Optimal Resource Allocation in Edge Computing for Mobile Blockchain by Genetic Algorithm

Pihe Hu
Department of Computer Science and Engineering
Shanghai Jiao Tong University

***Abstract*—Blockchain is a trending technique these days. Recently, more and more blockchain network has been deployed in mobile network. In terms of bitcoin, network nodes need to solve proof-of-work (PoW) puzzles to ensure the consensus of the ledger. However, the computation power required for solving PoW puzzles is not feasible for mobile devices in most cases. In this paper, we consider deploying edge computing service to enable the mobile blockchain. We propose optimal resource allocation scheme to maximize the network profit. We first formulate our profit maximization problem as a nonlinear integer programming problem. To solve it, we propose an algorithm inspired by genetic algorithm. Through numerical simulation, we verify that our proposed algorithm can converges to optimal solution very quickly with a low time complexity. Besides, we prove that our proposed resource allocation scheme can achieve the maximum network profit efficiently and outperform than traditional greedy algorithm.**

***Index Terms*—mobile blockchain, edge computing, resource allocation, genetic algorithm.**

I. Introduction

Blockchain was invented by Satoshi Nakamoto in 2008 [1] to serve as the public transaction ledger of the cryptocurrency bitcoin. The invention of the blockchain for bitcoin made it the first digital currency to solve the double-spending problem without the need of a trusted authority or central server. Bitcoin is an electronic asset, whose reliability depends on the hardness for mining a block by solving PoW puzzles. The traditional blockchain network is deployed in wired Ethernet, which provide reliable connection, as well as low delay and high bandwidth. However, heavy equipments and fixed access point has become shortcomings of traditional blockchain network, which limits its further development. Besides, with the development of wireless communication technology, the mobile blockchain network (MBN) is more and more often.

MBN [2] is a blockchain network deployed in mobile devices. Together they create a powerful second-level network, a wholly different vision for how the Internet can function. Every node is an "administrator" of the blockchain, and joins the network voluntarily. However, each one has an incentive for participating in the network: the chance of winning Bitcoins. Nodes are said to be "mining" Bitcoin, but the term is something of a misnomer. In fact, each one is competing to win Bitcoins by solving computational puzzles. The mining process is conducted in a tournament structure, and miners chase each other to obtain the solution. A proof of work is a piece of data which is difficult (costly, time-consuming) to produce but easy for others to verify and which satisfies certain requirements. Producing a proof of work can be a random process with low probability so that a lot of trial and error is required on average before a valid proof of work is generated. Bitcoin uses the Hashcash proof of work system.

Mobile Edge Computing (MEC) [3] is a network architecture concept that enables cloud computing capabilities and an IT service environment at the edge of any network. The basic idea behind MEC is that by running applications and performing related processing tasks closer to the cellular customer, network congestion is reduced and applications perform better. MEC technology is designed to be implemented at the cellular base stations or other edge nodes, and enables flexible and rapid deployment of new applications and services for customers.

Deploying MBN in MEC is a feasible way to solve the low-computing-power dilemma. Specifically, the PoW algorithm involves finding a nonce value that, together with additional fields about all valid and received transactions, the previous block and the timestamp, the output satisfies a given condition. If the nonce is found, the miner will combine it and additional fields into a block and then broadcast the block to peers in the blockchain network for verification and reaching consensus. Finally, the new block can be linked to the existing accepted chain of blocks. However, for a mobile user, it is unrealistic to continuously run such a computationally difficult program which requires high computing power and consumes a large volume of energy and time. Because the outstanding characteristics of edge computing [4]: low latency, mobility and wide-spread geographical distribution, we consider offloading the mining tasks to the ESPs.

In this paper, we propose optimal resource allocation scheme to maximize the network profit. We regard the MEC and MBN as a heterogeneous network, which is consisted of ESPs, MBN nodes (i.e. the bitcoin miners). The system profit is defined as the total expected reward of all miners minus MEC cost of computing hardwares. The network profit maximization problem is formulated as a nonlinear integer programming problems, which is shown in Section III. To solve it, we propose an algorithm inspired by genetic algorithm in Section IV. We conduct numerical simulation in Section V. Through numerical simulation, we find our proposed algorithm can converges to optimal solution very quickly while preserving a low time complexity. Besides, we find our proposed resource allocation scheme can achieve the maximum system profit efficiently and outperform than the traditional greedy algorithm.

II. Related Work

Basically, the related work of this paper is about the resource allocation scheme in MEC-aided MBN. In [5], the authors formulate a two-stage Stackelberg game to jointly maximize the profit of the edge computing service provider and the individual utilities of the miners. They apply the backward induction to analyze the sub-game perfect equilibrium in each stage for both uniform and discriminatory pricing schemes. Then in [6], the authors develop an optimal auction based on deep learning for the edge resource allocation. Specifically, they construct a multi-layer neural network architecture based on an analytical solution of the optimal auction. They use valuations of the miners as the data training to adjust parameters of the neural networks so as to optimize the loss function which is the expected, negated revenue of the Edge Computing Service Provider. Besides, in [7], the authors propose an auction-based edge computing resource market of the edge computing service provider, where they maximize the social welfare while guaranteeing the truthfulness, individual rationality and computational efficiency. In fact, our will follow a similar system model as that in [7].

To sum up, the present schemes are mainly based on two methodologies. The first one is pricing model based on Game Theory. The second one is auction model based on deep learning auction theory, which can be found in [8] and [9], respectively. However, our work may be different from these. We obtain the optimal solution by solving nonlinear integer programming (NIP) problems based on a genetic algorithm. It is worth to mention that in [15], the authors propose a method for solving the nonlinear integer programming (NIP) problem for the best compromise solution while holding a nonlinear property by using the genetic algorithm. Our proposed algorithm is exactly inspired by [15].

III. System Model

Figure 1 shows the system model of MEC-aided MB-N, which includes single ESP (possesses heavy equipments offering computing power) and N mobile devices running blockchain program. Every time, when mobile users try to solve PoW puzzles, they send request to ESP for computing power with specific resource demands. After receiving the request from all users, the ESP will make a decision of resource allocation scheme. The mobile users are exactly miners in blockchain. In the miners network, they take part in the mining process to contribute new blocks to the blockchain. Trough mining, the miners are expected to received rewards, which contributes to the network profit. The profit is exactly total expected reward from miners minus hardware cost at the ESP.

A. Assumptions

In this model, we assume that message transmission time can be neglected compared to computing time. Besides, all mobile users send request to ESP at the same time. In fact, we only consider the resource allocation problem in one round of resource request in this paper. Besides, we assume that the resource allocation scheme is totally determined by the ESP, who has the knowledge of the global network information, with the goal to maximize the network profit. Moreover, the resource in ESP can be uniformly quantified by numbers.

As shown in Figure 1, we consider a scenario where there is one ESP and a community of mobile users $\mathcal{U} = \{1, \cdots, N\}$. Each mobile user wants to be a miner, who runs a mobile blockchain application to record and verify the transactions or data sent to the community. Due to the computing limitation on their devices, mobile users want to offload the task of solving PoW to the nearby edge computing servers deployed by the ESP. In particular, the ESP is responsible for dealing with the request from mobile users, so as to make a decesion of resource allocation.

At the first place, the mobile user i will send request to ESP with resource demand d_i, for $i = 1, \cdots, N$. For the ESP, the resource demands from all mobile users is denoted as a vector $\mathbf{d} = (d1, \cdots, d_N)$. Besides, user i is dedicated at mining a block of size s_i, for $i = 1, \cdots, N$. After receiving the demands, the ESP will make the decision on resource allocation scheme. It will select the some mobile users as the successful miners and notifies the all mobile users the allocation scheme $\mathbf{x} = (x_1, \cdots, x_N)$, where binary variable $x_i = 1$ means user i's resource demand d_i has been approved, otherwise disapproved for $x_i = 0$. After the resource allocation scheme is informed to all mobile users throughly. The mobile users granted to use the resource will upload their tasks to ESP. All tasks from mobile users are run in parallel. Once a user's task has solved the PoW puzzle, the ESP will stop all tasks immediately and announce the result to all mobile users. Then a new round of resource allocation will begin.

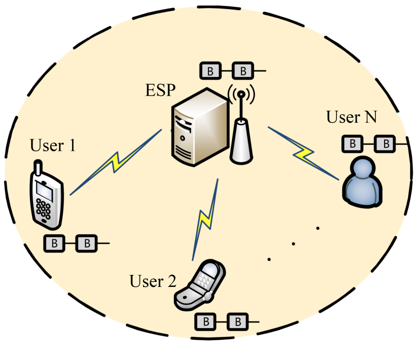

Fig. 1. System model of MEC-aided MBN

B. MEC-aided MBN

With the allocation x_i and demand d_i, miner is hash power h_i relative to other miners allocated resources can be calculated by:

$$h_i = \frac{d_i^\alpha x_i}{\sum_{j \in \mathcal{U}} d_j^\alpha x_j} \tag{1}$$

which is a fraction function that $\sum_{j\in\mathcal{U}} h_i = 1$. α is the curve fitting parameter of the hash power function h_i verified by real-world experiment [10]. In the mining tournament, miners compete to be the first to solve PoW with correct nonce value and propagate the block to reach consensus. The generation of new blocks follows a Poisson process with a constant rate $1/\lambda$ throughout the whole blockchain network [11]. Before the tournament, miners collect unconfirmed transactions into their blocks. When miner i propagates its block to the mobile blockchain network for consensus, the time for verifying each transaction is affected by the size of transactions s_i. The first miner which successfully has its block achieve consensus can get a reward R_i. The reward is composed of a fixed bonus T for mining a new block and a flexible transaction fee t determined by the size of its collected transactions s and the transaction fee rate r [12]. Thus, miner is expected reward R_i can be expressed by:

$$R_i = (T + rs_i)P_i \tag{2}$$

where $P_i(h_i, s_i)$ is the probability that miner i receives the reward by contributing a block to the blockchain.

From the mining tournament above, winning the reward depends on the successful mining and instant propagation. The probability of mining a new block P_i^s is equal to miner is hash power h_i, i.e., $P_i^s = h_i$. However, the miner may even lose the tournament if its new block does not achieve consensus as the first. This kind of mined block that cannot be added on to the blockchain is called orphaned block [12]. Moreover, the block containing larger size of transactions has higher chance becoming orphaned. This is because a larger block needs more propagation time, thus causing higher delay for consensus. Here, we assume miner is block propagation time τ_i is linear to the size of transactions in its block, i.e., $\tau_i = \xi s_i$, where ξ is a constant that reflects the impact of s_i on τ_i. Since the arrival of new blocks follows a Poisson distribution, miner is orphaning probability can be approximated as follows [13]:

$$P_i^o = 1 - \exp(-\frac{\tau_i}{\lambda}) \tag{3}$$

After substituting τ_i, we can expressP_i as follows:

$$P_i = P_i^s(1 - P_i^o) = h_i \exp(-\frac{\tau_i}{\lambda}) \tag{4}$$

A blockchain in PoW systems is only as secure as the amount of computing power dedicated to mining it. This results in positive network effects: as more mobile users participate in mining and more computing resources are invested, the value of reward given to miners increases since the blockchain network is more stable and secure. Empirically, we define the network effects by a common S-shaped utility function [14]:

$$\rho = \frac{1 - e^{-\nu d_{\mathcal{U}}}}{1 + \mu e^{-\nu d_{\mathcal{U}}}} \tag{5}$$

where $d_{\mathcal{U}} = \sum_{i\in\mathcal{U}} d_i$ is the total quantity of allocated resources and μ, ν are positive parameters. The monotonic increase of network effect function begins slowly from 0, then accelerates (convexly), and then eventually slows down (concavely) and converges asymptotically to 1.

IV. Profit Maximization Scheme

In this section, we propose the optimal resource allocation scheme for the ESP to allocate edge computing resources efficiently. We focus on maximizing the network profit while guaranteeing the fast convergence and low time complexity of the proposed algorithm.

A. Problem Formulation

Once receiving resource demands from all mobile users, ESP will make a decision on resource allocation scheme to maximize the network profit. Just as mentioned above, the network is the total rewards from all mobile users minus equipment cost at the ESP. However, the network effect in Eq. (5) should be taken in consideration. Thus, the network profit is given by

$$f(\mathbf{x}) = \sum_{i\in\mathcal{U}} wR_iP_i - \sum_{i\in\mathcal{U}} pd_ix_i \tag{6}$$

where p is the unit equipment cost in ESP. Furthermore, we can formulate the network profit maximization problem $\mathcal{P}$ as below:

$$\begin{aligned} \max_{\mathbf{x}} & \sum_{i\in\mathcal{U}} \frac{d_i^\alpha x_i}{\sum_{j\in\mathcal{U}} d_j^\alpha x_j} \frac{1 - e^{-\nu d_{\mathcal{U}}}}{1 + \mu e^{-\nu d_{\mathcal{U}}}} (T + rs_i) e^{-\frac{\tau_i}{\lambda}} \qquad (7) \\ & - \sum_{i\in\mathcal{U}} pd_ix_i \\ s.t. \quad & \text{C1} \quad \sum_{i\in\mathcal{U}} d_ix_i \le C \\ & \text{C2} \quad x_i \in \{0, 1\}, \forall i \in \mathcal{U} \end{aligned}$$

The constraint C1 defines the maximum quantity of computing resources that ESP can offer denoted by C, while constraint C2 ensures x_i is a binary variable. Generally, problem $\mathcal{P}$ is a integer programing problem with linear constraint and nonlinear objective function. Thus, $\mathcal{P}$ is a NIP problem, which belongs to NP-complete problems. There is no polynomial time algorithm to solve it. However, in the next section, we will propose a low complexity algorithm to solve it, based on genetic algorithm.

B. Optimal Solution by Genetic Algorithm

In computer science and operations research, a genetic algorithm (GA) [16] is a metaheuristic inspired by the process of natural selection that belongs to the larger class of evolutionary algorithms (EA). Genetic algorithms are commonly used to generate high-quality solutions to optimization and search problems by relying on bio-inspired operators such as mutation, crossover and selection.

In general, it is difficult to directly solve an optimization problem of systems reliability formulated as a NIP model. But we aim to solve this integer programming efficiently. In fact, we are inspired by genetic algorithm, and follow a similar procedure to solve problem $\mathcal{P}$ as that in [15]. We introduce a method based on genetic algo- rithm for solving a NIP problem as follows:

step 1: The parameters are setup as follows:

pop_size:	population size
p_c:	probability of crossover
p_m:	probability of mutation
$maxgen$:	maximum generation
$gen = 0$:	initial generation
$maxeval = 0$:	value of evaluation function

step 2: We generate the initial chromosome $\mathbf{v}_k(k = 1, \cdots, pop\_size)$ at random with N elements.

$$\mathbf{v}_k = [x_{k1}x_{k2}\cdots x_{kN}]$$

When generating the chromosomes, it satisfied the following conditions:

a) The element of each chromosome should be a binary variable,

$$x_{ki} \in \{0,1\} \quad \forall k \in \{1, \cdots, pop\_size\}, \forall i \in \mathcal{U}$$

b) Each chromosome satisfy the constraint C1 as follows:

$$\sum_{i \in \mathcal{U}} x_{ki} d_i \le C \quad \forall k \in \{1, \cdots, pop\_size\}$$

When we can not find a chromosome to satisfy the conditions, there was no feasible solution and the calculations were stopped.

step 3: We set up $gen = gen + 1$, calculate the evaluation function $eval(\mathbf{v}_k)$.

$$eval(\mathbf{v}_k) = f(\mathbf{v}_k), k = 1, 2, \cdots, pop\_size.$$

step 4:
- Crossover: Let the number of chromosome generated by crossover be $ccut = 0$. Create the random number $r_k(k = 1, 2, \cdots, pop\_size)$ from the range $[0, 1]$. Select the $\mathbf{v}_k$ that satisfy $r_k < p_c$,. Make a pair of $\mathbf{v}_k$ and set $ccnt = ccnt + 2$. Choose the position for crossover at random and undergo crossover. Let the chromosome that is newly generated be $\mathbf{v'}_{cnnt-1}$ and $\mathbf{v'}_{cnnt}$.
- Mutation: Let the number of chromosome generated by mutation be $mcnt = 0$. Create the random number $r_k(k = 1, 2, \cdots, n * pop\_size + n * ccnt)$ from the range $[0, 1]$. Select a gene that satisfy $r_k < p_m$ and set $mcnt = mcnt + 1$. In a case of the mutation in the same chromosome, it does not increase the value of $mcnt$. Mutate x_{kj} that is selected in $\{0, 1\}$. Let the chromosome that is newly generated be $\mathbf{v}'_{ccnt+mcnt}$.
- Selection: Calculate the evaluation function $eval(\mathbf{v'}_t), t = 1, 2, \cdots, ccnt + mcnt$ that is newly generated. Add the penalty for the chromosome that does not satisfy the system constraint. Select the chromosomes among the parents $\mathbf{v}_k(k = 1, 2, \cdots, pop\_size)$ and newly generated offsprings $\mathbf{v}_t(t = 1, 2, \cdots, ccnt + mcnt)$ in the order that is superior to others. The number to be selected is pop_size and let these chromosomes be the $\mathbf{v}_k$ for the next generation in which the chromosome that has the same structure is not selected.

step 5: If $maxeval < max\{eval(\mathbf{v}_k)\}$ then

$$\begin{aligned} \mathbf{v}^* &= \arg\max_{\mathbf{v}_k}\{eval(\mathbf{v}_k)\} \\ maxeval &= max\{eval(\mathbf{v}_k)\} \end{aligned}$$

step 6: If $gen < genmax$ then goto Step 3. If $gen = genmax$, then output $\mathbf{v}^*$ and stop.

Theorem 1. *The time complexity of proposed algorithm is* $O(maxgen * pop\_size \log(pop\_size)$

Proof: Consider the algorithm execution of one generation. Initially, the time complexity for crossover and mutation is $O(pop\_size)$. Second, the time complexity for selection is $pop\_size \log(pop\_size)$ because a sort for the values of evaluation functions of different individuals is needed. Then, the time complexity for one generation is $pop\_size \log(pop\_size)$. Besides, the time complexity for initialization is $O(pop\_size)$. Thus, the total time complexity is $O(maxgen * pop\_size \log(pop\_size)$. ■

V. Experiment Results

In this section, we conduct numerical simulation to verify the effectiveness of our proposed algorithm, as well as the performance of our proposed resource allocation scheme. We will first show the convergence pross of our proposed scheme under various crossover, mutation probabilities and population size, then the result of the proposed resource allocation strategy with a traditional greedy algorithm. Our settings for the parameters during the experiment is shown in table I.

Parameter	Value
N	$50 \sim 2000$
C	$25 \sim 1250$
α	1.2
μ	5
ν	0.005
T	2.5
r	0.007
λ	$200 \sim 2000$
ξ	1
p	0.002
s_i	$U[0, 1000]$
d_i	$1 \sim 11$
pop_size:	population size
p_c:	probability of crossover
p_m:	probability of mutation
$maxgen$:	maximum generation
$gen = 0$:	initial generation
$maxeval = 0$:	value of evaluation function

TABLE I
Parameter Settings

A. Performance of the Proposed Algorithm

In general, finding the optimal solution to a NIP problem is equivalent to conduct a search in the variable space. Genetic algorithm provides a kind of heuristic search, which is very likely to avoid falling into local optimum, while guaranteeing a fast convergence speed. Thus, in the following, we will conduct numerical experiment to verify our algorithm in terms of convergence speed and probability of falling into local optimum. Here, we set $N = 400, C = 200$ and $\lambda = 600$.

1) Convergence process under various crossover probability: We fixed $p_m = 0.1, pop\_size = 50$ and $maxgen = 100$. We try to compare the convergence speed of our algorithm under different crossover probabilities, which is shown in Figure 2.

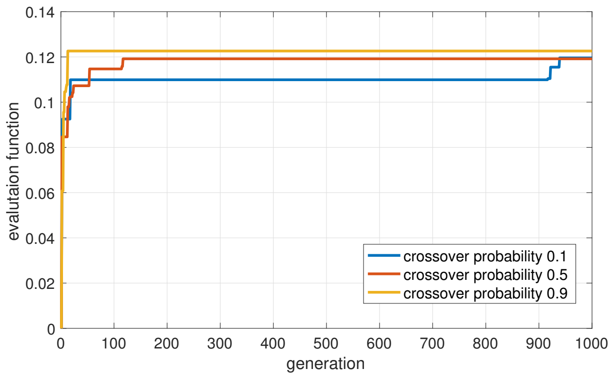

Fig. 2. Convergence process of crossover with probability 0.1, 0.5 and 0.9

It can be observed from the figure that three times of execution of the algorithm all convergent. However, the higher the crossover probability, the faster convergence speed of the algorithm. Indeed, high crossover probability avoids the search falling into local optimum.

2) Convergence process under various mutation probability: We fixed $p_c = 0.1, pop\_size = 50$ and $maxgen = 100$. We try to compare the convergence speed of our algorithm under different mutation probabilities, which is shown in Figure 3.

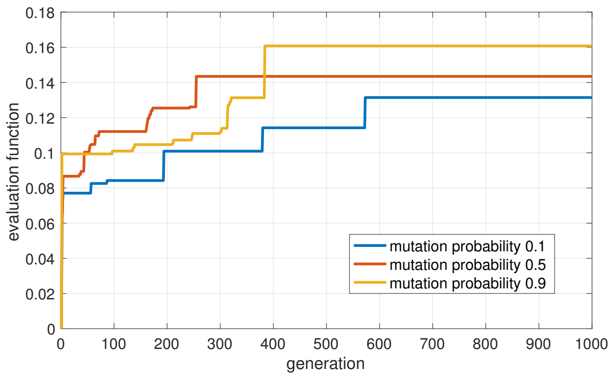

Fig. 3. Convergence process of mutation with probability 0.1, 0.5 and 0.9

From the figure above, we find that though three executions of the algorithm all convergent, convergence process under mutation probability of 0.1 and 0.5 fall into local optimum. In fact, we cannot conclude that convergence process under mutation probability of 0.9 convergent to global optimum. However, one thing is for sure that a higher mutation probability will ensure the algorithm falling into local optimum at a lower probability. Besides, in terms of convergence speed, we find that algorithm with mutation probability 0.5 has the fastest convergence speed, and then 0.9 and 0.1. Thus, we claim that there exist an optimal mutation probability that has the fastest convergence speed between 0.1 and 0.9. In other words, there is a tradeoff between convergence speed and probability of falling into local optimum.

3) Convergence process under various population size: We fixed $p_c = 0.5, p_m = 0.5$ and $maxgen = 1000$. We try to compare the convergence speed of our algorithm under different population size probability, which is shown in Figure 4.

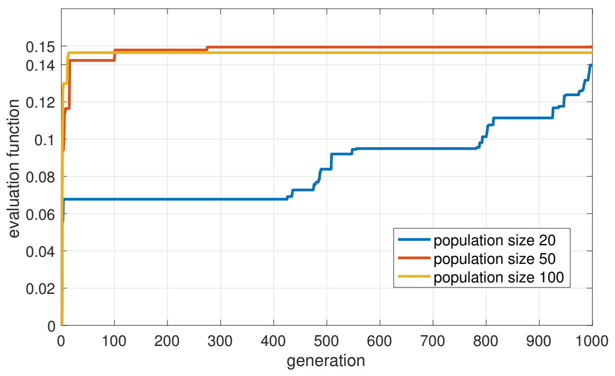

Fig. 4. Convergence process of population size with 20, 50, and 100

It can be observed from the figure that three times of execution of the algorithm all convergent. Similarly as that in 1), our insight is that the larger population size, the faster convergence speed and lower probability into local optimum. In fact, the algorithm under population size of 20 has an awful performance.

B. Simulation result of Optimal Resource Allocation

We vary the number of mobiles users N from 100 to 1000, and the averaging time λ for successfully mining a block, from 200 to 2000. The transaction size s of each user is uniformly distributed over [0,1000]. Here, we set $p_c = 0.5, p_m = 0.5, pop\_size = 40, maxgen = 100$ and $C = N/2$.

1) Impact of the number of mobile users N: Figure 5 shows the impact of the total number of mobile users N on the network profit and the number of selected users. We fix $\lambda = 600$. We observe that network profit and selected user number increase at diminishing rate as the base of mobile users becomes larger. Naturally, the ESP can select more users as miners to increase the network profit with more available resources. However, at the same time, the negative effects from the competition among a larger number of miners are apparent, which slows down the rise of the network profit as well as the number of assigned users.

2) Impact of the average time λ for successfully mining a block: In Figure 6, we fix $N = 400$. When the blockchain owner raises the difficulty of mining a block, represented by

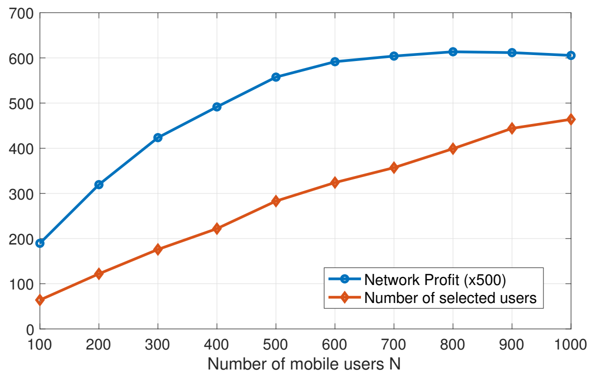

Fig. 5. Impact of the number of mobile users on network profit and number of selected users

λ, the network profit increases while the number of assigned mobile users almost remains the same. Note that the users expected reward R_i grows with increasing λ. When the difficulty λ is small and each user's reward is small, the ESP has to accept more users to maximize the network profit. However, if the difficulty of mining a block becomes high and each user gets reward more, the ESP will still select enough users to achieve the optimal network profit. Besides, one reason for not selecting more users is the increasingly intense competition among them. All these factor influence each together and the selected users will almost remain the same while increasing the network profit.

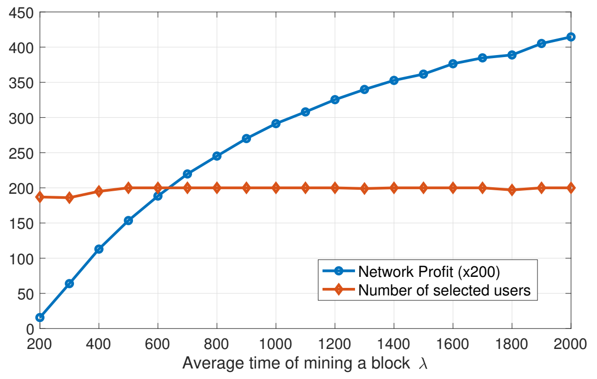

Fig. 6. Impact of the parameter λ

3) Comparison with greedy algorithm: To further evaluate our proposed genetic algorithm, we conduct a simulation to compare the performance of our proposed algorithm with a traditional greedy algorithm. In this greedy algorithm, the ESP will always select the user with maximum single profit (i.e. $\frac{1-e^{-\nu}}{1+\mu e^{-\nu}}(T + rs_i)e^{-\frac{\tau_i}{\lambda}}$) first. It continues in this fashion of selection until the present resource cannot satisfies any more users. In fact, the greedy algorithm is simple and widely used in practice. It is worth to mention that the greedy algorithm is supposed to find the optimal solution in some special cases, such as symmetrical system.

We vary the number of users N from 50 to 500, such that the network profit and selected user number of two algorithm are shown in Figure 7. Here, the user demands are integers uniformly distributed in $[1, 11]$.

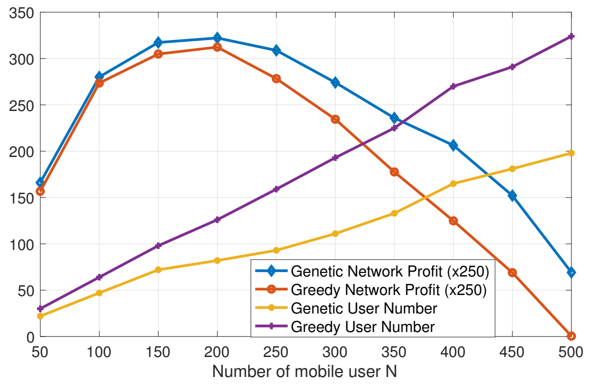

Fig. 7. Comparison of genetic algorithm and greedy algorithm

The result is consistent with our expectation. We find network profit of proposed genetic algorithm is larger than the greedy algorithm and the difference gets larger when user number increases. Interestingly, we find that the selected user number of proposed genetic algorithm is no more than the greedy user number, which means the optimal solution is not always tends to select more user. The underlying reason is that the more users, the fiercer competition and less probability to succeed in mining a block.

VI. Conclusion

In this paper, we consider the resource allocation in MEC-adied MBN. We propose the optimal resource allocation scheme to maximize the network profit. We first formulate our profit maximization problem as a nonlinear integer programming problems. To solve it, we propose an algorithm inspired by genetic algorithm. Through numerical simulation, we find that our proposed algorithm can converges to optimal solution very quickly while preserving a low time complexity. Besides, our proposed resource allocation scheme can achieve the maximum network profit efficiently.

Acknowledgment

Initially, please allow me to express my sincere thanks to teacher Fu. Luoyi, for her help during the course project, as well as her kindness and understanding for postponing my deadline of course project due to my personal reason. Thank you, again.

Besides, I need to say thank you to the authors in [7], on whose work my system model is based. Moreover, thank you to authors in [15] as well, whose work inspired me a lot in solving the NIP problems.

References

[1] Crosby, Michael, et al. ”Blockchain technology: Beyond bitcoin.” Applied Innovation 2 (2016): 6-10.
[2] Suankaewmanee, Kongrath, et al. ”Performance analysis and application of mobile blockchain.” arXiv preprint arXiv:1712.03659 (2017).

[3] Flinck, Hannu. ”Mobile Edge Computing.” (2016).
[4] Yi, Shanhe, Cheng Li, and Qun Li. ”A survey of fog computing: concepts, applications and issues.” Proceedings of the 2015 Workshop on Mobile Big Data. ACM, 2015.
[5] Xiong, Zehui, et al. ”Edge computing resource management and pricing for mobile blockchain.” arXiv preprint arXiv:1710.01567 (2017).
[6] Luong, Nguyen Cong, et al. ”Optimal auction for edge computing resource management in mobile blockchain networks: A deep learning approach.” arXiv preprint arXiv:1711.02844 (2017).
[7] Jiao, Yutao, et al. ”Social Welfare Maximization Auction in Edge Computing Resource Allocation for Mobile Blockchain.” arXiv preprint arXiv:1710.10595 (2017).
[8] Han, Zhu, et al. Game theory in wireless and communication networks: theory, models, and applications. Cambridge university press, 2012.
[9] Dtting, Paul, et al. ”Optimal auctions through deep learning.” arXiv preprint arXiv:1706.03459 (2017).
[10] Suankaewmanee, Kongrath, et al. ”Performance analysis and application of mobile blockchain.” arXiv preprint arXiv:1712.03659 (2017).
[11] Kraft, Daniel. ”Difficulty control for blockchain-based consensus systems.” Peer-to-Peer Networking and Applications 9.2 (2016): 397-413.
[12] Houy, Nicolas. ”The Bitcoin mining game.” (2014).
[13] Rizun, Peter R. ”A transaction fee market exists without a block size limit.” Block Size Limit Debate Working Paper (2015).
[14] Jackson, Matthew O. Social and economic networks. Princeton university press, 2010.
[15] Yokota, Takao, and Mitsuo Gen. ”Solving for nonlinear integer programming problem using genetic algorithm and its application.” Systems, Man, and Cybernetics, 1994. Humans, Information and Technology., 1994 IEEE International Conference on. Vol. 2. IEEE, 1994.
[16] Whitley, Darrell. ”A genetic algorithm tutorial.” Statistics and computing 4.2 (1994): 65-85.