INPT
المعهد الوطني للبريد والمواصلات
ⵓⵔⵣⵣⵓ ⵓⵏⵎⵓⵔⵔⴰⵙ ⵏ ⵜⵓⵙⵙⵓⵔⵜ ⴰ ⵙⵎⵔⵓⵙⵙⵓ
Institut National des Postes et Télécommunications

Go digital by INPT

Implementation of a Blockchain based application to manage lost Bitcoins

AUTHORS:

KARIM HABOUCH
MOHAMED AMINE AJINOU

SUPERVISORS:

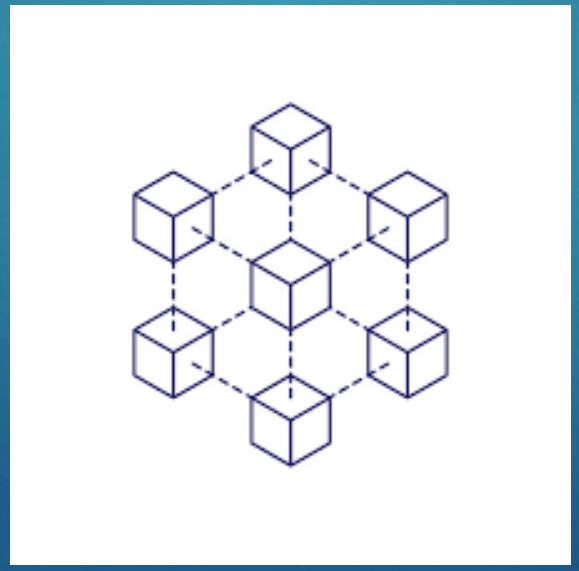
PR. MERYEM AYACHE
PR. AMJAD GAWANMEH

Saturday July, 17th 2021

Introduction

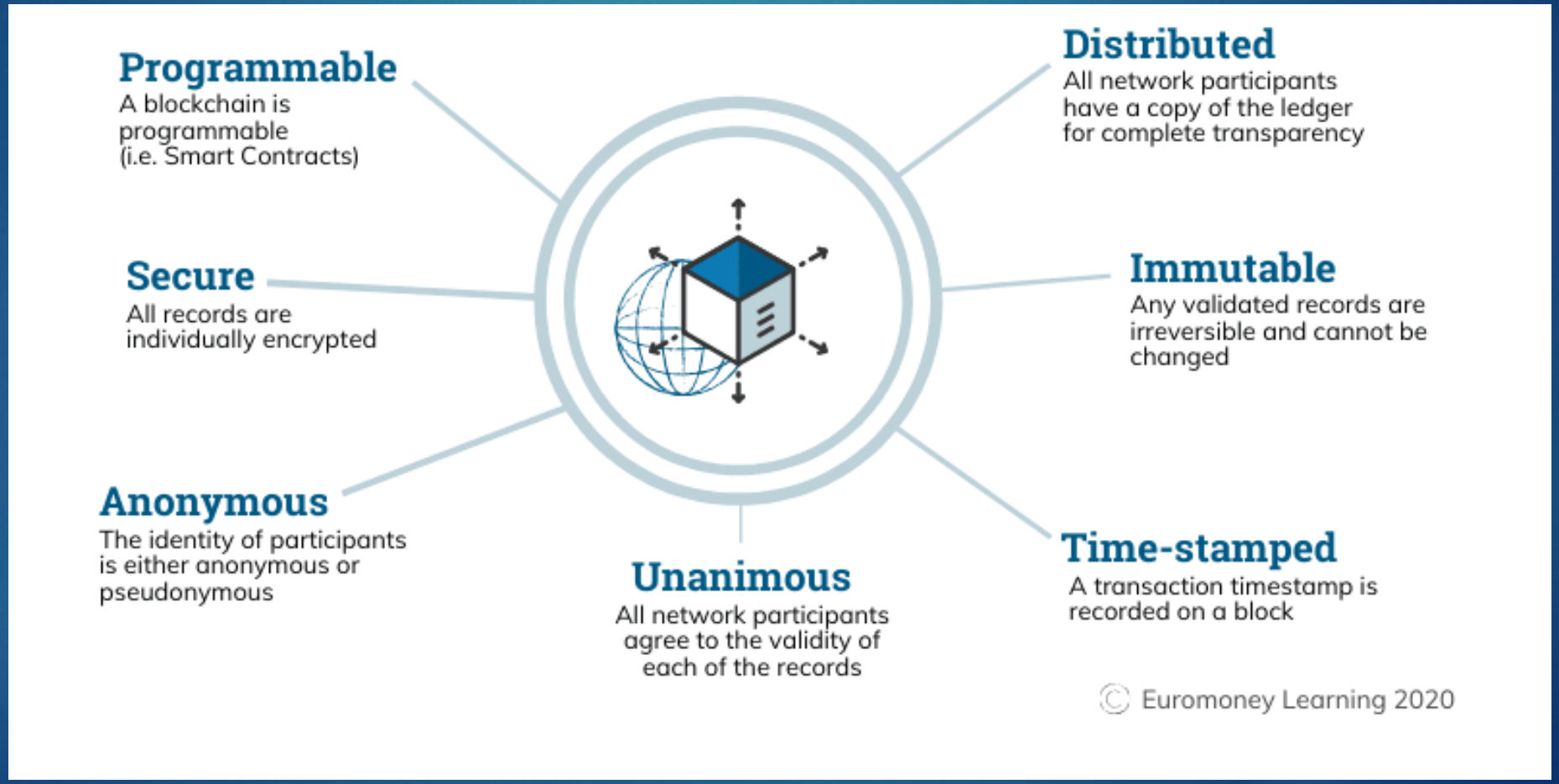
Blockchain

Blockchain is a system of recording information in a way that makes it difficult or impossible to change, hack, or cheat the system.



Introduction

Blockchain Features



Introduction

Blockchain use cases

Introduction

History of Bitcoin

Bitcoin is the first decentralized digital currency, and as such it is a revolutionary technology invention. It changed the way we compute things and the way we operate software and computers. Bitcoin is considered to be the next big wave of change after the internet.

Introduction

Rise of Bitcoin value

From 1$ in 2010 to nearly 50,000$ in 2021

Problematic

- Around 4 million bitcoins lost out of the 18 million that are mined so far (3 million bitcoins left to be mined !)
- Digital wallets are protected by cryptography algorithms, namely asymmetric RSA keys (public & private keys)
- When the private key of a wallet is lost, the access to the wallet is no longer possible.

PROBLEMATIC: How to recover lost wallets' balances to use during the mining rewards and get lost bitcoins back into network circulation?

Simulating the environment

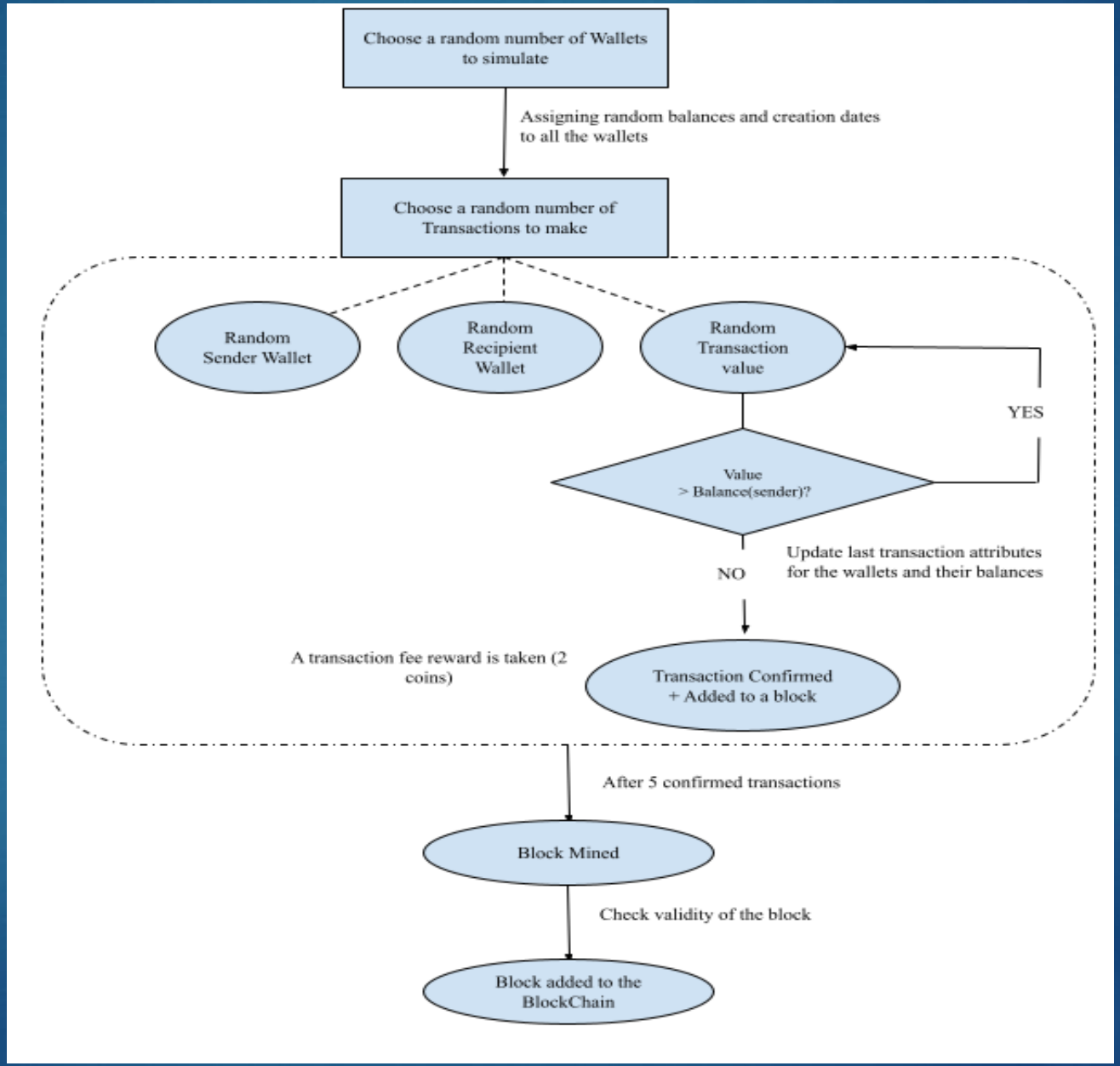
- Simulation using python
- « Blockchain » folder contains classes like blocks, wallets and transactions
- « Crypto » contains backend functions like rsa.py and utils.py (hashing and encryption functions)
- test.py are for main execution of the simulation
- Json files generated throughout the code to store the output of the simulation

```
Structure du dossier
Le numéro de série du volume est 2619-FCE2
C:.
    block.json
    proba.py
    test.py
    transaction.json
    wallet.json

├───blockchain
        block.py
        blockchain.py
        transaction.py
        wallet.py
        __init__.py

└───crypto
        rsa.py
        utils.py
        __init__.py
```

Simulation Overview



Automate wallets generation

- Setup a number of wallets with random balance and creation date
- Wallet_dict is a dictionary where all generated wallets are stored

```
class Wallet:

    # Class attributes
    start_date = datetime.date(2010, 1, 1)
    end_date = datetime.date(2017, 1, 1)
    transaction_end_date = datetime.date(2020, 1, 1)
    days_between = (end_date - start_date).days
    # Dictionary containing all the created wallets {name:instance Wallet}
    wallet_dict = {}

    def __init__(self,id):
        self.id = id
        self.name = "wallet {}".format(str(id))
        self.creation = self.__creation_date()
        self.balance = self.__rand_balance()
        self.last_transaction = self.creation
        self.__add_wallet_dict()
```

Automate wallets generation

To simulate a closer simulation to reality, the program takes into consideration the Bitcoin wallet distribution. The percentages of balance ranges are applied to the number of generated wallets and their balances are therefore updated respecting what is stated on the figure below (Bit Info Charts),

Balance, BTC	Addresses	% Addresses (Total)	Coins	$USD	% Coins (Total)
(0 - 0.001)	19553997	51.05% (100%)	3,978 BTC	134,207,147 USD	0.02% (100%)
[0.001 - 0.01)	9747695	25.45% (48.95%)	37,055 BTC	1,250,223,465 USD	0.2% (99.98%)
[0.01 - 0.1)	5835719	15.23% (23.51%)	189,455 BTC	6,392,108,218 USD	1.01% (99.78%)
[0.1 - 1)	2371517	6.19% (8.27%)	741,813 BTC	25,028,387,721 USD	3.96% (98.77%)
[1 - 10)	651032	1.7% (2.08%)	1,672,357 BTC	56,424,452,410 USD	8.92% (94.81%)
[10 - 100)	130544	0.34% (0.38%)	4,247,435 BTC	143,306,238,911 USD	22.65% (85.9%)
[100 - 1,000)	13947	0.04% (0.04%)	3,948,436 BTC	133,218,163,894 USD	21.06% (63.24%)
[1,000 - 10,000)	2063	0.01% (0.01%)	5,180,826 BTC	174,798,346,375 USD	27.63% (42.19%)
[10,000 - 100,000)	80	0% (0%)	2,047,564 BTC	69,083,721,331 USD	10.92% (14.56%)
[100,000 - 1,000,000)	4	0% (0%)	681,888 BTC	23,006,555,838 USD	3.64% (3.64%)

Transaction Class

- Different methods to calculate the hash and set random transaction date for every transation made
- Transaction_dict is a dictionary where all generated transactions are stored

```
class Transaction:

    sequence = 0  # id of the new transaction
    # Dictionary that stores all the transactions
    transaction_dict = {}

    def __init__(self, from_id, to_id, value):
        self.sender = from_id
        self.recipient = to_id
        self.value = value
        self.date = None
        self.transaction_id = None
```

Block Class

- Mine blocks after each set of 5 transactions with a chosen difficulty and check its integrity
- block_dict is a dictionary where all mined blocks are stored

```
class Block:

    # Dictionary containing all blocks
    block_dict = {}

    def __init__(self, block_id, previous_hash):
        self.previous_hash = previous_hash
        self.nonce = 0
        self.timestamp = None
        self.transactions = []
        self.hash = self.calculate_hash()
        self.name = "block {}".format(str(block_id))
```

Simulating 200,000 transactions

```
data_wallet = pd.read_json (r'wallet.json').T
data_wallet.sample(10)
```

	id	name	creation	balance	last_transaction
wallet 5759	5759	wallet 5759	2015-06-09	90	2019-09-19
wallet 2672	2672	wallet 2672	2012-08-19	0	2019-04-21
wallet 6964	6964	wallet 6964	2010-06-03	72	2019-12-29
wallet 1151	1151	wallet 1151	2014-06-28	18	2019-08-04
wallet 8183	8183	wallet 8183	2010-04-25	319	2019-09-16
wallet 3939	3939	wallet 3939	2011-08-09	7	2016-07-28
wallet 3545	3545	wallet 3545	2011-09-16	12	2019-12-29
wallet 7364	7364	wallet 7364	2015-06-07	46	2019-12-25
wallet 4430	4430	wallet 4430	2012-05-12	13	2019-12-17
wallet 8507	8507	wallet 8507	2010-12-25	821	2018-10-30

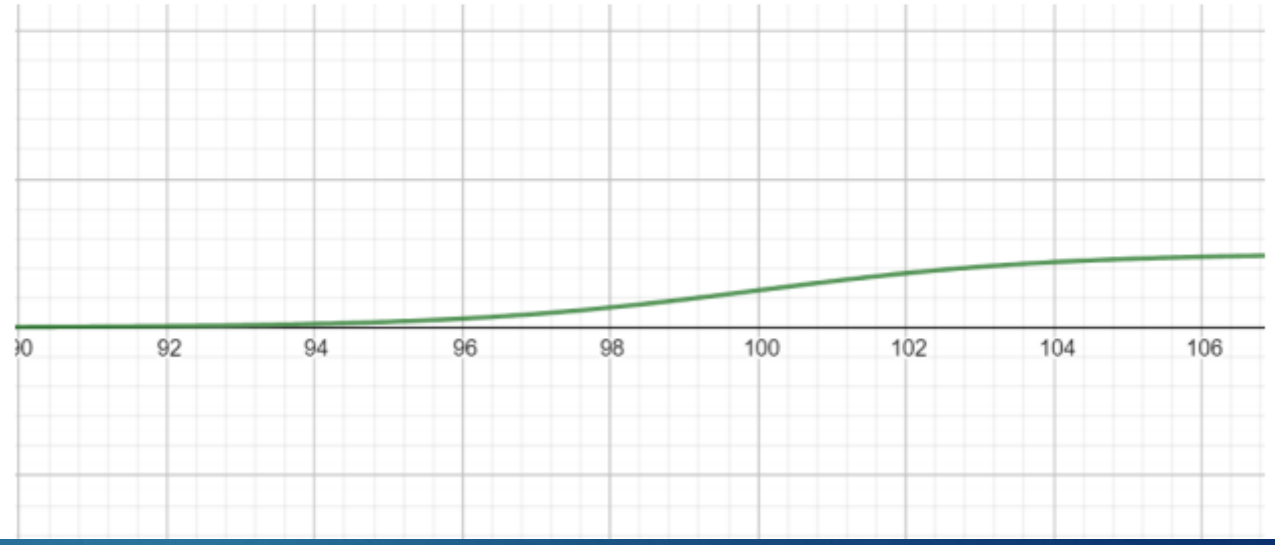
- Simulating 100,000 wallets and 200,000 transaction based on a randomization algorithm: random sender & recipient wallet IDs, balances and timestamps
- Store the output in json files for further analysis using pandas and statistics modules

Probabilistic Model

- This function takes the timestamp between now and the last transaction as an argument
- The hyperparameter T is calculated using the median of our values

Example below for T = 100

$$f(x) = \frac{1}{1 + e^{\frac{-(x-100)}{2}}}$$

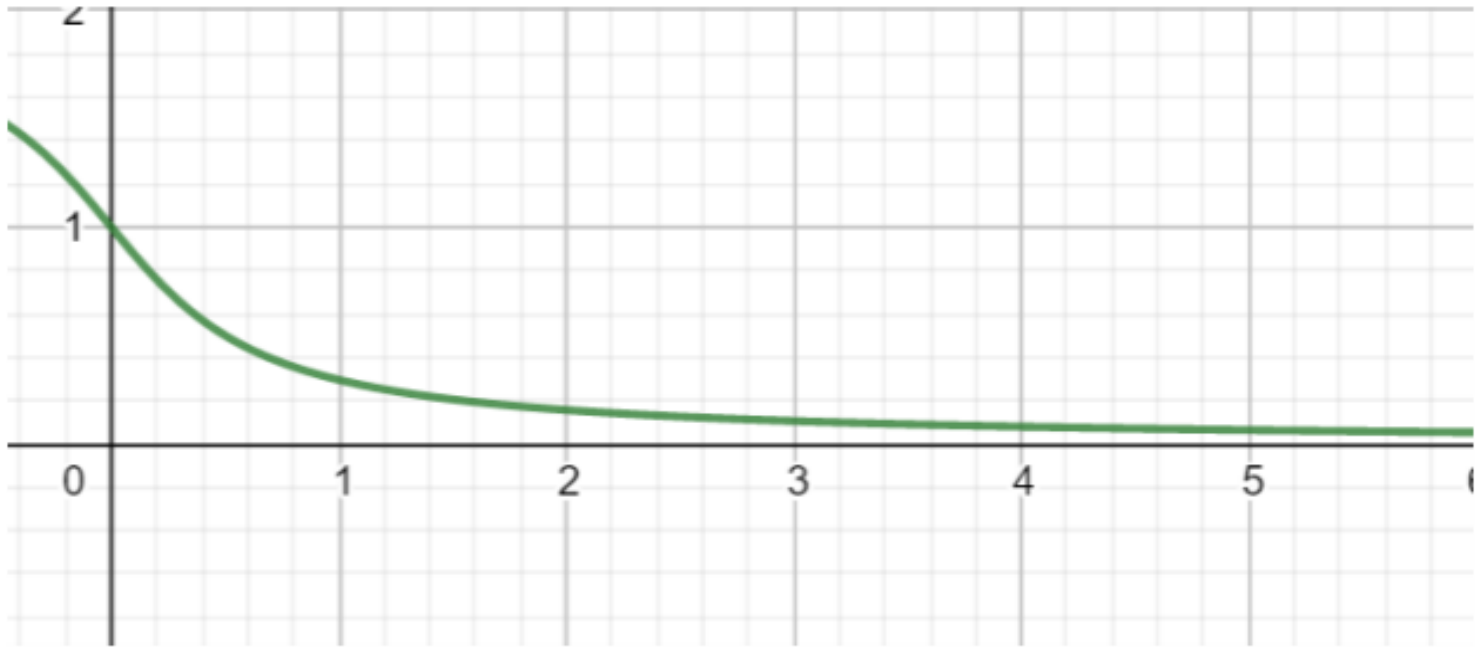


Probabilistic Model

- This function takes the wallet's balance as an argument
- The hyperparameter N is calculated using the median of our balances

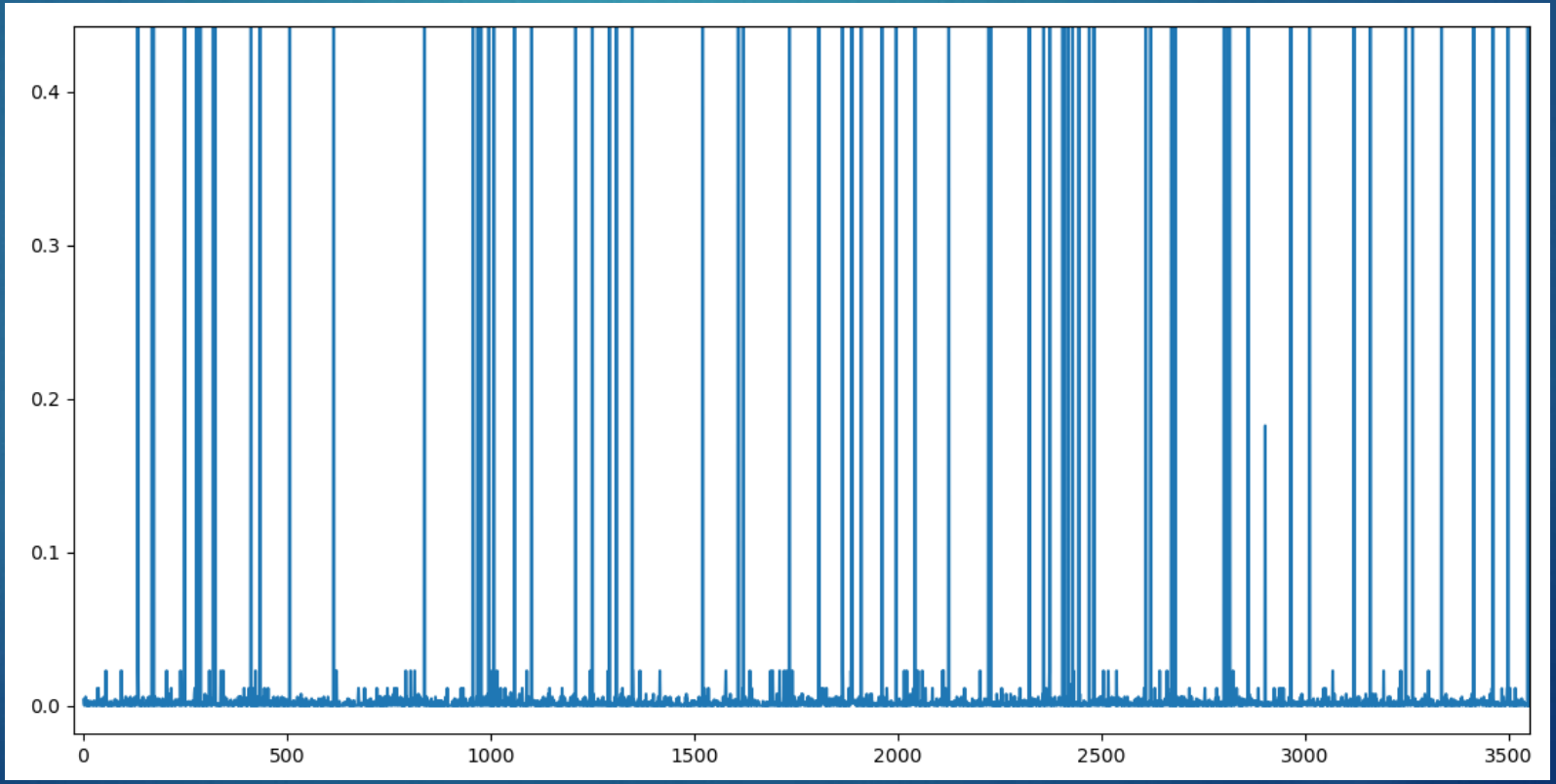
$$f(x) = 1 - \frac{2}{\pi} \tan^{-1}(2x)$$

For N = 2



Model result (plot)

- The multiplication of the two functions images generates the probability we are looking for.
- The higher the probability is, the most likely our wallet is considered lost



Model result (plot)

Considering the graph, we set P=0,4 as our border value

```
The Wallet that is most likely to be lost Nº 91:      wallet 1621     Probability ==> 1.0
The Wallet that is most likely to be lost Nº 92:      wallet 1609     Probability ==> 1.0
The Wallet that is most likely to be lost Nº 93:      wallet 320      Probability ==> 0.9999999999999996
The Wallet that is most likely to be lost Nº 94:      wallet 2418     Probability ==> 0.9999999999999993
The Wallet that is most likely to be lost Nº 95:      wallet 6690     Probability ==> 0.9999999999999971
The Wallet that is most likely to be lost Nº 96:      wallet 2042     Probability ==> 0.9999999999999656
The Wallet that is most likely to be lost Nº 97:      wallet 4280     Probability ==> 0.9999999999622486
The Wallet that is most likely to be lost Nº 98:      wallet 2801     Probability ==> 0.9999999995400946
The Wallet that is most likely to be lost Nº 99:      wallet 4869     Probability ==> 0.9999999992417439
The Wallet that is most likely to be lost Nº 100:     wallet 3498     Probability ==> 0.9999999987498471
The Wallet that is most likely to be lost Nº 101:     wallet 2860     Probability ==> 0.9999998144608981
The Wallet that is most likely to be lost Nº 102:     wallet 134      Probability ==> 0.9999998144608981
The Wallet that is most likely to be lost Nº 103:     wallet 3264     Probability ==> 0.999997739675702
The Wallet that is most likely to be lost Nº 104:     wallet 434      Probability ==> 0.9999938558253978
The Wallet that is most likely to be lost Nº 105:     wallet 324      Probability ==> 0.9999724643088853
```

After applying the model

- After each transaction, the reward will be taken firstly from the wallet to be more likely lost to unlock the lost bitcoins by adding them again to the mining pool

Conclusion

- This probabilistic model needs to be sharpened more using AI and be applied to different random datasets reflecting reality more closely
- Since this proposed method is not practical on Bitcoin, it can be considered for similar new public coins in the future !

Thank you for your attention