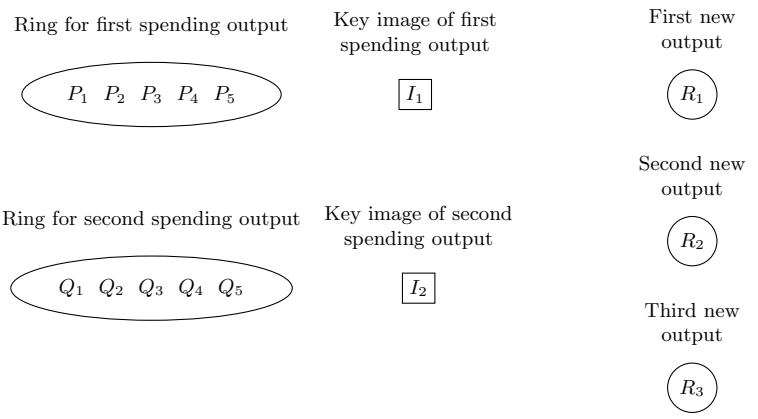

Fig. 1. A CryptoNote transaction with two inputs and three outputs

key associated with an output is given by $P = xG$ where G is the base point of an elliptic curve group and x is the secret key. Let $H_{\mathrm{p}}(\cdot)$ denote the Keccak hash function, whose outputs can be interpreted as points on the elliptic curve. The key image I of the output associated with P is given by $xH_{\mathrm{p}}(P)$. If the owner of the output corresponding to P tries to spend the coins associated with it more than once, then the key image I would appear again in the second transaction, identifying it as a double spending transaction. Such transactions are not included in blocks by miners as the resulting block would be considered invalid by the network.

Consider a CryptoNote transaction which spends from two existing outputs and creates three new outputs as illustrated in Fig. 1. The new outputs are denoted by R_1, R_2, R_3. The transaction has two rings of outputs of size five each, $(P_1, P_2, \ldots, P_5)$ and $(Q_1, Q_2, \ldots, Q_5)$. Exactly one output from each ring is being spent in the transaction. The key images I_1 and I_2 of the outputs being spent are revealed in the transaction. Note that the two rings can have common outputs.

For the purpose of illustration, suppose that the two rings have two outputs in common. Let $Q_1 = P_4$ and $Q_2 = P_5$. The relationship between the ring outputs and the key images in this transaction can be represented by the bipartite graph in Fig. 2. The union of the two ring output sets forms one vertex class and the two key images form the other vertex class. An edge between an output and a key image indicates that the latter could be the true key image of that output. Note that the new outputs R_1, R_2, R_3 play no role in the construction of the bipartite graph. We will refer to such output/key image bipartite graphs as *transaction graphs*.

As each key image must have been generated from a unique output, any pair of edges (P_i, I_1) and (Q_j, I_2) such that $P_i \neq Q_j$ is a plausible candidate for the true relationship between the outputs and key images. Recall that a matching

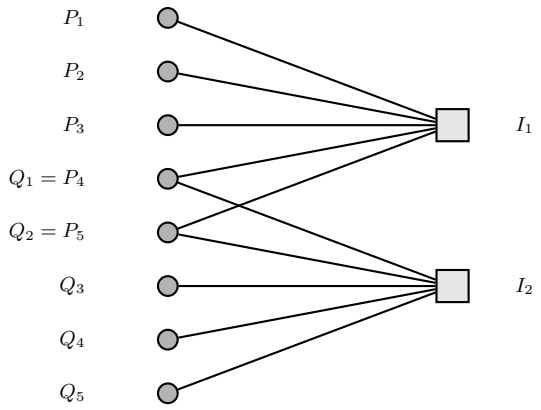

Fig. 2. Transaction graph corresponding to the transaction in Fig. 1

on a graph is a subset of the edges such that no two edges in the subset share a vertex. The pair of edges (P_i, I_1) and (Q_j, I_2) with $P_i \neq Q_j$ is a matching on the graph in Fig. 2. In fact, it is a matching of maximum size as any three edges in this graph would have two which meet in either I_1 or I_2.

Let us now consider a similar bipartite graph induced by the set of all transactions which have appeared up to the block having height h. The key image vertex class $\mathcal{K}_h$ in this graph is the set of all key images which have appeared on the blockchain up to block height h. The output vertex class $\mathcal{O}_h$ is the set of all outputs which have *appeared in at least one transaction ring* in the blocks up to height h. Note that $\mathcal{O}_h$ is *not* the set of all outputs which have appeared on the blockchain in blocks up to height h. We represent the edge set of the transaction graph induced by the CryptoNote transaction rings as a subset E of $\mathcal{O}_h \times \mathcal{K}_h$. For $P \in \mathcal{O}_h$ and $I \in \mathcal{K}_h$, the edge (P, I) belongs to E if the output P appeared in the transaction ring used to create I (via the linkable ring signature).

Since each key image $I \in \mathcal{K}_h$ is generated from a unique output $P \in \mathcal{O}_h$, we have $|\mathcal{K}_h| \leq |\mathcal{O}_h|$. In a bipartite graph with vertex classes of cardinality m and n, the size of a maximum matching can be at most $\min(m, n)$. Since the edges corresponding to the true association between outputs and key images form a matching of size $|\mathcal{K}_h|$, the induced bipartite graph always has a maximum matching. In fact, we have the following principle which has been discussed by Monero Research Lab researchers [9] and others [18].

Any maximum matching on a CryptoNote transaction graph is a plausible candidate for the ground truth, i.e. the true association between outputs and key images.

The Dulmage-Mendelsohn (DM) decomposition of a bipartite graph [8] reveals constraints which are satisfied by every maximum matching on the graph. It identifies vertices which are matched in every maximum matching. It classifies edges as *admissible* or *inadmissible*. An edge is called admissible if it appears in

at least one maximum matching and is called inadmissible if it does not appear in any maximum matching.

The DM decomposition of a CryptoNote transaction graph reveals the true output being spent in a transaction (making the ring traceable) if the edges from all other outputs to the key image are inadmissible. It also reveals the set of outputs which can be marked as spent (precisely those outputs which are matched by every maximum matching). The DM decomposition is optimal in the sense it identifies all inadmissible edges, i.e. edges between outputs and key images that are incompatible with the transaction history. In comparison, previously proposed methods are based on heuristics which fail to identify some inadmissible edges.

1.1 Related Work

The first traceability analyses on CryptoNote blockchains were performed by Kumar et al. [11] and Möser et al. [12]. They both studied the Monero blockchain history and found that zero-mixin transactions have a cascade effect of rendering other transactions traceable. They also considered heuristics for tracing transactions like the guess-newest heuristic and the output merging heuristic. But these methods are not conclusive and can lead to false positives. The *cascade attack* proceeds in an iterative manner. First it marks the outputs in zero-mixin transactions as spent. Then it marks these outputs as mixins in other (non-zero-mixin) transactions. If all but one output in a transaction ring are marked as mixins, then the remaining output is marked as spent (and the transaction becomes traceable). The outputs which have been newly marked as spent in a ring are marked as mixins in other rings. The process continues until no new outputs can be marked as spent. Kumar et al. reported results for 5 iterations of this algorithm for Monero history up to block 1,240,503. Recent work by Ye et al. [17] reported that the algorithm required 27 iterations to converge for Monero history up to block 2,077,094. While the cascade attack was able to trace a significant percentage of non-RingCT transactions, RingCT transactions were immune to it (Ye et al. reported that the algorithm could not trace any RingCT transactions). This was primarily because RingCT transactions did not allow zero-mixin rings.

Wijaya et al. [16] observed that a zero-mixin effect could be created in RingCT transactions by spending n times from a ring of size n. The n outputs in the ring can then be marked as mixins in other transaction rings. As a proof of concept, they created five outputs in Monero block 1,468,425 and then spent all of them using the other four as mixins in five transaction rings in block 1,468,439. This behavior does not arise naturally due to the mixin sampling strategy in Monero. Up to Monero block 2,330,000 (April 1, 2021), the ring of size 5 created by Wijaya et al. is the only RingCT ring which exhibits this behavior.

Yu et al. [19] defined a *closed set* to be a set of n outputs which can be represented as a union of n transaction rings. As each transaction ring must spend a unique output, the outputs in a closed set can be marked spent. They

proposed an algorithm for tracing CryptoNote transactions by identifying all closed sets. Their algorithm first performs the cascade attack [11,12]. Then it uses an approximate algorithm to identify closed sets called the *clustering algorithm*. An approximate algorithm is used due to the impractical complexity of the exhaustive search algorithm. In the Monero blockchain up to block 1,541,236, the clustering algorithm was able to identify 3,017 closed sets in the size range 2 to 55. While the performance of their algorithm is better than the cascade attack, they reported that no RingCT transactions were traced by their algorithm.

Several projects have forked the Monero blockchain resulting in multiple blockchains with large numbers of common outputs. When a common output is spent in two different forks, the same key image appears in both spending transactions. The real output is then contained in the intersection of the transaction rings of such transactions. Hinteregger et al. [10] used repeated key images which appeared in Monero and two hard forks, Monero Original [4] and MoneroV [6], to trace transactions in all three chains. While their method was able to trace transactions in Monero which were not traced by previous methods, they reported that the overall impact of their technique was small. This was attributed to the low adoption of the Monero hard forks.

The Monero reference implementation includes a tool for identifying spent outputs using the techniques described above [3]. It implements the cascade attack, finds transactions which cause the zero-mixin effect as characterized by Wijaya et al. [16], attempts to identify closed sets, and performs the cross-chain analysis proposed by Hinteregger et al. [10]. It is included in every Monero release as an executable with the name `monero-blockchain-mark-spent-outputs`. It is informally called the "blackball tool" in the Monero community as the set of spent outputs represent a blacklist which should be avoided when sampling mixins.

1.2 Our Contributions

Our contributions are as follows.

1. We show that the Dulmage-Mendelsohn (DM) decomposition of bipartite graphs can be used to trace transactions in CryptoNote blockchains. By classifying each edge in a CryptoNote transaction graph as admissible or inadmissible, the DM decomposition technique gives the best performance one can expect from any method which does not produce false positives.
2. We compute the empirical performance of the DM decomposition technique on Monero and show that it outperforms the method proposed by Yu et al. [19]. The latter results are the best results on Monero traceability which do not use information from hard forks.
3. While previous traceability attacks have been effective against non-RingCT transactions in Monero, RingCT transactions have been mostly immune. Only cross-chain analysis which uses information from hard forks has been able to trace Monero RingCT transactions [10]. To check the immunity of Monero RingCT transactions against the DM decomposition technique

which incorporates hard fork information, we constructed a transaction graph using four different hard forks: Monero Original, MoneroV, Monero v7, and Monero v9. We computed the performance of the DM decomposition technique on this graph up to Monero block height 2,330,000 (April 1, 2021). We found that 62,809 RingCT transaction rings out of 26,098,794 are traceable, i.e. only 0.24% of the RingCT rings are traceable. This result is further evidence supporting the claim that Monero RingCT transactions are mostly immune to traceability attacks.

2 The Dulmage-Mendelsohn Decomposition

Consistent with notation used by Dulmage and Mendelsohn [8], we define an undirected bipartite graph K as a triple (S, T, E) where S and T are non-empty sets representing vertex classes and $E \subseteq S \times T$ represents the edge set. So an edge in K is given by an ordered pair (s, t) where $s \in S$ and $t \in T$. The ordering of the vertices in the edge (s, t) is simply a consequence of putting S before T in the triple (S, T, E), and does not imply directivity. We say that an edge (s, t) belongs to the graph K, written as $(s, t) \in K$, to mean that $(s, t) \in E$. We only consider bipartite graphs K where both S and T are finite sets.

Definition 1. *Let $K = (S, T, E)$ be a bipartite graph. Let A and B be subsets of S and T respectively. A pair of such sets (A, B) is called a **vertex cover** for a bipartite graph K if for each edge $(s, t) \in K$, either $s \in A$ or $t \in B$ (both conditions can also hold).*

We state and prove a simple lemma for later reference.

Lemma 1. *Suppose (A, B) is a vertex cover of a bipartite graph $K = (S, T, E)$. Then $E \cap (A^c \times B^c) = \emptyset$.*

Proof. We want to argue that the graph cannot have edges in the set $A^c \times B^c$. Suppose that $(s, t) \in E \cap (A^c \times B^c)$. Then $s \in A^c$ and $t \in B^c$. This contradicts the assumption that (A, B) is a vertex cover of K. □

Definition 2. *The **size** of a vertex cover (A, B) is defined as $|A| + |B|$ where $|X|$ denotes the cardinality of a set X.*

Since S and T are assumed to be finite sets, every vertex cover of K will have a finite size.

Definition 3. *The **cover number** of a bipartite graph K is the minimum of $|A| + |B|$ over all vertex covers (A, B) of K.*

Definition 4. *A vertex cover (A, B) of a bipartite graph K whose size equals the cover number of K is called a **minimum cover**.*

We now define matchings on bipartite graphs and relate them to vertex covers. We say that edges (s, t) and (s', t') share a vertex if either $s = s'$ or $t = t'$.

Definition 5. *A **matching** on a bipartite graph $K = (S, T, E)$ is a subset M of the edge set E such that no two edges in M share a vertex. The cardinality $|M|$ is called the **order** of the matching M.*

Definition 6. *A **maximum matching** on a bipartite graph K is a matching on K of maximum order.*

The following theorem by König relates cover numbers to orders of maximum matchings.

Theorem 1. *The cover number of a finite bipartite graph equals the order of maximum matchings on the graph.*

The following definition classifies edges according to their membership in maximum matchings on K.

Definition 7. *An edge (s, t) of a bipartite graph K is said to be **admissible** if there exists a maximum matching M on K such that $(s, t) \in M$. An edge which is not admissible is said to be **inadmissible**.*

In a CryptoNote transaction graph, if we can show that an edge (P, I) is inadmissible, then P cannot be the true output corresponding to the key image I. This fact reduces the effective ring size of the transaction which created I. If we can classify all the edges incident on I except one as inadmissible, then the true output corresponding to I is identified and the transaction ring which created I becomes traceable. We now state a central theorem (proved in [8]) which characterizes inadmissible edges in terms of minimum covers.

Theorem 2. *An edge (s, t) of a bipartite graph K is inadmissible if and only if there exists a minimum cover (A, B) of K such that (s, t) belongs to $A \times B$.*

We will need the following corollary of Theorem 2 in a later argument.

Corollary 1. *Let K be a bipartite graph with finite cover number and let (A, B) be a minimum cover of K. Every maximum matching M on K has $|A|$ edges in $A \times B^c$ and $|B|$ edges in $A^c \times B$.*

Proof. See Appendix A. □

Theorem 2 is also related to the definition of sets of spent outputs given by Monero Research Lab [13]. See Appendix B for the relevant discussion.

The following two theorems were proved by Dulmage and Mendelsohn [8].

Theorem 3. *If (A_1, B_1) and (A_2, B_2) are minimum covers of a bipartite graph K having finite cover number, then $(A_1 \cap A_2, B_1 \cup B_2)$ and $(A_1 \cup A_2, B_1 \cap B_2)$ are both minimum covers of K.*

Theorem 4. *Let (A_1, B_1) and (A_2, B_2) be minimum covers of a bipartite graph K having finite cover number. If $A_1 \subseteq A_2$, then $B_1 \supseteq B_2$.*

Setting $A_1 = A_2$ in the above theorem gives us the following corollary.

Corollary 2. *If (A, B_1) and (A, B_2) are both minimum covers of a bipartite graph K having finite cover number, then $B_1 = B_2$.*

The following theorem will be useful in identifying spent outputs in CryptoNote transaction graphs. We give a proof as it was not explicitly stated by Dulmage and Mendelsohn [8], although it follows from their results.

Theorem 5. *Let (A_1, B_1) and (A_2, B_2) be minimum covers of a bipartite graph K having finite cover number such that $A_1 \subseteq A_2$. Then every maximum matching M on K has $|A_2| - |A_1|$ edges in the set $(A_2 \setminus A_1) \times (B_1 \setminus B_2)$.*

Proof. See Appendix C. □

If a matching on a graph has an edge incident on a vertex, we say that the vertex is *matched* by the matching. The following corollary of Theorem 5 says that all the vertices in the difference between two minimum covers are matched by every maximum matching.

Corollary 3. *Let (A_1, B_1) and (A_2, B_2) be minimum covers of a bipartite graph K having finite cover number such that $A_1 \subseteq A_2$. Let M be any maximum matching on K. Then all the vertices in the sets $A_2 \setminus A_1$ and $B_1 \setminus B_2$ are matched by M.*

Proof. See Appendix D. □

By this corollary, if we can find two distinct minimum covers of the transaction graph induced by a CryptoNote blockchain, then we would have identified some outputs which are matched by *every* maximum matching on this graph. Thus every candidate for the true association between outputs and key images has these outputs marked as spent.

For a bipartite graph K, let $\mathcal{C}$ be the set of all minimum covers. Let us define the following sets obtained by taking intersections and unions of the components of the minimum covers.

$$A_* = \bigcap_{(A,B)\in\mathcal{C}} A, \qquad A^* = \bigcup_{(A,B)\in\mathcal{C}} A, \tag{1}$$

$$B_* = \bigcap_{(A,B)\in\mathcal{C}} B, \qquad B^* = \bigcup_{(A,B)\in\mathcal{C}} B. \tag{2}$$

By Theorem 3, if K has a finite cover number then the pairs (A_*, B^*) and (A^*, B_*) are both minimum covers of K.

With the above definitions in place, we are ready to describe the DM decomposition.

Definition 8. *Let $K = (S, T, E)$ be a bipartite graph having a finite cover number. The **Dulmage-Mendelsohn decomposition** of K is a partition of $S \times T$ into three disjoint sets R_1, R_2, R_3 which satisfy the following properties:*

1. *The set of admissible edges in K equals $E \cap R_1$.*

2. *The set of inadmissible edges in K equals $E \cap R_2$.*
3. *$E \cap R_3 = \emptyset$.*

The structure of the sets R_1, R_2, R_3 depends on the minimum covers of K. Let us consider two cases.

Case 1: $A_* = A^*$. If $A_* = A^*$, then the graph K has only one minimum cover given by $(A_*, B^*) = (A^*, B_*)$. In this case, the following theorem holds.

Theorem 6. *Let $K = (S, T, E)$ be a bipartite graph having a finite cover number. If K has only one minimum cover given by (A_*, B^*), then the Dulmage-Mendelsohn decomposition of K is given by the partition of $S \times T$ into the sets R_1, R_2, R_3 given by*

$$\begin{aligned} R_1 &= (A_* \times (B^*)^c) \bigcup ((A_*)^c \times B^*), \\ R_2 &= A_* \times B^*, \\ R_3 &= (A_*)^c \times (B^*)^c. \end{aligned} \tag{3}$$

Proof. It is clear that $S \times T = R_1 \cup R_2 \cup R_3$ and $R_i \cap R_j = \emptyset$ for $i \neq j, 1 \leq i, j \leq 3$. Since (A_*, B^*) is a vertex cover, Lemma 1 tells us that $E \cap R_3 = \emptyset$. As (A_*, B^*) is the only possible minimum cover of K, Theorem 2 tells us that the set of inadmissible edges in K equals $E \cap R_2$. Furthermore, the theorem tells us that the edges in $E \cap R_2^c$ are admissible. Since there are no edges in $E \cap R_3$, the set of admissible edges in K equals $E \cap R_1$. □

Case 2: $A_* \neq A^*$. Now suppose $A_* \neq A^*$. By definition, $A_* \subseteq A^*$. So A_* must be a proper subset of A^*. Then there exists at least one non-empty set $X \subset S$ such that $A_* \cap X = \emptyset$ and $(A_* \cup X, Y)$ is a minimum cover of K for some $Y \subset T$. The existence of such a set follows from the fact $A^* \setminus A_*$ is a candidate for X. Let S_1 be a set of smallest cardinality among all candidates for X. There may be many possibilities for S_1, all having the same smallest cardinality. We can pick any one of them.

Let (A_1, B_1) be a minimum cover with $A_1 = A_* \cup S_1$. By Corollary 2, B_1 is uniquely determined by A_1. As $A_* \subseteq A_1$, Theorem 4 tells us that $B_1 \subseteq B^*$. As all minimum covers of K have the same size, we have $|A_*| + |B^*| = |A_1| + |B_1|$. Since $|A_1| > |A_*|$, we have $|B_1| < |B^*|$. Thus B_1 is a proper subset of B^*. Let $T_1 = B^* \setminus B_1$. Since $|A_1| - |A_*| = |B^*| - |B_1|$, we have $|S_1| = |T_1|$.

If $A_1 = A^*$, the process stops. Otherwise, there exists at least one non-empty set $X \subset S$ such that $A_1 \cap X = \emptyset$ and $A_1 \cup X$ is the first component of a minimum cover of K. Let S_2 be a set of smallest cardinality among all candidates for X. Let (A_2, B_2) be a minimum cover with $A_2 = A_1 \cup S_2 = A_* \cup S_1 \cup S_2$. As before, B_2 is uniquely determined by A_2 and $B_2 \subset B_1$. Let $T_2 = B_1 \setminus B_2$. Since $|A_2| - |A_1| = |B_1| - |B_2|$, we have $|S_2| = |T_2|$. Since $B^* = T_1 \cup B_1$ and $T_2 = B_1 \setminus B_2$, we have $B^* = T_1 \cup T_2 \cup B_2$.

If we proceed in this manner, the process will stop for some k where

$$A_* \cup S_1 \cup S_2 \ldots \cup S_k = A^*. \tag{4}$$

At this point, (A^*, B_*) will be the resulting minimum cover. Furthermore, the T_i's satisfy

$$B^* = T_1 \cup T_2 \cup \ldots T_k \cup B_*. \tag{5}$$

In the intermediate stages of this process, (A_i, B_i) is a minimum cover for K for each $i \in \{1, 2, \ldots, k\}$ where

$$A_i = A_* \cup S_1 \cup S_2 \cup \ldots \cup S_i, \tag{6}$$
$$B_i = T_{i+1} \cup T_{i+2} \cup \ldots \cup T_k \cup B_*. \tag{7}$$

Equations (4) and (5) give the following decompositions of the vertex classes S and T.

$$S = A^* \bigcup (A^*)^c = A_* \cup S_1 \cup S_2 \ldots \cup S_k \bigcup (A^*)^c, \tag{8}$$
$$T = (B^*)^c \bigcup B^* = (B^*)^c \bigcup T_1 \cup T_2 \ldots \cup T_k \cup B_*. \tag{9}$$

The $k+2$ sets in the unions on the extreme right of both the above equations form a partition of S and T respectively. These partitions are unique except for a permutation of the S_i's having same cardinality, with the T_i's appropriately permuted.

The DM decomposition is given by the following theorem. While a proof of this theorem can be found in [8], we give an outline of a proof in Appendix E.

Theorem 7. *Let $K = (S, T, E)$ be a bipartite graph having a finite cover number. Then the Dulmage-Mendelsohn decomposition of K is given by the partition of $S \times T$ into the sets R_1, R_2, R_3 given by*

$$R_1 = (A_* \times (B^*)^c) \bigcup (S_1 \times T_1) \bigcup \ldots \bigcup (S_k \times T_k) \bigcup ((A^*)^c \times B_*)\,, \tag{10}$$
$$R_2 = (A_* \times B^*) \bigcup (A^* \times B_*) \bigcup_{i<j} (S_i \times T_j)\,, \tag{11}$$
$$R_3 = ((A_*)^c \times (B^*)^c) \bigcup ((A^*)^c \times (B_*)^c) \bigcup_{i>j} (S_i \times T_j)\,. \tag{12}$$

Proof. See Appendix E. □

To visualize the DM decomposition, suppose that the vertices in S are ordered according to the partition in equation (8), i.e. the vertices in A_* appear first, followed by vertices in $S_1, S_2, \ldots, S_k$, and $(A^*)^c$. Similarly, suppose that the vertices in T are ordered according to the partition in equation (9). Then the DM decomposition can be represented by Fig. 3, where the rows correspond to

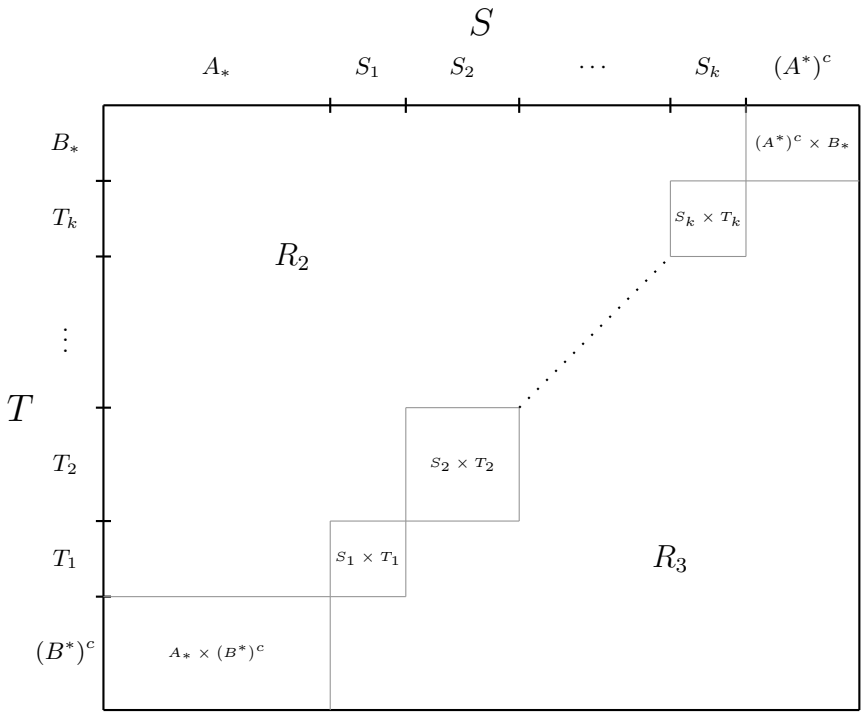

Fig. 3. The DM decomposition of a graph.

vertices in T and the columns correspond to vertices in S. The admissible edges lie in blocks along the diagonal, the inadmissible edges lie above these blocks, and there are no edges below these blocks.

Furthermore, by applying Theorem 5 to adjacent minimum covers in the sequence $(A_*, B^*), (A_1, B_1), \ldots, (A_{k-1}, B_{k-1}), (A^*, B_*)$, we conclude that every maximum matching on the graph has $|S_i|$ edges in $S_i \times T_i$ for $i = 1, 2, \ldots, k$. By Corollary 3, every maximum matching matches all the vertices in S_i and T_i. *If we choose S to be the set of all outputs in the bipartite graph induced by a CryptoNote blockchain transaction history, then the sets S_i contain only spent outputs for all $i = 1, 2, \ldots, k$.* These S_i's correspond to the closed sets defined by Yu et al. [19].

Computing the DM Decomposition

The DM decomposition of a bipartite graph K can be computed by finding a maximum matching M on K, then finding subsets of vertex classes unreachable from M via alternating paths, and finally by finding strongly connected components of the subgraph induced by the unreachable vertices (see [14] for

details). Both open source [7] and proprietary [1] implementations of the DM decomposition algorithm are available.

3 DM Decomposition of the Monero Transaction Graph

To evaluate the effectiveness of the DM decomposition in tracing transaction rings, we used the results obtained by Yu et al. [19] on Monero as the benchmark. The latter results are the best results on Monero traceability which do not use information from hard forks. As the algorithm proposed by Yu et al. first applies the cascade attack followed by a clustering algorithm to find closed sets, we henceforth refer to their algorithm as the cascade/clustering (CC) algorithm.

Yu et al. considered Monero transactions contained in blocks with height up to 1,541,236 (March 30, 2018). This data set contains 23,164,745 transaction rings (each one contributing a key image) and 25,126,033 outputs. The corresponding bipartite graph has 58,791,856 edges. In Monero, RingCT outputs have amounts hidden in Pedersen commitments. They were introduced in Monero in January 2017 and became mandatory in September 2017 [5]. Out of the 23,164,745 transaction rings in the data set, 4,330,234 were RingCT rings and the remaining 18,834,511 were pre-RingCT rings.

Previous work [11], [12], [19], on Monero traceability has shown that RingCT transactions in Monero are immune to traceability attacks. The same observation holds for the DM decomposition approach. None of the 4,330,234 RingCT rings could be traced by the DM decomposition. Table 1 compares the number of pre-RingCT transaction rings traced by the CC algorithm and the DM decomposition. Each row in the table gives results for transaction rings which have a certain number of mixin outputs. The results for all transaction rings with 10 or more mixin outputs are combined in the row with label "≥ 10".

All the 16,335,308 rings traced by the DM decomposition are associated with a set S_i with $|S_i| = 1$. The singleton set T_i corresponding to S_i has the key image of the output in S_i. As seen from the last row, the DM decomposition identifies only 341 more traceable rings than the CC algorithm. These new rings are only among the transaction rings having 2, 3, or 4 mixins. Thus, for transactions up to block height 1,541,236 the advantage of using the DM decomposition for tracing Monero transactions is marginal.

Yu et al. report finding 3017 closed sets with sizes in the range 2 to 55. In the DM decomposition, each S_i is a closed set. The DM decomposition is able to find 3045 closed sets with 3041 of them having sizes in the range 2 to 55. The remaining four closed sets have sizes 103, 106, 119, and 122. This discrepancy is due to the approximate nature of the clustering algorithm used by Yu et al. to find closed sets.

The DM decomposition marked 15,633,140 out of the 58,791,856 edges in the bipartite graph as inadmissible. Each inadmissible edge reduces the effective mixin size of a transaction ring. Table 2 gives the counts of transaction rings with a certain number of mixins before and after the DM decomposition. Yu et al. presented the corresponding counts after execution of the CC algorithm in

No. of mixins	No. of pre-RingCT rings	Traced by CC	Traced by DM
0	12,209,675	12,209,675	12,209,675
1	707,786	625,641	625,641
2	2,941,525	1,779,134	1,779,446
3	1,345,574	952,855	952,862
4	972,457	451,959	451,981
5	143,793	74,186	74,186
6	366,894	202,360	202,360
7	12,361	4,296	4,296
8	9,148	3,506	3,506
9	6,396	2,178	2,178
≥ 10	118,902	29,177	29,177
Total	18,834,511	16,334,967	16,335,308

Table 1. Monero traceability of pre-RingCT rings by the CC algorithm vs DM decomposition (up to block 1,541,236)

bar graph form. So we are unable to compare the exact numbers. As expected, transaction rings with smaller effective mixin size are more frequent after the DM decomposition.

To check if the transactions which have appeared after block 1,541,236 have affected the traceability of RingCT rings, we computed the DM decomposition of the subgraph induced exclusively by RingCT transaction rings in all blocks up to height 2,330,000 (April 1, 2021). This subgraph has 26,098,794 key images and 29,588,617 outputs with 252,843,948 edges between them. Let $\mathcal{K}$ be the set

Effective no. of mixins	No. of rings before DMD	No. of rings after DMD
0	12,209,675	16,335,308
1	707,786	1,413,028
2	4,496,490	2,369,796
3	1,486,593	279,377
4	3,242,625	2,369,578
5	319,352	186,257
6	432,875	73,690
7	21,528	13,086
8	30,067	23,615
9	17,724	13,071
≥ 10	200,030	87,939
Total	23,164,745	23,164,745

Table 2. Effective number of mixins before and after DM decomposition (up to block 1,541,236) without using fork data. Only 17 RingCT rings experience a change in effective number of mixins.

Fork Name	Fork block	Number of blocks in fork	Number of common RingCT keyimages
Monero Original	1,546,000	238,682	63,885
MoneroV	1,564,966	146,325	6,595
Monero v7	1,685,555	29	1,027
Monero v9	1,788,000	73	1,581

Table 3. Information about the four Monero hard forks

of all the key images in this subgraph. Its DM decomposition revealed only two minimum covers, $(\emptyset, \mathcal{K})$ and $(S_1, \mathcal{K} \setminus T_1)$ where $|S_1| = |T_1| = 5$. The set S_1 consists of RingCT outputs with indices 3890287, 3890288, 3890289, 3890290, and 3890291. These five outputs were created by Wijaya et al. [16] in block 1,468,425. All of them were spent using the other four as mixins in five transaction rings in block 1,468,439 (Dec 17, 2017), to demonstrate that a set of outputs can be considered spent without relying on zero-mixin transactions. These five outputs are also marked as spent by the Monero blackball tool [13]. Thus, the DM decomposition of the Monero RingCT subgraph (using only main chain data) does not identify any new outputs as spent.

There were 22,785,298 RingCT transaction rings in the blocks with heights from 1,468,426 to 2,330,000. The five spent RingCT outputs were chosen as mixins in only 17 of these RingCT rings. Each of the 17 rings has its effective number of mixins reduced by one. The latest affected ring appears in block 1,521,556 (March 3, 2018). Thus, the change in effective number of mixins shown in Table 2 is mostly in pre-RingCT rings.

To check the immunity of Monero RingCT transactions against the DM decomposition technique which incorporates hard fork information, we constructed a transaction graph using four different hard forks: Monero Original, MoneroV, Monero v7, and Monero v9.[1] Table 3 gives the information regarding these forks where the last column contains the number of RingCT keyimages which appeared both in the Monero main chain and the fork chain. We computed the performance of the DM decomposition technique on this graph up to Monero block height 2,330,000 (April 1, 2021). We found that 62,809 RingCT transaction rings out of 26,098,794 are traceable, i.e. only 0.24% of the RingCT rings are traceable. Note that the number of traceable RingCT rings is less than the total number of common keyimages shown in Table 3. This is because the appearance of key image in both the main chain and the fork chain does not imply traceability. If the transaction rings in both cases have more than one output in common, the true output being spent may not be identified.

Table 4 gives the counts of transaction rings with a certain number of mixins before and after the DM decomposition when data from hard forks is used. As

[1] As these forks are no longer operational, we used the blockchain databases made available by Justin Ehrenhofer [2]. We thank him for making these databases freely available.

Effective no. of mixins	No. of RingCT rings before DMD	No. of RingCT rings after DMD
0	0	62,809
1	0	5,220
2	1,554,965	1,556,082
3	141,019	213,028
4	2,313,491	2,251,916
5	179,063	241,408
6	1,619,076	1,515,709
7	295,199	285,560
8	55,123	52,661
9	16,435	56,514
≥ 10	19,924,423	19,857,887
Total	26,098,794	26,098,794

Table 4. Effective number of mixins in RingCT transaction rings before and after DM decomposition (up to block 2,330,000) while using fork data.

RingCT transactions were introduced with a minimum ring size of 3, there are no transaction rings with 0 or 1 mixins before the DM decomposition is applied. As before, transaction rings with smaller effective mixin size are more frequent after the DM decomposition.

4 Conclusion

We have described how the Dulmage-Mendelsohn decomposition of bipartite graphs can be used to characterize the information revealed by CryptoNote transaction rings. While the decomposition does not reveal much more about Monero than what was known before, it is preferable as it avoids the heuristics and computational bottlenecks of previous methods.

A natural question arises: *How should the mixin sampling strategy in CryptoNote blockchains be designed to avoid revealing information via the DM decomposition?* We do not have an answer. Empirically, the existing sampling strategy in Monero seems to be robust to the decomposition. Can one expect this robustness to continue in the future? Yu et al. [19] gave estimates on the probability of existence of a closed set for a uniform sampling strategy when each ring has 3 mixins. Similar analyses with more realistic assumptions are needed to understand the information leakage risks of the sampling strategies used in practice.

A Proof of Corollary 1

By König's theorem (Theorem 1), the maximum matching M has $|A|+|B|$ edges. Lemma 1 tells us that M cannot have any edges in $A^c \times B^c$ and Theorem 2 tells us that M has no edges in $A \times B$. Thus all the edges of M must lie in either $A \times B^c$ or $A^c \times B$.

As distinct edges in the matching M cannot share a vertex, for any two distinct edges (s_1, t_1) and (s_2, t_2) of M in $A \times B^c$ we must have $s_1 \neq s_2$. Thus the number of edges of M in $A \times B^c$ is at most $|A|$. Similarly, the number of edges of M in $A^c \times B$ is at most $|B|$. Since M has exactly $|A|+|B|$ edges, the sets $A \times B^c$ and $A^c \times B$ must have exactly $|A|$ and $|B|$ edges of M, respectively. □

B Relationship between Theorem 2 and MRL-0007

Sets of spent outputs were defined in MRL-0007 [13] as given below. In this appendix, we show that a union $\cup_{i=1}^{n} R_i$ of transaction rings has size n if it is the first member of a minimum cover of the transaction graph. Furthermore, we show that every size n first member of a minimum cover is a union of n transaction rings.

Definition 9. *Let $\mathcal{O}$ be the set of outputs on a CryptoNote-style blockchain. Let $R_i \subset \mathcal{O}$ be a transaction ring of outputs for $i = 1, 2, \ldots, n$. One output in each transaction ring is spent resulting in a unique key image. We say that each R_i is spent if*

$$\left| \bigcup_{i=1}^{n} R_i \right| = n.$$

An output is spent if it is an element of a spent ring.

The reasoning behind this definition is as follows. Each ring R_i has a unique key image I_i associated with it. Since $\cup_{i=1}^{n} R_i$ has only n outputs, all of them must have been spent to create the n key images $I_1, I_2, \ldots, I_n$.

Let $\cup_{i=1}^{n} R_i = \{P_1, P_2, \ldots, P_n\}$. Suppose we draw the bipartite graph induced by the entire blockchain history while listing $P_1, \ldots, P_n$ and $I_1, \ldots, I_n$ before the other vertices on each side. Let $P_{n+1}, \ldots, P_M$ be the other outputs on the blockchain. Let $I_{n+1}, \ldots, I_N$ be the other key images on the blockchain where $N \leq M$. Fig. 4 illustrates the bipartite graph. Since each key image in $I_1, I_2, \ldots, I_N$ corresponds to a unique true output on the left hand side, there exists a maximum matching of order N on this graph. Then $(\emptyset, \{I_1, \ldots, I_N\})$ is a minimum cover of the graph.

Note that there cannot be any edges from the key images $I_1, \ldots, I_n$ to the outputs $P_{n+1}, P_{n+2}, \ldots, P_M$. To see this, suppose there is an edge from I_j to P_k for some $j \in \{1, 2, \ldots, n\}$ and $k \in \{n+1, \ldots, M\}$. Then P_k must belong to the ring R_j as it is the only ring which contributes edges incident on I_j. This would mean P_k belongs to $\cup_{i=1} R_i = \{P_1, P_2, \ldots, P_n\}$, which is a contradiction as $k \geq n+1$. So all the edges incident on $I_1, \ldots, I_n$ must have an output from $P_1, \ldots, P_n$ on the other end.

The above argument shows that $(\{P_1, \ldots, P_n\}, \{I_{n+1}, \ldots, I_N\})$ is a minimum cover of the graph. Thus the union $\cup_{i=1}^{n} R_i$ of the spent rings as defined in Definition 9 is the first member of a minimum cover of the transaction graph.

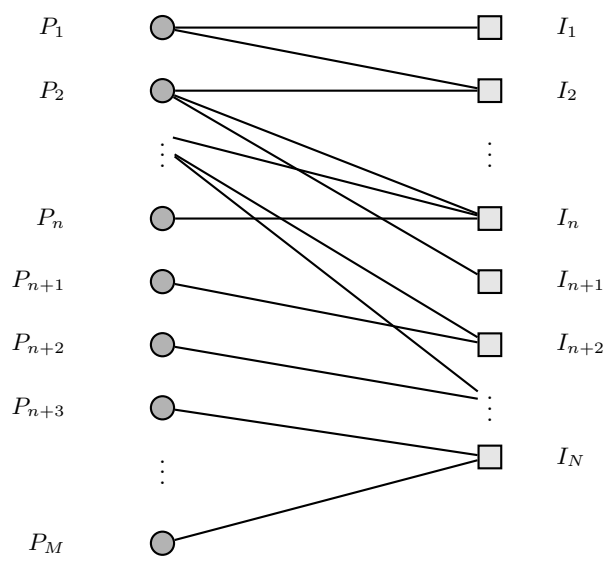

Fig. 4. Transaction graph to illustrate the connection of Definition 9 to Theorem 2.

One can also prove the other direction. We claim that if (A, B) is a minimum cover of the transaction graph where $A \neq \emptyset$, then there exist transaction rings $R_{i_1}, R_{i_2}, \ldots, R_{i_n}$ such that

$$A = \bigcup_{j=1}^{n} R_{i_j} \quad \text{and} \quad \left| \bigcup_{j=1}^{n} R_{i_j} \right| = n. \tag{13}$$

Let $\mathcal{O} = \{P_1, \ldots, P_M\}$ be the set of all outputs and $\mathcal{K} = \{I_1, \ldots, I_N\}$ be the set of all key images, which have appeared on the blockchain at some block height. For a minimum cover (A, B), let $B^c = \mathcal{K} \setminus B$ be the set of key images not in B. Suppose $B^c = \{I_{i_1}, I_{i_2}, \ldots, I_{i_n}\}$. Each key image I_{i_j} in B^c is associated with a unique transaction ring R_{i_j} which contains the true output corresponding to it.

Since $(\emptyset, \mathcal{K})$ is a minimum cover of the graph, every minimum cover must have size N. This implies that $|A| + |B| = N$. As $n = |B^c| = N - |B|$, the set A must have n outputs.

Since (A, B) is a cover of the bipartite graph, every edge incident on key images in B^c must be covered by an output in A (as B can only cover edges incident on the key images in it). The ring R_{i_j} associated with a key image $I_{i_j} \in B^c$ is the set of outputs adjacent to I_{i_j} in the graph. So the other endpoints of edges incident on I_{i_j} are in R_{i_j}. This implies that the transaction ring R_{i_j} is a subset of A for every $I_{i_j} \in B^c$. Thus $\cup_{j=1}^{n} R_{i_j} \subseteq A$.

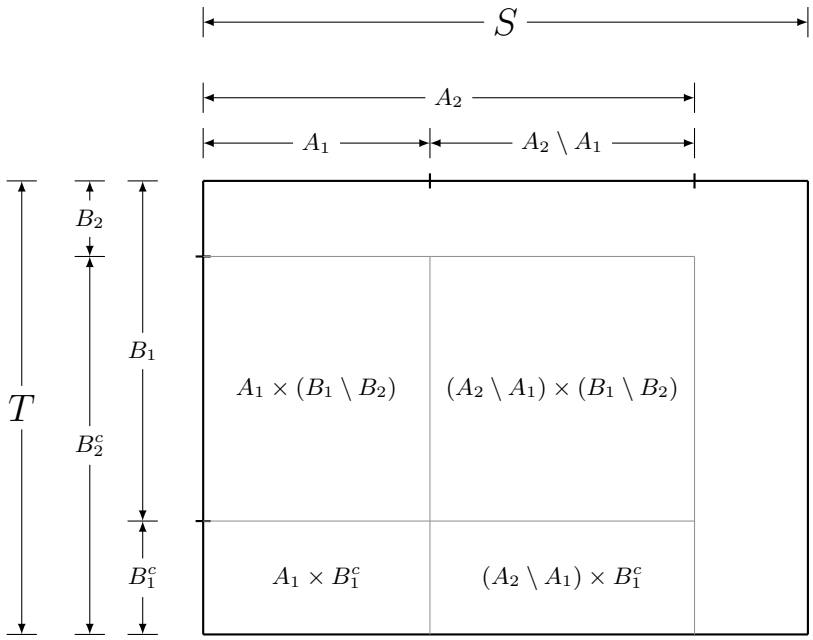

Fig. 5. Partition of $A_2 \times B_2^c$ in the proof of Theorem 5.

Furthermore, $\left|\cup_{j=1}^{n} R_{i_j}\right| \geq n$ because each of the n key images $I_{i_1}, I_{i_2}, \ldots, I_{i_n}$ has a unique true output in $\cup_{j=1}^{n} R_{i_j}$. Putting all this together, we have

$$n \leq \left|\bigcup_{j=1}^{n} R_{i_j}\right| \leq |A| = n. \tag{14}$$

Thus, we conclude that $A = \cup_{j=1}^{n} R_{i_j}$ and that $\left|\cup_{j=1}^{n} R_{i_j}\right| = n$.

C Proof of Theorem 5

Since $A_1 \subseteq A_2$, Theorem 4 tells us that $B_2 \subseteq B_1$. Since (A_1, B_1) and (A_2, B_2) are both minimum covers, Corollary 1 tells us that every maximum matching M of K has exactly $|A_1|$ edges in $A_1 \times B_1^c$ and exactly $|A_2|$ edges in $A_2 \times B_2^c$.

As $A_1 \times B_1^c \subseteq A_2 \times B_2^c$, every edge of M in $A_1 \times B_1^c$ is contained in $A_2 \times B_2^c$. Thus M has $|A_2| - |A_1|$ edges in $(A_2 \times B_2^c) \setminus (A_1 \times B_1^c)$. As illustrated in Fig. 5,

the set $A_2 \times B_2^c$ can be partitioned as

$$\begin{aligned} A_2 \times B_2^c &= [A_1 \cup (A_2 \setminus A_1)] \times [(B_1 \cup B_1^c) \cap B_2^c] \\ &= [A_1 \cup (A_2 \setminus A_1)] \times [(B_1 \setminus B_2) \cup B_1^c] \\ &= [A_1 \times (B_1 \setminus B_2)] \ \cup \ [A_1 \times B_1^c] \\ &\quad \cup \ [(A_2 \setminus A_1) \times (B_1 \setminus B_2)] \ \cup \ [(A_2 \setminus A_1) \times B_1^c]\,. \end{aligned} \tag{15}$$

Since (A_1, B_1) is a minimum cover and $A_1 \times (B_1 \setminus B_2) \subseteq A_1 \times B_1$, Theorem 2 tells us that the matching M cannot have any edges in $A_1 \times (B_1 \setminus B_2)$.

Since (A_1, B_1) is a vertex cover and $(A_2 \setminus A_1) \times B_1^c = (A_2 \cap A_1^c) \times B_1^c \subseteq A_1^c \times B_1^c$, Lemma 1 tells us that the graph K cannot have any edges in $(A_2 \setminus A_1) \times B_1^c$. Consequently, the matching M cannot have any edges in this set.

The above observations tell us that two of the partition elements in equation (15) cannot have edges from a maximum matching M. Thus the $|A_2| - |A_1|$ edges of M in $(A_2 \times B_2^c) \setminus (A_1 \times B_1^c)$ must belong to $(A_2 \setminus A_1) \times (B_1 \setminus B_2)$. □

D Proof of Corollary 3

Theorem 5 tells us that any maximum matching M must have $|A_2| - |A_1|$ edges in the set $(A_2 \setminus A_1) \times (B_1 \setminus B_2)$. Since $A_1 \subseteq A_2$, $|A_2 \setminus A_1| = |A_2| - |A_1|$. As any two distinct edges in M cannot have a vertex in common, each vertex in $A_2 \setminus A_1$ must have exactly one of the $|A_2| - |A_1|$ edges of M incident on it.

Since (A_1, B_1) and (A_2, B_2) are both minimum covers, $|A_1| + |B_1| = |A_2| + |B_2| \implies |B_1| - |B_2| = |A_2| - |A_1|$. As $B_1 \subseteq B_2$, $|B_1 \setminus B_2| = |B_1| - |B_2|$. Thus each vertex in $B_1 \setminus B_2$ has exactly one of the $|A_2| - |A_1|$ edges of M incident on it. □

E Proof Outline of Theorem 7

We need to show that the sets R_1, R_2, R_3 given in equations (10), (11), (12) satisfy the properties given in Definition 8. Let us first show that the graph cannot have edges in the set R_3. Since (A_*, B^*) and (A^*, B_*) are vertex covers, Lemma 1 tells us that the graph has no edges in $(A_*)^c \times (B^*)^c$ and $(A^*)^c \times (B_*)^c$. For $i \geq 2$, we have

$$A_{i-1}^c = S \setminus A_{i-1} = S_i \cup S_{i+1} \ldots \cup S_k \bigcup (A^*)^c\,, \tag{16}$$

$$B_{i-1}^c = T \setminus B_{i-1} = (B^*)^c \bigcup T_1 \cup T_2 \ldots \cup T_{i-1}, \tag{17}$$

as seen by the representations of A_i, B_i in equations (6), (7) and the representations of S, T in equations (8), (9). For $i > j$, each $S_i \times T_j$ is contained in (A_{i-1}^c, B_{i-1}^c). As (A_{i-1}, B_{i-1}) is vertex cover, by Lemma 1 the graph cannot have edges in $S_i \times T_j$ for $i > j$. This completes the proof that the edge set E of the graph K satisfies $E \cap R_3 = \emptyset$ for the R_3 in equation (12).

Now let us show that the set of inadmissible edges in K equals $E \cap R_2$ for the R_2 given in equation (11). Since (A_*, B^*) and (A^*, B_*) are minimum covers of the graph, Theorem 2 tells us that graph edges in $A_* \times B^*$ and $A^* \times B_*$ are inadmissible. Observe that for $i < j$ the set $S_i \times T_j$ is contained in $A_i \times B_i$, as seen in equations (6), (7). As each (A_i, B_i) is a minimum cover of the graph, Theorem 2 once again tells us that graph edges in $S_i \times T_j$ for $i < j$ are inadmissible. But these results merely tell us that $E \cap R_2$ is a subset of the set of inadmissible edges. We want to show that it *equals* the set of inadmissible edges in K.

In the case of $A_* = A^*$, the graph had only one minimum cover (A_*, B^*), which simplified the task of finding the set of inadmissible edges. For $A_* \neq A^*$, there could be minimum covers (A, B) which are not equal to any of (A_*, B^*), (A_1, B_1), (A_2, B_2), $\ldots$, (A_{k-1}, B_{k-1}), (A^*, B_*). However, in their 1958 paper [8], Dulmage and Mendelsohn proved that the any minimum cover (A, B) of K can be represented by a combination of the S_i's and T_i's as described in the following theorem.

Theorem 8. *For a bipartite graph K having a finite cover number, let A_*, B_*, S_1, S_2, $\ldots$, S_k, T_1, T_2,$\ldots$,T_k be the sets obtained in the procedure described earlier in this section. Let (A, B) be any minimum cover of K. Then there exist complementary subsets Δ and Π of $\{1, 2, \ldots, k\}$ such that*

$$A = A_* \bigcup \left(\bigcup_{i \in \Delta} S_i \right),$$

$$B = \left(\bigcup_{j \in \Pi} T_j \right) \bigcup B_*.$$

This theorem (in combination with Theorem 2) tells us that the set of inadmissible edges equals the union of $E \cap (A \times B)$ as the set Δ varies over the 2^k subsets of $\{1, 2, \ldots, k\}$ with $\Pi = \Delta^c$. But all such sets $E \cap (A \times B)$ are contained in R_2. To see this, note that

$$\begin{aligned}
E \cap [A \times B] &= E \cap \Big[(A_* \times B) \bigcup (\cup_{i \in \Delta} S_i \times B) \Big] \\
&= E \cap \Big[(A_* \times B) \bigcup (\cup_{i \in \Delta} S_i \times B_*) \bigcup (\cup_{i \in \Delta} S_i \times \cup_{j \in \Pi} T_j) \Big] \\
&\subseteq E \cap \Big[(A_* \times B^*) \bigcup (A^* \times B_*) \bigcup (\cup_{i \in \Delta} S_i \times \cup_{j \in \Pi} T_j) \Big] \qquad (18) \\
&= E \cap \left[(A_* \times B^*) \bigcup (A^* \times B_*) \bigcup_{i \in \Delta, j \in \Pi, i < j} (S_i \times T_j) \right] \qquad (19) \\
&\subseteq E \cap R_2, \qquad (20)
\end{aligned}$$

where the subset relation in (18) follows from equations (4) and (5) which show that $\cup_{i \in \Delta} S_i \subseteq A^*$ and $B \subseteq B^*$. The equality in (19) follows from two observations: (a) the graph cannot have edges in $S_i \times T_j$ for $i > j$, as discussed in our

argument showing $E \cap R_3 = \emptyset$, and (b) $i \neq j$ when $i \in \Delta$ and $j \in \Pi = \Delta^c$. The subset relation in (20) follows from definition of R_2 in (11). Thus, we conclude that the set of inadmissible edges in the graph K equals $E \cap R_2$.

Finally, as R_1, R_2, R_3 form a partition of $S \times T$ with $E \cap R_3 = \emptyset$, the set of admissible edges must equal $E \cap R_2^c$ which is equal to $E \cap R_1$. This completes the proof that the expressions for R_1, R_2, R_3 in equations (10), (11), (12), satisfy the properties of a DM decomposition given in Definition 8. □

References

1. dmperm: MATLAB function for Dulmage-Mendelsohn decomposition, `https://in.mathworks.com/help/matlab/ref/dmperm.html`
2. Monero Blackball Databases, `https://github.com/monero-blackball/monero-blackball-site`, Last Accessed: August 25, 2021
3. Monero Blackball Tool Code, `https://github.com/monero-project/monero/blob/master/src/blockchain_utilities/blockchain_blackball.cpp`
4. Monero Original GitHub Repository, `https://github.com/XmanXU/monero-original`
5. Monero Scheduled Software Upgrades, `https://github.com/monero-project/monero/#scheduled-software-upgrades`
6. MoneroV GitHub Repository, `https://github.com/monerov/monerov`
7. Davis, T.A.: CSparse: A concise sparse matrix package, `https://people.engr.tamu.edu/davis/suitesparse.html`
8. Dulmage, A.L., Mendelsohn, N.S.: Coverings of bipartite graphs. Canadian Journal of Mathematics **10**, 517–534 (1958). https://doi.org/10.4153/CJM-1958-052-0
9. Goodell, B.: Perfect privacy or strong deniability? In: Monero Konferenco (2019), `https://youtu.be/xicn4rdUj_Q`
10. Hinteregger, A., Haslhofer, B.: An empirical analysis of Monero cross-chain traceability. In: Financial Cryptography and Data Security. pp. 150–157 (2019)
11. Kumar, A., Fischer, C., Tople, S., Saxena, P.: A traceability analysis of Monero's blockchain. In: European Symposium on Research in Computer Security. pp. 153–173. Springer (2017)
12. Möser, M., Soska, K., Heilman, E., Lee, K., Heffan, H., Srivastava, S., Hogan, K., Hennessey, J., Miller, A., Narayanan, A., Christin, N.: An empirical analysis of traceability in the Monero blockchain. Proceedings on Privacy Enhancing Technologies **2018**(3), 143–163 (2018). https://doi.org/10.1515/popets-2018-0025
13. Noether, S.: Sets of spent outputs. Monero Research Lab Technical Report MRL-0007 (Nov 2018), `https://www.getmonero.org/resources/research-lab/`
14. Pothen, A., Fan, C.J.: Computing the block triangular form of a sparse matrix. ACM Trans. Math. Softw. **16**(4), 303–324 (Dec 1990). https://doi.org/10.1145/98267.98287
15. Saberhagen, N.v.: CryptoNote v 2.0. White paper (2013), `https://cryptonote.org/whitepaper.pdf`
16. Wijaya, D.A., Liu, J., Steinfeld, R., Liu, D.: Monero ring attack: Recreating zero mixin transaction effect. In: 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE). pp. 1196–1201 (2018). https://doi.org/10.1109/TrustCom/BigDataSE.2018.00165

17. Ye, C., Ojukwu, C., Hsu, A., Hu, R.: Alt-coin traceability. Cryptology ePrint Archive, Report 2020/593 (2020), https://eprint.iacr.org/2020/593
18. Yu, J., Au, M.H.A., Esteves-Verissimo, P.: Re-thinking untraceability in the Cryptonote-style blockchain. In: 2019 IEEE 32nd Computer Security Foundations Symposium (CSF). pp. 94–106 (2019). https://doi.org/10.1109/CSF.2019.00014
19. Yu, Z., Au, M.H., Yu, J., Yang, R., Xu, Q., Lau, W.F.: New empirical traceability analysis of Cryptonote-style blockchains. In: Financial Cryptography and Data Security. pp. 133–149 (2019)