Mathematical models for UTXOs selection

Nguyen Huynh-Tuong[1]

[1]HCMC University of Technology, VNU-HCM, Vietnam
htnguyen@hcmut.edu.vn

***Abstract*—In this work, we need to propose mathematical model for effectively choosing a set of Unspent Transaction Outputs (UTXOs) in transaction-based blockchains in terms of two major objectives. The first one is to minimize the transaction size, as a result, minimize the transaction fee for miners paid by users. The second one is to shrink the UTXO set size that consequently reduces the searching space and computation overhead. We evaluate the proposed models on real transactions collected from Bitcoin network and transactions generated by Highest Value First (HVF) and Lowest Value First (LVF) based approaches. The experimental results show that our proposed models gain better performance compared to other existing approaches.**

***Index Terms*—Blockchain transaction, Unspent Transaction Ouput, transaction size, UTXOs pool, mathematical modeling.**

I. Introduction

A decentralized cryptocurrency is a digital asset using cryptography systems collectively to secure the transactions and integrity without the third party's 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 storage[1].

In blockchain networks, cryptocurrency balances are managed in two different models including account-based and transaction-based models (UTXO model). For transaction-based 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 [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 [10].

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.

The remainder of this paper is organized in the following manners. Section II presents some preliminaries related to transaction and transaction fee. We discuss existing approaches including their benefits and drawbacks in Section 3. Then, we present our work in Section 4. Section 5 summaries the performance evaluation and discussion on experimental results. Finally, we conclude our work in the last section.

II. Related Work

The Bitcoin system is built on a decentralized trust of blockchain technology which is achieved by a mechanism of distributed consensus of all transactions [2]. Each Bitcoin user could use any wallets for managing private keys and credits, and transacting with other users by generating transactions. Each transaction may contain several selected inputs from UTXO pool and outputs for receivers.

For example, Alice wants to pay salary to her twenty employees, so she creates a new transaction that consists of at least twenty outputs and several inputs that are appropriate

[1] https://coinmarketcap.com

UTXOs selected from the UTXO pool. A change output will be returned to Alice if the different amount of inputs and outputs is larger than a dust threshold. Otherwise, this different amount will be added up to the transaction fee for the miner. This means that if the UTXOs selection for inputs is not good, the UTXO pool size will dramatically grow and then may degrade performance in querying information from the blockchain network.

When a transaction is created, then it will connect other Bitcoin clients for propagating out on the peer-to-peer network. However, this transaction is still not included in any block of the blockchain ledger until it is explored by a mining process. Each transaction has enough power computation problem that must be solved by miners before being confirmed as a valid transaction and bundled into a block. If a miner successfully solves this competitive computation for validating a block, he will receive a returned credit so-called coinbase as well as a transaction fee for each transaction.

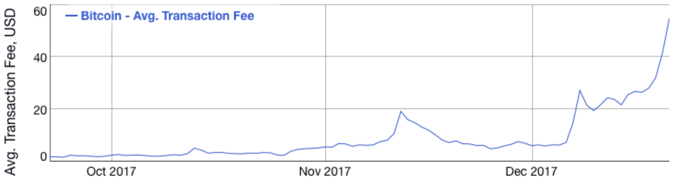

Figure 1. Bitcoin Average Transaction Fee historical chart

2

Some studies were carried out to discover the latent relation between transaction fee and the delay time of a transaction until it is confirmed, proving that the higher fee makes a short time of confirmation. By learning this miner's strategy, some wallets may propose a higher fee to their customers if they want their transactions to be confirmed in a shorter time. Figure 1 shows the average of Bitcoin transaction fee over the last three months of the year 2017[3]. The data shown on the graph describes a significant increase and hit a peak at over $54 per transaction on December. In practical, the following formula describes how to calculate a transaction fee:

$$fee = fee\_rate * (allInputs\_Size(Byte) + \\ allOutputs\_Size(Byte))$$

where fee_rate is the amount of satoshi that can be changed over by time or by transaction trends. However, this fee_rate is usually a constant in a specific period. Additionally, the number of outputs (i.e. $allOutputs\_Size$) is usually deterministic for a given transaction. Therefore, the transaction fee is proportional to $allInputs\_Size$ (i.e. UTXO set size).

Sergi et al. [7] introduced a tool (STATUS) to study and analyze the UTXO set. It is noticeable that the size of UTXO set directly impacts on the storage requirements of a bitcoin node as the way of how to store UTXO set significantly affects to the node's validation speed. In terms of community goals, how to shrink UTXO pool plays an important role which would help query information about UTXO as fast as possible. An essential enhancement from the Bitcoin Core's Release (v0.15) is the internal representation of the chainstate in favor of a better performance both in reading time and memory usage.

Another research on coin selection for a transaction was done by Mark Erhardt [8]. He created a simulation to evaluate the efficiency of UTXO selection strategy on many well-known wallets such as CoreWallet, MyceliumWallet, BreadWallet, AndroidWallet, DoubleWallet, and RandomWallet. After carrying out the simulation, CoreWallet FIFO provided the best of final UTXO set size, and AndroidWallet based on priority achieved the smallest average set size such that could cause the dramatic growth of UTXO. However, total fees of AndroidWallet was the smallest and much better than CoreWallet. So all strategies are trading off between final UTXO set size and total fees.

The library Bitcoinjs under MIT license used by about 15 million bitcoin wallets provides five algorithms for coin selection [3]. The two first ones based on Highest Value First (HFV) mechanism sort UTXO in descending order of value and then select from the head list until reaching the target value. These strategies ensure minimizing the input size at the beginning. However, it could cause the growth of numerous small UTXOs due to the returns of small change outputs. Then it causes to create future transactions with large input sets. The third algorithm is similar to the first one except it does not include the input that has the value greater than the target value. It means that this algorithm only considers smaller inputs for the main objective of shrinking UTXO pool. For the same purpose, another algorithm called Lowest Value First (LVF) chooses the smallest UTXOs until this set could fund for the target value. The two last ones break the input values into equal denominations of output or split the input values evenly between all outputs for optimizing the further usages.

The Bitcoin Core implements Knapsack solution for grouping smaller UTXOs under a given target from a descending value list [4]. This approach has some drawbacks as it tends to choose the largest UTXO that is smaller than the target. This would cause of the forever remaining of many smallest ones in the UTXO pool. Another problem is that it must pay a high fee because it may propose the large transaction with numerous inputs. More importantly, as this process will iteratively run to find the best solution, it will have expensive computation and therefore remarkably influence the responsiveness of a real-time wallet system.

III. Our Work

A. Problem Description

For a given transaction, the wallet will choose some UTXOs from UTXO pool such that has sufficient value for funding called the target value. The best selection is to have an exact match with the target because it will not generate change

[3] https://bitinfocharts.com/comparison/bitcoin-transactionfees.html

output returning to the sender for minimizing the transaction size as well as not making the UTXO pool size exploded. The change output is the remaining amount after funding and must be larger than a dust threshold that is the output of a transaction in which the fee to redeem it is greater than 1/3 of its value. The objective of the dust is to prevent spam transactions when someone tries to degrade the network by intentionally generating very small transactions that may consume a large bandwidth. Additionally, the Bitcoin and other similar systems currently charge for each transaction to discourage bad behaviors and ensure only valid transactions relying on the blockchain network.

Our objective is to propose an efficient strategy of choosing an appropriate set of UTXOs for a given transaction which is likely to satisfy multiple constraints such as (1) minimizing the transaction size for having a minimum transaction fee and (2) shrinking the UTXO pool. It is noticeable to realize that our proposal explicitly brings benefits to users and community goals. Additionally, our proposed strategy would like transactions to be confirmed as fast as possible by using a suitable fee rate depending on user's demand. This is an implicit benefit to miners.

Strategy 1 Proposed UTXO Selection

Objective Determine a subset of affordable UTXOs such that satisfy multiple constraints including hard constraints H_1 and soft constraints S_1.

Input
U : a set of UTXOs
a : output amount for sending to transaction receivers
α : fee rate for mining
Other parameters are summarized in Table I

Output

- A set of chosen UTXO which may contain only exact match singleton.
- A possible change output.

Hard constraints H_1

1) A transaction must have sufficient value for consuming.
2) A transaction size may not exceed maximum block data size.
3) All the transaction outputs must be higher than the dust threshold to certain that this transaction is relayed to the network and confirmed.

Soft constraints are convincing to be considered S_1

1) The transaction size is minimized.
2) The number of selected UTXOs is maximized to shrink the UTXO pool size.

Table I
ALL INPUT PARAMETERS OF OUR DEFINED WORK

Input Parameters	**Description**
$U = \{u_1, \ldots, u_n\}$	set of UTXOs
$O = \{o_1, \ldots o_m\}$	set of transaction outputs
$V^u = \{v^u_1, \ldots v^u_n\}$	set of UTXO's values
$V^o = \{v^o_1, \ldots v^o_m\}$	set of transaction output's values
$S^u = \{s^u_1, \ldots, s^u_n\}$	set of transaction input size, with input is chosen from UTXO u_i
$S^o = \{s^o_1, \ldots, s^o_m\}$	set of transaction output's size.
M	maximum size of a transaction
α	fee rate
T	dust threshold
ε	minimum of change output that is set to avoid creating a very small output

B. Mathematical Modeling

The above-proposed UTXO selection strategy is mathematically modeled by two different models, **Model 1** and **Model 2**. Each model is represented in term of Variables, Intermediate Variables, Constraints, and Objective Function. The **Model 1** is to minimize the transaction fee as follows.

1) Model 1 - Variables: Decision variable:

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

2) Model 1 - Intermediate variables:

- y: transaction size
- z_v: a value of change output (amount)
- z_s: size of change output

$$z_s = \begin{cases} 0, & 0 \leq z_v \leq \varepsilon \\ \beta, & z_v > \varepsilon \end{cases} \tag{2}$$

3) 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

$$y = \sum_{i|u_i \in U} s^u_i * x_i + \sum_{j|o_j \in O} s^o_j + z_s \leq M \tag{3}$$

- A transaction must have sufficient value for consuming.

$$\sum_{i|u_i \in U} v^u_i * x_i = \sum_{j|o_j \in O} v^o_j + \alpha * y + z_v \tag{4}$$

- All the transaction outputs must be higher than the dust threshold to certain that this transaction is relayed to the network and confirmed.

$$T \leq \sum_{j|o_j \in O} v^o_j \tag{5}$$

- The relation between change output value z_v and its size z_s is defined as follow.

$$z_s \leq \left\lfloor \frac{z_v}{\varepsilon} \right\rfloor * \beta \tag{6}$$

If $z_v \leq \varepsilon$, z_s should be zero; otherwise, z_s should be equal to β

- x_i is binary variable

$$\forall i | u_i \in U : x_i \in \{0, 1\} \tag{7}$$

4) Model 1 - Objective function: Minimizing transaction size

$$\text{minimize} \quad y \tag{8}$$

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.

5) Model 2 - Variables: include all variables of **Model 1**

6) Model 2 - Constraints: include all constraints of **Model 1** and an extra constraint as follow :

$$y < (1 + \gamma) \times Y, \tag{9}$$

- Y is the minimal transaction size obtained by Model 1.
- γ: is a coefficient $(0 < \gamma < 1)$

If γ 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.

7) Objective function: Maximizing the number of UTXOs

$$\text{maximize} \quad (\sum_{i|u_i \in U} x_i - z_s/\beta) \tag{10}$$

IV. Experiments

A. Dataset Collection

A dataset was crawled from May $14th$ 2018 to May $19th$ 2018 of BTC UTXOs. After pre-processing, the size of the dataset is 13055 instances. We developed a built-in back-end system for crawling new transactions from a peer-to-peer network and recording UTXOs which will be used to create transactions. Observing this dataset helps us know how UTXOs are chosen on the network. Figure 2 shows the frequency of UTXO for each address in UTXO pool. The highest column illustrates that Bitcoin addresses having 1 UTXO occupied mostly the dataset, e.g. there are 10611 instances in total 13055 instances (about 81.28%). The number of instances having UTXO from 2 to 1000 is 2235 (about 17.12%) and those having more than 1000 UTXO is 209 (about 1.6%). Therefore, we created 3 datasets: **DS1** contains full of 13055 instances, **DS2** has 2235 instances with the number of UTXO from 2 to 1000, and **DS3** has 209 instances having UTXO greater than 1000.

B. Experimental Results

Our proposed **Model 1** and **Model 2** are solved by GUROBI 5.1.0. The γ coefficient of Model 2 is 5%, 10%, 20%, 40%, 50%. We also implement two existing methods including HVF and LVF for performance evaluation and comparison.

Table II summarizes the total transaction size for all three dataset DS1, DS2, D3 corresponding to UTXO selection by real transactions, HVF and Model 1 respectively. Additionally, Figure 3 shows the difference of transaction size of real transactions and others generated by HVF, Model 1. The data represents that the UTXOs chosen by HVF and solver on Model 1 give better results than those of real transactions in term of transaction size. When comparing all three strategies of choosing UTXO, all Bitcoin addresses having only 1 UTXO (about 81.28%) have the same choice because they have only one selection. For all addresses having the number of UTXO greater than 1, that result obtained from Model 1 is better than HVF.

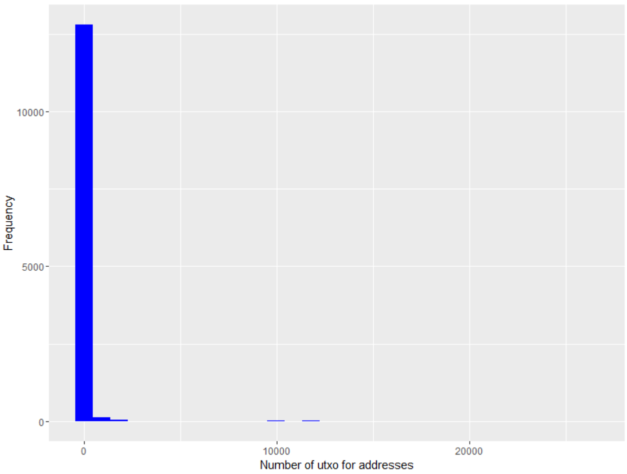

Figure 2. Frequency of UTXO in UTXO pool

Table II
Experimental result of total transaction size

Method	**DS1**	**DS2**	**DS3**
Real Transaction	17317872	4330028	784672
HVF	17228518	4272714	752632
Model 1	17218197	4264129	750896

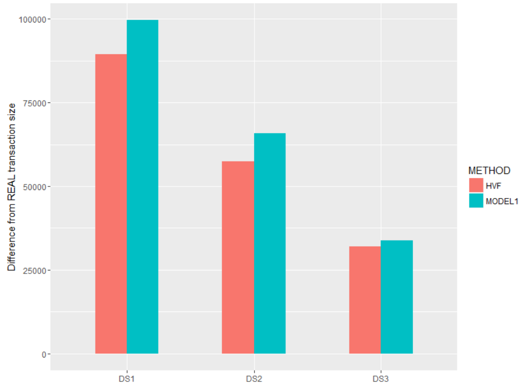

Figure 3. Performance comparison in terms of transaction size

Table III represents that the LVF algorithm outperforms the ability to clean up UTXO pool as much as possible, but it has a side effect on transaction size that can be seen on the Table IV. LVF may produce a huge transaction which certainly costs a high fee 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, i.e., choose a lot of UTXO but also ensure that transaction is generated with adequate size.

Table III
EXPERIMENTAL RESULTS OF THE NUMBER OF SELECTED UTXOS

Method	**DS1**	**DS2**	**DS3**
Real Transaction	17059	5717	731
LVF	**513987**	23470	479906
Model 2 (γ = 5%)	16426	5286	529
Model 2 (γ = 10%)	16489	5320	558
Model 2 (γ = 20%)	16654	5406	637
Model 2 (γ = 40%)	17004	5608	785
Model 2 (γ = 50%)	17273	5807	855

Table IV
PERFORMANCE COMPARISON IN TERMS OF TRANSACTION SIZE (IN BYTES)

Method	**DS1**	**DS2**	**DS3**
Real Transaction	17317872	4330028	784672
LVF	**90865154**	6959648	71702334
Model 2 (γ = 5%)	17226596	4268179	755245
Model 2 (γ = 10%)	17235501	4272984	759345
Model 2 (γ = 20%)	17260814	4286764	770878
Model 2 (γ = 40%)	17313888	4318482	792234
Model 2 (γ = 50%)	17356251	4349849	803230

V. CONCLUSION

In this paper, we proposed two mathematical models for tackling two essential objectives when creating a new transaction on blockchains. The first model minimizes the transaction size so that it could produce a small fee for mining task which is responsible for validating this transaction on the network. The second one is constructed to restraint the explosion of UTXO pool by selecting as much as possible the number of UTXOs while maintaining the transaction size to help user paying the affordable and suitable cost. Our experiments have shown better results compared to current real transactions, HVF, and LVF strategies. Although our proposed models are applicable in realistic, we need to carry out further experiments on a bigger dataset for measuring runtime and resource consuming which are also two important factors, especially for deployment in mobile devices.

ACKNOWLEDGMENT

We would like to thank Infinity Blockchain Labs Co., LTD[4] for supporting and funding this research.

[4] http://www.blockchainlabs.asia

REFERENCES

[1] 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).

[2] Antonopoulos, A. M.: Mastering Bitcoin. 2nd edn. O'Reilly Media, CA 95472 (2014).

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

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

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

[6] 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).

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

[8] 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).

[9] 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).

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