Correlation-Aware Anonymization of High-Dimensional Data

Andrea Straforini
Giacomo Benedetti
Gabriele Armanino

Introduction

- Classic method for anonymization, such as K-anonymity, are based on generalization and suppression and are not suitable for high-dimensional datasets.[1]

- The Cahd technique bring three specific contribution:
  - It introduces a novel representation of transaction data which takes advantage of data sparseness, preserves correlations among items and arrange transactions with similar QID in close proximity to each other.
  - It devises an efficient heuristic to create anonymized groups with low information loss.
  - It outperform already existing method in both data utility and computational overhead

Notations

- Transaction set to be anonymized $T = \{t_1, t_2, ..., t_n\}, n = |T|$
- Set of items $I = \{i_1, i_2, ..., i_n\}, d = |I|$
- Set of sensitive items $S = \{s_1, s_2, ..., s_m\} \subset I, m = |S|$
- QID, quasi-identifier, if associated with external knowledge it can lead to the identification of the individual.

Cahd Objectives

Privacy Requirement

- In group G the privacy is: $p^G = min|G|/f_i$
- f_i^G ,number of occurrences of the sensitive item i in group G.

- In the whole P partitioning of T $p^P = \min_{G \in P} p^G$

Utility Requirement

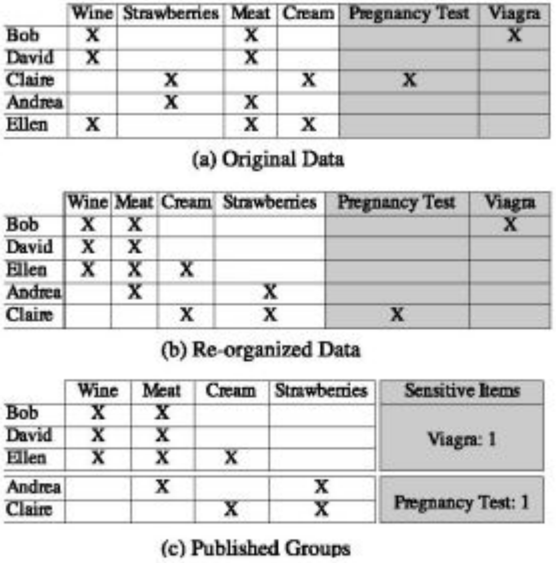
- In our case we want to minimize the reconstruction error. A metric is provided to evaluate the total information lost after performing data anonymization.

$$KL\_Divergence(Act, Est) = \sum_{\forall cellC} Act_C^s log \frac{Act_C^s}{Est_C^s}$$

Cahd Steps

1. The first step is to convert the sparse matrix into a band matrix through the use of the RCM algorithm. This way we make sure that neighboring transactions are similar. (greater number of QIDs in common). *(b)*

2. The second step is the creation of groups of *p* elements containing non-conflicting sensitive transactions and which ensure a degree of privacy *p*. *(c)*



	Wine	Strawberries	Meat	Cream	Pregnancy Test	Viagra
Bob	X		X			X
David	X		X			
Claire		X		X	X	
Andrea		X	X			
Ellen	X		X	X		

(a) Original Data

	Wine	Meat	Cream	Strawberries	Pregnancy Test	Viagra
Bob	X	X				X
David	X	X				
Ellen	X	X	X			
Andrea		X		X		
Claire			X	X	X	

(b) Re-organized Data

	Wine	Meat	Cream	Strawberries	Sensitive Items
Bob	X	X			Viagra: 1
David	X	X			
Ellen	X	X	X		
Andrea		X		X	Pregnancy Test: 1
Claire			X	X	

(c) Published Groups

Step 1. Band Matrix

1. The transaction data table is converted into an A squared matrix adding fake products if the number of transactions is not equal to the products' one.
2. Then we compute a symmetric matrix $B = A + A^T$
3. We consider the newly created matrix as an adjacency list and create a graph to be used by the Reverse-Cuthill_McKee algorithm.
4. The matrix obtained is called Band Matrix.

Notes:

The original papers compute the transaction matrix using $B = AA^T$
We noticed that, in our implementation, the computational cost derived from it is too high and therefore we decided to use the sum.

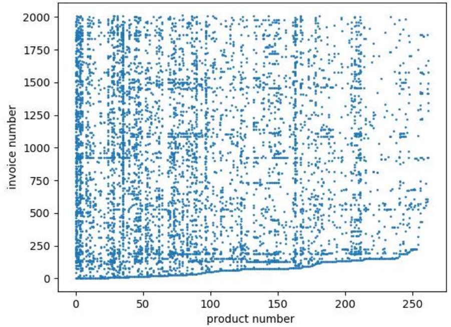
Implementation Details

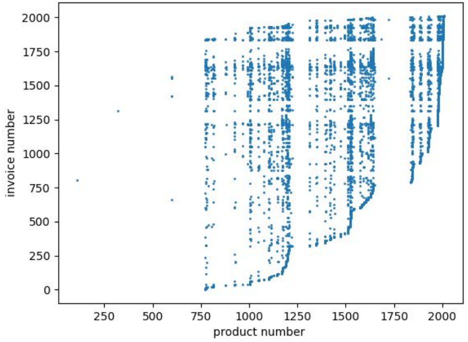
For the creation and management of the sparse matrices we have made use of the Eigen library.*[1]*

We have encapsulated the management and the behavior of the data table in the TransactionTable class which allows, thanks to auxiliary data structures, access to rows in constant time and access to specific elements in log(|row|).

Band Matrix

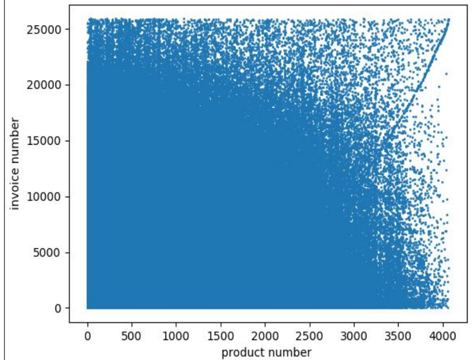
KDD Cup 2000: Online retailer website clickstream analysis repo (BMS1) *limited to 2013 transactions*

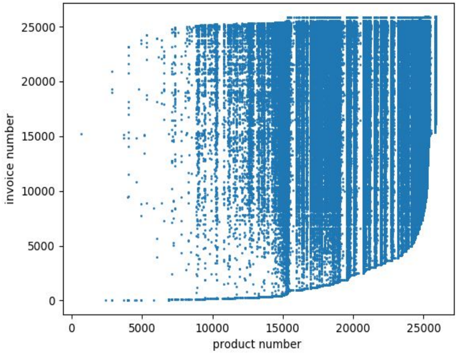




Online Retail Data Set from UCI ML repo

transactions 2010-2011 for a UK-based and registered non-store online retail 25899 transactions[2]





Note:Y axis is inverted

Step 2. Cahd Group Formation Heuristic

- To satisfy the desired privacy requirement, each sensitive transaction needs to be grouped with non-sensitive ones or with other transactions that do not contain the same sensitive item.

- CAHD is an heuristic which tries to obtain a near-optimal solution exploiting the band matrix representation. In this type of representation similar transaction will be positioned close to each other.

Implementation Details

For the creation of the band matrix we have made use of the Boost Graph Library implementation of the Rcm algorithm.*[3]*

We have used the Version 2 of the rcm algorithm which finds a good starting vertex using the pseudo-peripheral pair heuristic (among each component).

It is probable that, given the particular arrangement of the adjacency matrix (A+AT), a better heuristic can be used.

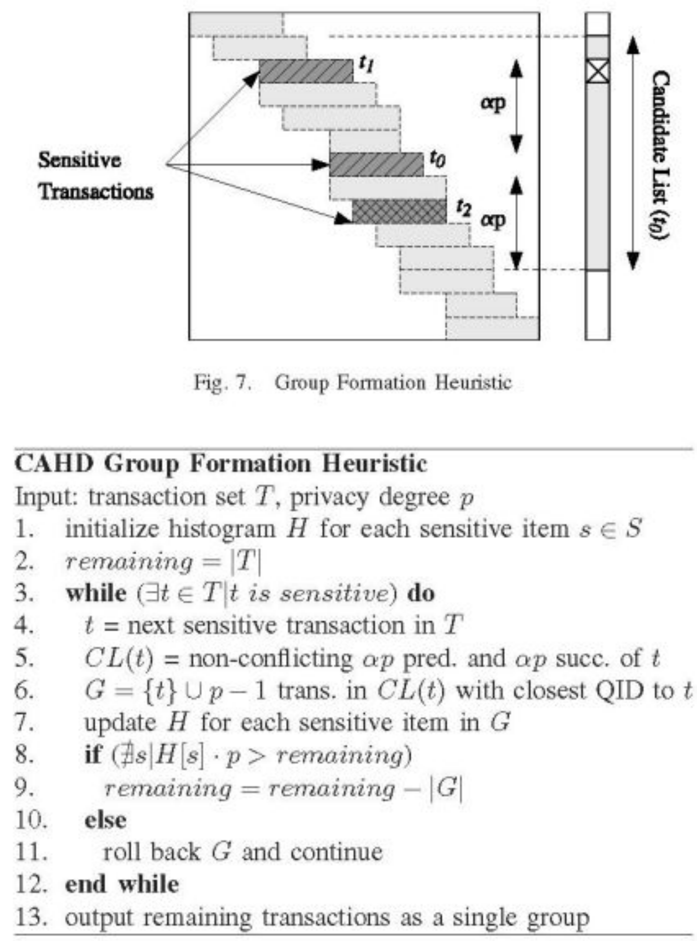
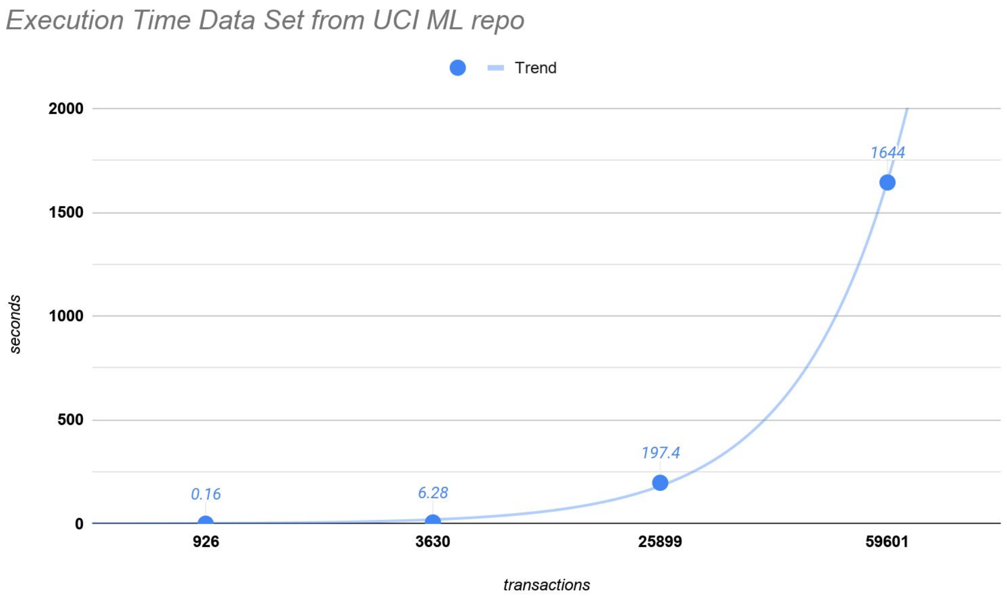

Sensitive
Transactions
t_1
t_0
t_2
αp
αp
Candidate List (t_0)

Fig. 7. Group Formation Heuristic

CAHD Group Formation Heuristic

```
Input: transaction set $T$, privacy degree $p$
initialize histogram $H$ for each sensitive item $s \in S$
$remaining = |T|$
while ($\exists t \in T | t$ $is$ $sensitive$) do
  $t$ = next sensitive transaction in $T$
  $CL(t)$ = non-conflicting $\alpha p$ pred. and $\alpha p$ succ. of $t$
  $G = \{t\} \cup p-1$ trans. in $CL(t)$ with closest QID to $t$
  update $H$ for each sensitive item in $G$
  if ($\nexists s | H[s] \cdot p > remaining$)
    $remaining = remaining - |G|$
  else
    roll back $G$ and continue
end while
output remaining transactions as a single group
```

Execution Time



Results data

p=4 a=3

file name	transazioni	prodotti	n*c	items sensibili	items name	execution time
shuf1000Data.csv	926	751	695426	2+2	22729-22616	0.16
shuf80_000Data.csv	3630	3059	11104170	48+64	22816-22377	6.28
shufData.csv(541909)	25899	4070	105408930	266+209	22816-22377	197.4
BMS1formattedshuf.csv(149639)	59601	497	29621697	1389+347	10877-18423	1644
BMS1formatted6000.csv	2013	263	529419	33+26	10877-18423	3.6

Possible Vulnerability

In the Transaction Group, the sensitive transaction is always in the first position; so, for each group, we need to shuffle the transaction list.

The use of random functions exposes the implementation to vulnerabilities of type: *random number generator attack[4]* which could completely deanonymize the table.
We therefore left the task of implementing/use effective randomization, because it is outside the scope of the current project.
It is also likely that this vulnerability could occur in numerous CAHD implementations.

Resources:

[1]Eigen Library : *http://eigen.tuxfamily.org/index.php?title=Main_Page*

[2]Online Retail Data Set from UCI ML repo : https://www.kaggle.com/jihyeseo/online-retail-data-set-from-uci-ml-repo/data

[3] R.C.M Boost c++ Libraries. *https://www.boost.org/doc/libs/1_72_0/libs/graph/doc/cuthill_mckee_ordering.html*

[4]Random number generator attack: *https://en.wikipedia.org/wiki/Random_number_generator_attack*