VIETNAM NATIONAL UNIVERSITY HCMC
HCMC UNIVERSITY OF TECHNOLOGY
COMPUTER SCIENCE AND ENGINEERING

MATHEMATICAL MODELING

Assignment

Mathematical model for UTXO selection

Tutor: Huynh Tuong Nguyen (htnguyen@hcmut.edu.vn)
Student members group 3: Nguyen Nhat Tan - 1713074
Nguyen Trung Tinh - 1713521
Dang Van Dung - 1710853
Cao Dang Dung - 1710849
Nguyen Truong Dinh Quan - 1712825
Chung Minh De - 1711020
Trinh Duc Truong - 1713759

HO CHI MINH city, 5/2019

Contents

1 Introduction

A decentralized cryptocurrency is a digital asset using cryptography systems collectively to secure the transactions and integrity without the third partys intervention. All the transactions in the system are registered on a ledger called blockchain that is constituted of a sequence of blocks. Each block contains an unfixed number of transactions and also a hash of the previous block so that all transactions in the blockchain are immutable and valid. A significant example of this kind cryptocurrency is Bitcoin introduced in 2008 which has currently over 141 billion in the coin market, with an average of 229K transactions daily and around 183.89 GB of storage1.

In blockchain networks, cryptocurrency balances are managed in two different models including account-based and transaction-based models (UTXO model). For transactionbased blockchains such as Bitcoin and Altcoin, they use transaction outputs to spend on new transactions as inputs. Any transaction output that has not become the input of any transaction yet is called Unspent Transaction Output (UTXO). Each cryptocurrency balance for an address is represented by a set of Unspent Transaction Outputs (UTXOs). When a user consumes his currency to transact with another one, a transaction is generated by selecting his UTXOs as inputs, and creating new UTXOs as outputs for his receiver. On the other hand, for account-based blockchains such as Ethereum, each cryptocurrency balance works like the traditional banking world. This means that each account updates current balance and experiences the information transfers with state transition 1https://coinmarketcap.com [6]. In order to take advantage of both models, Zahnentferner et al. [9] defined a new transaction type in which inputs and outputs are merely mapped from addresses to values through the transaction and then update the account balances of these addresses.

In transaction-based blockchains, the selection strategy of UTXOs for the transaction plays an essential role in cryptocurrency balance management of any wallet. An optimized coin selection strategy should satisfy hard constraints and essential goals of three principal groups such as users, community and miners. Owing to consumers, they would like to create a transaction that minimizes the transaction fee and also reserves the privacy of their behaviors. By contrast, miners focus on mining transactions which have a higher fee as much as possible. To the community, the large UTXO pool size becomes a dramatic problem because it drops down the transaction processing performance and also produces a high cost of memory consumption. Like Bitcoin system, a snapshot of the current state required additional space in the memory to store objects for processing transactions .

In this work, we proposed an effective strategy to select a set of UTXOs for a given transaction which should cost a minimum fee for miners or gather as much as possible small UTXO in order to reduce the UTXO pool size. Our proposal models will be evaluated on Bitcoin that is currently a great snapshot of blockchain and attracts many customers creating lots of transactions daily

2 Problem formulation

Our goal is to propose an effective strategy in selecting an UTXO file suitable for a certain transaction, able to meet the following constraints:
1. Minimize transaction size (transaction size) to have a dark transaction fee minimum.
2. Minimize UTXO file (UTXO pool)

3 Proposed model

1.Model 1
Variables:Decision

$$x_i = \begin{cases} 1 & \text{if UTXO } u_i \text{ is chosen} \\ 0 & \text{otherwise} \end{cases}$$

Model 1 - Intermediate variables:
y: transaction size
z_v: a value of change output (amount)
z_s: size of change output
$z_s = \beta * \sigma$

$$\sigma = \begin{cases} 0 & 0 \leq z_v \leq \epsilon \\ 1 & \epsilon < z_v \end{cases}$$

Model 1 - Constraints: are defined in the proposed selection strategy can be formulated as followings:
A transaction size may not exceed maximum block data size

$$y = \sum_{i|u_i \in U} s_i^u + \sum_{j|o_j \in U} s_j^o + \beta * \sigma \leq M$$

A transaction must have sufficient value for consuming.

$$\sum_{i|u_i \in U} v_i^u * x_i = \sum_{j|o_j \in O} v_j^o + fee + z_v$$

With $fee = \alpha * (\sum_{i|u_i \in U} s_i^u + \sum_{j|o_j \in U} s_j^o + \beta * \sigma)$
All the transaction outputs must be higher than the dust threshold to certain that this transaction is relayed to the network and confirmed.

$$T \leq \sum_{j|oj \in O} v_j^o$$

x_i is binary variable

$$i|u_i \in U : x_i \in 0, 1$$

Objective function: Minimizing transaction size

$$\text{minimize y}$$

2)Model 2:
The Model 2 is to maximize the number of selected UTXOs for shrinking the UTXO pool size. The Model 2 is built based on the result obtained Model 1 as follows.
Model 2 - Variables: include all variables of Model 1
Model 2 - Constraints: include all constraints of Model
1 and an extra constraint as follow

$$y < (1 + \gamma)Y$$

y is the minimal transaction size obtained by Model 1.
γ: is a coefficient $(0 \leq \gamma \leq 1)$
If y closes to 0, we would like to keep minimum transaction size obtained from Model 1. In otherwise, a transaction with an appropriate size is created by a number of UTXOs as large as possible.
Objective function: Maximizing the number of UTXOs

$$\text{maximize } (\sum_{i|u_i \in U} x_i - \sigma)$$

4 Experimental evaluation

Input format:

Input Parameters	Description
$U = \{u_1, ..., u_n\}$	set of UTXOs
$O = \{o_1, ..., o_n\}$	set of transaction outputs
$V^u = \{v_1^u, ..., v_n^u\}$	set of UTXO's values
$V^o = \{v_1^o, ..., v_m^o\}$	set of transaction output's values
$S^u = \{s_1^u, ..., s_n^u\}$	set of transaction input size, with input is chosen from UTXO ui
$S^o = \{s_1^o, ..., s_m^o\}$	set of transaction output's size.
M	maximum size of a transaction
α	fee rate
T	dust threshold
ϵ	minimum of change output

Output format:

```
Model_1:
MINIMIZE
34*sigma + 148*x_0_ + 148*x_1_ + 148*x_2_ + 148*x_3_ + 68
SUBJECT TO
_C1: 34 sigma + 148 x_0_ + 148 x_1_ + 148 x_2_ + 148 x_3_ <= 1048508

_C2: - 255 sigma + 4898890 x_0_ + 46690094 x_1_ + 19948890 x_2_
 + 20948890 x_3_ - z_v = 51580494

_C3:0 >= -51575889

_C4: - 1e+19 sigma + z_v >= -1e+19

_C5: - 1e+19 sigma + z_v <= 4095

VARIABLES
0 <= sigma <= 1 Integer
0 <= x_0_ <= 1 Integer
0 <= x_1_ <= 1 Integer
0 <= x_2_ <= 1 Integer
0 <= x_3_ <= 1 Integer
z_v Continuous

sigma 1.0
x_0_ 1.0
x_1_ 1.0
x_2_ 0.0
x_3_ 0.0
z_v 8235.0
Optimal
398.0
```

Implementation:

```
    import pulp
# ------------------------ Input parameters -------------------------

N = 10000000000000000000
n = 4
m = 2
M = 1048576
alpha = 7.5
beta = 34
T = 4095
epsilon = 4095

V_u = [4900000,46691204,19950000,20950000]
V_o = [12000000,39579984]
S_u = [148, 148, 148,148]
S_o = [34,34]

# ---------------------- MODEL 1---------------------------------

model = pulp.LpProblem ("Model_1", pulp.LpMinimize)

# ---------------------- Variables in model 1 -----------------------------

z_v = pulp.LpVariable ('z_v', lowBound= 0)

sigma = pulp.LpVariable ('sigma', cat='Binary')

x = {}
for i in range(n):
    x[i] = pulp.LpVariable ('x[%s]' %i, cat= 'Binary')

sizeInput = pulp.lpSum([S_u[i]*x[i] for i in range(n)])
sizeOutput=pulp.lpSum([S_o[j] for j in range(m)])
valueInput=pulp.lpSum([V_u[i]*x[i] for i in range (n)])
valueOutput=pulp.lpSum([V_o[j] for j in range (m)])
fee=alpha*(sizeInput + sizeOutput+beta*sigma)

# ---------------------- Objective function --------------------------

# Optimize transaction size
model +=  sizeInput + sizeOutput+beta*sigma

# ---------------------- Binding ---------------------------

# Transaction size must not exceed the maximum data block size
model += sizeInput + sizeOutput+ beta*sigma <= M

# A transaction must have sufficient consumption value
```

```
model += valueInput ==  valueOutput+fee + z_v

# All trading outputs must be greater than the DUST threshold
model += pulp.lpSum([V_o[j] for j in range (m)]) >= T

# If z_v >= epsilon then z_s = 34, else z_s = 0
model += z_v>=epsilon+0.001-N*(1-sigma)
model += z_v<=epsilon+N*sigma

# -------------------------- OPTIMAL ---------------------------------

model.solve()
print(model)
for v in model.variables():
    print(str(v.name) +" " + str(v.varValue))

print(pulp.LpStatus[model.status])
print (pulp.value(model.objective))

#---------------Model 2-----------

prob = pulp.LpProblem ("Model_2", pulp.LpMaximize)

# Input parameters such as model 1, add only gamma and ymin parameters to determine the maximum s
gamma=0.5
ymin=pulp.value(model.objective)  # Retrieved from the optimal results of model 1

#----------- Variables in Model 2 ----------

z_v = pulp.LpVariable ('z_v', lowBound= 0)

sigma = pulp.LpVariable ('sigma', cat='Binary')

x = {}
for i in range(n):
    x[i] = pulp.LpVariable ('x[%s]' %i, cat= 'Binary')

sizeInput = pulp.lpSum([S_u[i]*x[i] for i in range(n)])

sizeOutput=pulp.lpSum([S_o[j] for j in range(m)])

valueInput=pulp.lpSum([V_u[i]*x[i] for i in range (n)])

valueOutput=pulp.lpSum([V_o[j] for j in range (m)])

fee=alpha*(sizeInput + sizeOutput+beta*sigma)

y=sizeInput + sizeOutput+beta*sigma
```

```
# ---------------------- OBJECTIVE FUNCTION --------------------------

# Find the maximum number of UTXO selected to shrink the size of the original UTXO group

prob += pulp.lpSum([1*x[i] for i in range(n)]) -sigma

# ---------------------- Binding --------------------------

# Transaction size must not exceed the maximum data block size

prob += sizeInput + sizeOutput+ beta*sigma <= M

# A transaction must have sufficient consumption value

prob += valueInput ==  valueOutput+fee + z_v

# All trading outputs must be greater than the DUST threshold

prob += pulp.lpSum([V_o[j] for j in range (m)]) >= T

#Nu z_v >= epsilon th z_s = 34, ngc li th z_s = 0

prob += z_v>=epsilon+0.001-N*(1-sigma)
prob += z_v<=epsilon+N*sigma

# Determine the maximum size to select the maximum number of UTXO

prob += y<=(1+gamma)*ymin

# ------------ OPTIMAL -----------

prob.solve()
print(prob)
for v in prob.variables():
    print(str(v.name) +" " + str(v.varValue))

print(pulp.LpStatus[prob.status])
print (pulp.value(prob.objective))
```

```
Model_2:
MAXIMIZE
-1*sigma + 1*x_0_ + 1*x_1_ + 1*x_2_ + 1*x_3_ + 0
SUBJECT TO
_C1: 34 sigma + 148 x_0_ + 148 x_1_ + 148 x_2_ + 148 x_3_ <= 1048508

_C2: - 255 sigma + 4898890 x_0_ + 46690094 x_1_ + 19948890 x_2_
 + 20948890 x_3_ - z_v = 51580494

_C3:0 >= -51575889

_C4: - 1e+18 sigma + z_v >= -1e+18

_C5: - 1e+18 sigma + z_v <= 4095

_C6: 34 sigma + 148 x_0_ + 148 x_1_ + 148 x_2_ + 148 x_3_ <= 529

VARIABLES
0 <= sigma <= 1 Integer
0 <= x_0_ <= 1 Integer
0 <= x_1_ <= 1 Integer
0 <= x_2_ <= 1 Integer
0 <= x_3_ <= 1 Integer
z_v Continuous

sigma 1.0
x_0_ 0.0
x_1_ 1.0
x_2_ 1.0
x_3_ 1.0
z_v 36007125.0
Optimal
2.0
```

Model 2 result

Experimental Results

Give the coefficient γ of model 2 is 5%, 10%, 20%, 40%, 50%.We also implement two existing methods including HVF and LVF to evaluate and compare performance. Table I summarizes the total transaction size for all three data sets DS1, DS2, and DS3, corresponding to UTXO selection by HVF and Model 1 respectively..The data shows that the UTXO selected by HVFhave better results than the Model 1 .

Table1

EXPERIMENTAL RESULT OF TOTAL TRANSACTION SIZE

Method	DS1	DS2	DS3
HVF	0	54474	30302
Model 1	0	54255	30302

Table II shows that the LVF algorithm is superior to the ability to clean up the UTXO group as much as possible. LVF can create a great deal that definitely costs high for miners. Regarding the Model 2, the results of the solver combine the selection of many UTXOs and the transaction size within an appropriate coefficient, the parameters 40% and 50% give better results, choose a lot of UTXO but also ensure that transaction is generated with adequate size.

Table2
EXPERIMENTAL RESULTS OF THE NUMBER OF SELECTED UTXOS

Method	DS1	DS2	DS3
LVF	0	665	563
Model 2 (5%)	0	242	20
Model 2 (10%)	0	251	21
Model 2 (20%)	0	267	25
Model 2 (40%)	0	271	26
Model 2 (50%)	0	283	28

5 Conclusion

In this article, we have realized the mathematical model for solving two essential goals when creating new transactions on the blockchain. The first model minimizes transaction size so that it can generate a small fee for the mining task responsible for confirming this transaction on the network. The second is designed to control the explosion of the UTXO group by selecting as much as possible the amount of UTXO while maintaining the transaction size to help users pay affordable and appropriate costs. .

References

[1] wikipedia. link: http://en.wikipedia.org/, , last access: 05/05/2015.

[2] Frey, D., Makkes, M. X., Roman, P.-L., Taiani, F., Voulgaris, S.: Bringing secure Bitcoin transactions to your smartphone. The 15th International Workshop on Adaptive and Reflective Middleware, (2016)

[3] Antonopoulos, A. M.: Mastering Bitcoin. 2nd edn. OReilly Media, CA 95472 (2014).

[4] Bitcoinjs: Open Source Organisation for Bitcoin JavaScript Libraries,https://github.com/bitcoinjs. Last accessed 15 August 2018.

[5] Bitcoinj: Library for working with the Bitcoin protocol,https://bitcoinj.github.io. Last accessed 10 August 2018.

[6] Yanovich, Y., Mischenko, P., Ostrovskiy, A.: Shared Send Untangling in Bitcoin, White paper, Bitfury Group Limited (2016).

[7] Dai, P., Mahi, N., Earls, J., Norta, A.: Smart-Contract Value-Transfer Protocols on a Distributed Mobile Application Platform, https://qtum.org/uploads/files/cf6d69348ca50dd985b60425ccf282f3.pdf, (2016).

[8] Sergi, D.-S., Cristina, P.-S., Guillermo, N.-A., Jordi, H.-J.: Analysis of the Bitcoin UTXO set, IACR Cryptology ePrint Archive, (2017)

[9] Erhardt, M.: An Evaluation of Coin Selection Strategies, Master thesis, Karlsruhe Institute of Technology, URL: http://murch.one/wp-content/uploads/2016/11/erhardt2016coinselection.pdf, (2016).

[10] Zahnentferner, J.: Chimeric ledgers: Translating and unifying utxo-based and account-based cryptocurrencies, Cryptology ePrint Archive, Report 2018/262, 2018. https://epri nt. iacr. org/2018/262, (2018).

[11] Chepurnoy, A., Kharin, V., Meshkov, D.: A Systematic Approach To Cryptocurrency Fees. IACR Cryptology ePrint Archive, (2018)