Detecting Ponzi schemes on Ethereum

Reproduce documentation - Marion le Tilly – July 2018

# Dataset aggregation

## Ponzi addresses

Addresses available at : <goo.gl/CvdxBp>.

## Non Ponzi addresses

Addresses available at: <https://docs.google.com/spreadsheets/d/1pd9rO2Hykqhe3U9kzMyLvV23nbeP-4ILulPlT1VOqlU/edit?usp=sharing>

Or using a scrapper available at: <https://github.com/woniesong92/etherscan-scraper>

They’re pulled from etherscan verified smart contracts.

## Data aggregation

In order to gather data about all the addresses I used the web3 python API to obtain the bytecode directly from Ethereum blockchain, and the Etherscan API to gather normal and internal transactions for each contract.

# Features computation

## Code features

To diassemble the bytecode and turn it into opcodes, I used *evmdis* diassembler (available at: <https://github.com/Arachnid/evmdis>) and then I computed each opcode frequency.

## Transactions features

I modify the data structure of my dataset to extract the features describe in the paper

# Classification models

## Models performance

I used weka (weka-3-8-2) to train several models using cross-validation: J48, Random Forest and SGD, using some filters such as SpreadSubSample to subsampled my dataset.

I calculated the performance of my model by taking the average performance considering different random seed.

## Features robustness

I used weka attributes selection to rank features according to their relevance in the classification, using both information gain and chi-square metrics.