A New Differential Evolution Algorithm for Joint Mining Decision and Resource Allocation in a MEC-Enabled Wireless Blockchain Network

Yong Wang[a,b], Chun-Rong Chen[a,b], Pei-Qiu Huang[a,b], Kezhi Wang[c]

[a]*School of Automation, Central South University, Changsha 410083, China*

[b]*Hunan Xiangjiang Artificial Intelligence Academy, Changsha 410083, China*

[c]*Department of Computer and Information Sciences, Northumbria University, Newcastle NE1 8ST, UK*

Abstract

This paper studies a mobile edge computing-enabled wireless blockchain network, in which a set of Internet of Things (IoT) devices can act as miners to participate in mining. In this blockchain network, we jointly optimize the mining decision and resource allocation to maximize the total profit of all miners. When using evolutionary algorithms to solve this problem, each individual usually represents the mining decisions and resource allocations of all miners, which results in the redundant search space due to the fact that not all miners participate in mining. In this paper, we propose a new differential evolution (DE) algorithm, called DEMiDRA. In DEMiDRA, each individual represents the resource allocation of a participating miner and the resource allocations of all participating miners constitute the whole population. Then, DE is adopted to optimize the resource allocation. As for the optimization of the mining decision, we need to select miners to participate in mining and update the number of participating miners. Since the population

size is equal to the number of participating miners, we transform the update of the number of participating miners into the adjustment of the population size and design an adaptive strategy. Besides, a tabu strategy is developed to prevent unpromising miners from participating in mining. The effectiveness of DEMiDRA is verified by comparing it with three other algorithms on a set of instances.
Keywords: blockchain, mobile edge computing, differential evolution, encoding, mining decision, resource allocation

1. Introduction

With the popularity of digital cryptocurrencies represented by Bitcoin (Böhme et al., 2015), blockchain networks have received widespread attention in past years. They are decentralized peer-to-peer networks that do not require the participation of third parties and can guarantee tamper-proof ledger, and transparent and secure transactions (Yuan and Wang, 2018). Due to these advantages, blockchain networks have been applied in various fields, such as Internet of Things (IoT) (Zhang et al., 2019), smart manufacturing (Leng et al., 2020), supply chain (Azzi et al., 2019), and smart grid (Wang et al., 2019c).

Blockchain networks record data as blocks and form a linked list structure. In order to add a new block to the current blockchain network, blockchain users (i.e., miners) need to address the Proof-of-Work (PoW) puzzle (also known as mining) to obtain a hash value to link previous and current blocks. After the PoW puzzle is solved, the results will be broadcast to other miners in the blockchain network for verification. If most miners reach consensus,

the new block is successfully added (Xiong et al., 2018). The mining process involves an exhaustive query of an anti-collision hash function, which is generally resource-intensive and requires high computing capabilities of miners. However, usually limited resources can be provided to miners. This gives rise to the question of how to manage resources.

Recently, several researches have devoted to resource management in blockchain networks. They mainly focus on two key issues: mining decision and resource allocation. The former is to decide whether a miner participates in mining or not and the latter is to determine how many resources are allocated to each participating miner. Houy (2016) studied the mining decision in a two-miner Bitcoin network. Kiayias et al. (2016) adopted the stochastic game for the mining decision in a multi-miner Bitcoin network. Due to the randomness of the mining process, miners are usually willing to join in mining pools to obtain stable profits. To this end, Liu et al. (2018b) considered the dynamics of mining pool selection in a blockchain network, and then modeled the mining pool selection problem as an evolutionary game and provided a theoretical analysis of evolutionary stability. Since some miners in the mining pool may exhibit malicious behaviors, resulting in wasted computing resources, Tang et al. (2019) designed a game-theoretic framework to motivate miners to participate honestly in mining. Although the success of these methods has been reported, all of them are studied on wired blockchain networks. With the rise of IoT devices (IoTDs), wireless blockchain networks running on IoTDs have attracted much attention (Ali et al., 2019). However, due to limited computing capabilities, IoTDs cannot support mining on local devices (Jiang et al., 2019).

Mobile edge computing (MEC) is a promising technology that can enhance computing capabilities of IoTDs by offloading tasks to the MEC server (Wang et al., 2019b). Several works have been conducted on MEC-enabled wireless blockchain networks. For instance, Liu et al. (2018a) designed a MEC-enabled wireless blockchain network and proposed an alternating direction method of multipliers for resource management. However, Liu et al. (2018a) ignored the profit of the MEC service provider (MSP). When its profit is too low, the MSP may no longer provide computing services for miners, and the wireless blockchain networks may not be able to work. Luong et al. (2018) adopted a deep learning approach for resource management in MEC-enabled wireless blockchain networks. Jiao et al. (2019) studied on two bidding schemes for resource management in MEC-enabled wireless blockchain networks: the constant-demand scheme and the multi-demand scheme. In the former, an auction mechanism is designed to achieve the optimal social welfare. In the latter, an approximate algorithm is proposed to simultaneously take the truthfulness, individual rationality, and computational efficiency into account. In addition, Xiong et al. (2019) modeled the resource management problem as a two-stage Stackelberg game and then obtained the Stackelberg equilibrium under two different price schemes. However, these papers (Luong et al., 2018; Jiao et al., 2019; Xiong et al., 2019) do not consider the transmission delay between the miners and the MEC server. If too many miners offload tasks to the MEC server at the same time, they may face severe interference, resulting in a high transmission delay (Huang et al., 2019).

Different from existing works, in this paper, we consider both the profit of

the MSP and the transmission delay in a MEC-enabled wireless blockchain network. In order to maximize the total profit of all miners, we jointly optimize the mining decision and resource allocation. When using evolutionary algorithms (EAs) to solve this problem, each individual usually represents the mining decisions and resource allocations of all miners. Since not all miners participate in mining, encoding resource allocations of all miners into individuals will generate the redundant search space, thus causing poor performance. To avoid this issue, a new differential evolution (DE) algorithm is proposed, called DEMiDRA. The main contributions of this paper are summarized as follows:

- We devise a new encoding scheme, in which the resource allocation of each participating miner is encoded as an individual and the whole population represents the resource allocations of all participating miners. As a result, the population size is equal to the number of participating miners. In this way, only the resource allocations of participating miners are considered, thereby eliminating the redundant search space.

- Because the population size is equal to the number of participating miners, the update of the number of participating miners is essentially equivalent to the adjustment of the population size. To this end, we design an adaptive strategy to adjust the population size. Besides, a tabu strategy is proposed to prevent unpromising miners from participating in mining.

- Extensive experiments are carried out on a set of instances to investigate the performance of DEMiDRA. Its effectiveness is verified by

comparing it with the other three algorithms.

The rest of this paper is organized as follows. Section 2 presents the system model and problem formulation. Section 3 describes the details of DEMiDRA. The experimental studies are shown in Section 4. Finally, Section 5 concludes this paper.

2. System Model and Problem Formulation

Nomenclature

$\mathbf{f}$	Computing resources allocated to all participating miners
$\mathbf{m}$	Mining decisions of all miners
$\mathbf{p}$	Transmission power allocated to all participating miners
$\mathcal{N}$	Set of miners
$\mathcal{N}'$	Set of participating miners
D_i	Block size of the task of the ith miner
E_i^m	Mining energy consumption of the task of the ith participating miner
E_i^t	Transmission energy consumption of the task of the ith participating miner
F^{miner}	Profit of all miners
F^{MSP}	Profit of the MSP
F_i^{miner}	Profit of the ith participating miner

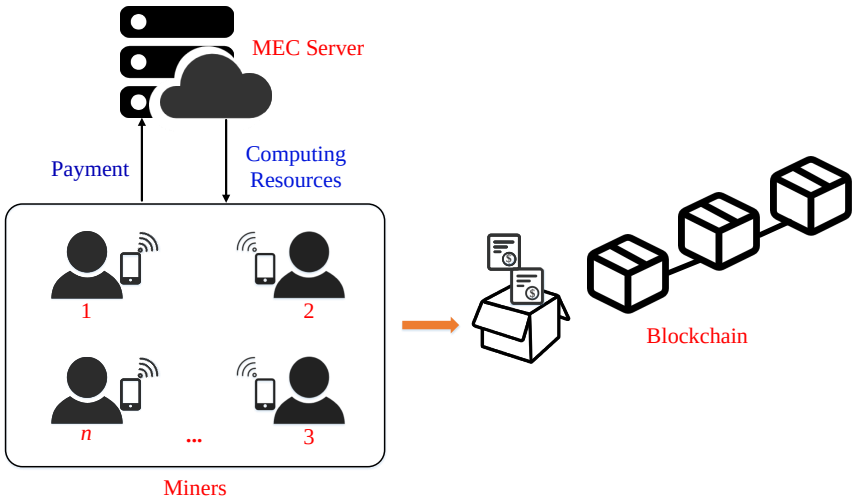

Figure 1: A MEC-enabled wireless blockchain network.

n	Number of miners
n'	Number of participating miners
P_i^m	Probability that the ith participating miner obtains the reward
P_i^o	Orphaning probability of the ith participating miner
R_i	Transmission rate of the ith participating miner
T_i^m	Mining time of the task of the ith participating miner
T_i^t	Transmission time of the task of the ith participating miner
X_i	Computing intensity (in CPU Cycles/bit) of the task of the ith miner

As shown in Fig. 1, a MEC-enabled wireless blockchain network is considered, in which a set of n IoTDs, denoted as $\mathcal{N} = \{1, 2, \ldots, n\}$, can act as miners to participate in mining and each of them has a mining task to

execute (Liu et al., 2018a). For the sake of simplicity, we define the mining task of the ith miner as a 2-tuple: $\{D_i, X_i\}$, where D_i and X_i represent the block size of the task of the ith miner and the computing intensity (in CPU Cycles/bit) of the task of the ith miner, respectively. Due to the limited computing capabilities of IOTDs, if a miner decides to participate in mining, it needs to purchase computing resources from the MSP and then offload its mining task to the MEC server. In the studied blockchain network, the mining task is considered to be completed only after three steps (i.e., offloading, mining, and propagation steps) are successfully completed, and then the miner can obtain a reward. If a miner decides not to participate in mining or fails to complete its mining task, it cannot obtain any reward (Liu et al., 2018a).

In this paper, $\mathbf{m} = \{m_1, \ldots, m_n\}$ is defined as the mining decisions of all miners, where $m_i = 1$ or $m_i = 0$ ($i \in \mathcal{N}$) indicates that the ith miner decides to or decides not to participate in mining, respectively. Herein, $\mathcal{N}'$ is used to represent the set of participating miners. Thus, the number of participating miners is equal to $n' = \sum_{i \in \mathcal{N}} m_i$.

In addition, we need to allocate resources (including the transmission power of IoTDs and the computing resources of the MEC server) to the participating miners. In this paper, $\mathbf{p} = \{p_1, \ldots, p_{n'}\}$ and $\mathbf{f} = \{f_1, \ldots, f_{n'}\}$ are defined as the transmission power and computing resources allocated to all participating miners, respectively, where p_i and f_i ($i \in \mathcal{N}'$) represent the transmission power and computing resources (CPU Cycles/s) allocated to the ith participating miner, respectively.

2.1. Offloading Step

In this step, participating miners simultaneously transmit tasks to the MEC server. The transmission rate of the ith participating miner is expressed as (Chen, 2015; Huang et al., 2019)

$$R_i = B\log_2\left(1 + \frac{p_i H_i}{\sigma^2 + \sum_{j\in\mathcal{N}'\setminus i} m_j p_j H_j}\right),\ \forall i \in \mathcal{N}' \tag{1}$$

where B denotes the bandwidth, H_i denotes the channel state information of the ith participating miner, and $\sum_{j\in\mathcal{N}'\setminus i} m_j p_j H_j$ represents the interference received by the ith participating miner from other participating miners, and σ^2 represents the background noise power.

Then, the transmission time and energy consumption of the task of the ith participating miner can be respectively given as

$$T_i^t = \frac{D_i}{R_i},\ \forall i \in \mathcal{N}' \tag{2}$$

and

$$E_i^t = p_i T_i^t,\ \forall i \in \mathcal{N}'. \tag{3}$$

2.2. Mining Step

In this step, the MEC server executes the mining tasks transmitted by participating miners. The mining time and energy consumption of the task of the ith participating miner are respectively expressed as

$$T_i^m = \frac{D_i X_i}{f_i},\ \forall i \in \mathcal{N}' \tag{4}$$

and

$$E_i^m = k_1 f_i^3 T_i^m,\ \forall i \in \mathcal{N}' \tag{5}$$

where k_1 is the effective capacitance coefficient.

In blockchain networks, a miner who executes a mining task faster has a greater probability of obtaining a reward (Xiong et al., 2019). Therefore, it is assumed that the probability that a miner obtains the reward is inversely proportional to its mining time, which is expressed as

$$P_i^m = \frac{k_2}{T_i^m}, \ \forall i \in \mathcal{N}' \tag{6}$$

where k_2 is the scale factor.

2.3. Propagation Step

After completing the mining step, if the result propagates slowly, the miner still cannot obtain a reward. The reason is that under this condition, the consensus may not be reached and the block may be discarded, which is called orphaning (Liu et al., 2018a). In blockchain networks, the block is generated following a Poisson process with a constant mean rate of λ, and its propagation time T_i^o is linearly related to the block size D_i(Xiong et al., 2019). The orphaning probability of the ith participating miner is given as

$$\begin{aligned} P_i^o &= 1 - e^{-\lambda(T_i^o + T_i^s)} \\ &= 1 - e^{-\lambda(zD_i + T_i^t)}, \ \forall i \in \mathcal{N}' \end{aligned} \tag{7}$$

where z denotes a given delay factor and T_i^s is the time when the ith participating miner starts mining its block. In this paper, the mining task of the ith participating miner will be executed once it is received by the MEC server; thus, $T_i^s = T_i^t$.

2.4. Profit Model

The reward obtained by a miner consists of two parts: a fixed reward and a variable reward. In addition, a miner needs to consume certain communication and computing costs. Thus, the profit of the ith participating miner is specified by

$$F_i^{miner} = (w + \alpha D_i)P_i^m(1 - P_i^o) - c_1 E_i^t - c_2 f_i, \ \forall i \in \mathcal{N}' \tag{8}$$

where w represents the fixed reward, αD_i represents the variable reward, α denotes the variable reward factor, and c_1 and c_2 denote the prices of the transmission energy and computing resources, respectively. Thus, the profit of all miners is

$$F^{miner} = \sum_{i \in \mathcal{N}'} F_i^{miner}. \tag{9}$$

In addition, the MSP obtains the revenue by selling computing resources to the miners, but it needs to pay for its cost of energy consumption, which includes the mining energy consumption and the no-load energy consumption. Therefore, the profit of the MSP can be expressed as

$$F^{MSP} = \sum_{i \in \mathcal{N}'} (c_2 f_i - c_3 E_i^m) - c_3 E_0 \tag{10}$$

where c_3 denotes the price of energy consumed by the MSP and E_0 represents the no-load energy consumption of the MSP.

2.5. Problem Formulation

In the studied blockchain network, we jointly optimize the mining decision (i.e., $\mathbf{m}$) and resource allocation (i.e., $\mathbf{p}$ and $\mathbf{f}$), with the aim of maximizing

the total profit of all miners. The problem is formulated as:

$$\begin{aligned}
\max_{\mathbf{m},\mathbf{p},\mathbf{f}} \quad & F^{miner} = \sum_{i \in \mathcal{N}'} F_i^{miner} \\
\text{s.t.} \quad & C1 : m_i \in \{0,1\},\ \forall i \in \mathcal{N} \\
& C2 : p^{min} \leq p_i \leq p^{max},\ \forall i \in \mathcal{N}' \\
& C3 : f^{min} \leq f_i \leq f^{max},\ \forall i \in \mathcal{N}' \\
& C4 : \sum_{i \in \mathcal{N}'} f_i \leq f^{total} \\
& C5 : F^{MSP} \geq 0 \\
& C6 : T_i^t + T_i^m + T_i^o \leq T_i^{max},\ \forall i \in \mathcal{N}'
\end{aligned} \tag{11}$$

where $\mathcal{C}1$ indicates that each miner can decide whether to participate in mining or not; $\mathcal{C}2$ specifies the minimum and maximum transmission power allocated to each participating miner; $\mathcal{C}3$ states the minimum and maximum computing resources allocated to each participating miner; $\mathcal{C}4$ indicates that the total computing resources allocated to participating miners cannot exceed the total capacity of the MEC server; $\mathcal{C}5$ ensures that the profit of the MSP should not be less than 0; and $\mathcal{C}6$ states that the total time of offloading, mining, and propagation steps cannot exceed the maximum time constraint. Note that, in the studied blockchain network, we assume that IoTDs are homogeneous and each of them has the same range of the transmit power (i.e., $[p^{min}, p^{max}]$) and the same range of the computing resources (i.e., $[f^{min}, f^{max}]$).

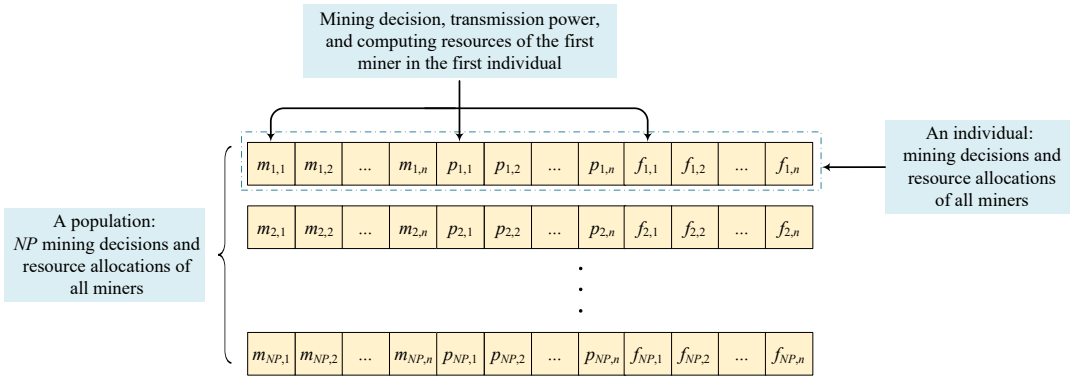

Figure 2: The commonly used encoding scheme, where NP denotes the population size of EAs.

3. Proposed Approach

3.1. Motivation

It can be observed that (11) is a mixed-variable nonlinear optimization problem since $\mathbf{m}$ is a binary vector, and $\mathbf{p}$ and $\mathbf{f}$ are two continuous vectors. As a result, (11) is difficult to solve by traditional optimization methods (Liu et al., 2020). In this paper, we use EAs to solve (11) as they have great potential in dealing with complex optimization problems (Yi et al., 2020; Luo et al., 2020; Xiang et al., 2019). The encoding scheme in Fig. 2 is commonly used in EAs (Jiang et al., 2019; Guo et al., 2018), where an individual represents the mining decisions and resource allocations of all miners. It is worth noting that if a miner decides not to participate in mining (i.e., $m_i = 0$), this miner does not need to transmit its mining task to the MEC server or purchase computing resources to execute its mining task. In this case, the resource allocation for this non-participating miner is not required. Therefore, the commonly used encoding scheme generates the

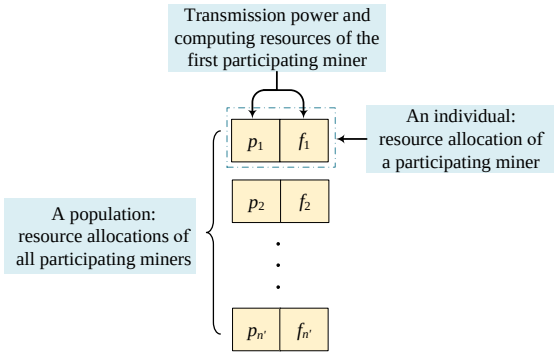

Figure 3: The proposed encoding scheme.

redundant search space, which may cause performance degradation.

In this paper, eliminating the redundant search space is essential to improve the performance of the algorithm. In addition, we can observe that the search region of the resource allocation of each participating miner is the same (i.e., $[p_{min}, p_{max}] \times [f_{min}, f_{max}]$). Motivated by (Wang et al., 2018b; Huang et al., 2020), a novel encoding scheme is proposed as shown in Fig. 3. To be specific, each individual represents the resource allocation of a participating miner and the resource allocations of all participating miners constitute the whole population. As a result, the population size is equal to the number of participating miners. It is noteworthy that although the mining decisions are not encoded into individuals directly, they are implicitly considered in the proposed encoding scheme since we focus on the resource allocations of participating miners.

Compared with the commonly used encoding scheme, the proposed encoding scheme has the following advantages:

- With respect to the commonly used encoding scheme, the resource

allocations of all miners are considered even if some miners do not participate in mining. However, in the proposed encoding scheme, only the resource allocations of participating miners are considered; thus, the redundant search space can be eliminated.

- Regarding the commonly used encoding scheme, each individual contains both binary and continuous variables. In contrast, in the proposed encoding scheme, each individual contains only continuous variables and no binary variables. Therefore, we only need to adopt the evolutionary operators designed for continuous variables.

- For the commonly used encoding scheme, the length of each individual in the population is $3n$, which will give rise to a huge search space when the value of n is big (e.g., $n > 100$). But for the proposed encoding scheme, the length of each individual is very short (i.e., two), regardless of the value of n. Thus, the optimal solution is searched in a two-dimensional search space.

- In the commonly used encoding scheme, an important parameter (i.e., the population size) needs to be set in advance. However, the population size is equal to the number of participating miners in the proposed encoding scheme. Therefore, this parameter does not need to be preset.

Based on the proposed encoding scheme, a new DE algorithm is devised for jointly optimizing the mining decision and resource allocation, called DEMiDRA.

Algorithm 1 General Framework of DEMiDRA

Input: The number of participating miners: n_{min}, the maximum number of fitness evaluations (FEs): $MaxFEs$

Randomly choose n_{min} miners as the participating miners and initialize the mining decisions: $\mathbf{m}$;

Initialize the resource allocations of all participating miners: $\mathbf{p}$ and $\mathbf{f}$, which form an initial population $\mathcal{P}$;

Evaluate the objective function value and degree of constraint violation of $\mathcal{P}$ based on (11);

$FEs = 1$; // FEs denotes the number of FEs

Initialize the selection probability vector $\mathbf{r} = [r_1, r_2, r_3]$ to represent the selection probabilities of the insertion, deletion, and replacement operators;

Initialize $Tlist = \emptyset$ to record unpromising miners;

while $FEs < MaxFEs$ **do**

  Implement the mutation and crossover operators to generate an offspring population $\mathcal{Q}$;

  for i=1:$|\mathcal{Q}|$ **do**

    Update $\mathbf{m}$ and $\mathcal{P}$ according to **Algorithm 2**;

    Update $\mathbf{r}$ and $Tlist$ according to **Algorithm 3**;

  end for

end while

Output: $\mathbf{m}$ and $\mathcal{P}$

3.2. General Framework

The general framework of DEMiDRA is presented in **Algorithm 1**. In the initialization, we first randomly choose n_{min} miners[1] as the participating miners. Besides, the resource allocations of all participating miners are randomly generated in the search space, which form an initial population $\mathcal{P}$. Then, the objective function value and degree of constraint violation of $\mathcal{P}$ are evaluated based on (11). Subsequently, we initialize a probability vector $\mathbf{r} = [r_1, r_2, r_3]$, where r_1, r_2, and r_3 represent the selection probabilities of the insertion, deletion, and replacement operators, respectively. Moreover, a tabu list $Tlist$ is initialized as an empty set, with the aim of recording unpromising miners. In the evolution, an offspring population $\mathcal{Q}$ with the same size as $\mathcal{P}$ (i.e., $|\mathcal{P}| = |\mathcal{Q}|$) is generated via the "DE/rand/1" mutation operator and the binomial crossover operator (Wang et al., 2019a, 2018a). [2] Then, by making use of each individual in $\mathcal{Q}$, $\mathbf{m}$ and $\mathcal{P}$ are updated via **Algorithm 2**. In addition, $\mathbf{r}$ and $Tlist$ are updated in **Algorithm 3**. The evolution is repeated until the stopping criterion is met (i.e., the maximum number of fitness evaluations (FEs) $MaxFEs$ is reached). The updates of $\mathbf{m}$, $\mathcal{P}$, $\mathbf{r}$, and $Tlist$ are introduced in the following.

[1]The reason why n_{min} miners are chosen to participate in mining in the initialization is that $\mathcal{C}4$ cannot be satisfied when the number of participating miners is less than n_{min}, where $n_{min} = \lfloor \frac{4c_3^2 E_0 k_1 \min_{i \in \mathcal{N}}(D_i X_i)}{c_2^2} \rfloor$. A detailed proof is given in the Appendix.

[2]Note that, there are various mutation operators of DE. The reason why the "DE/rand/1" mutation operator is selected is that it is the most commonly used one (Wang et al., 2018b; Huang et al., 2020).

Algorithm 2 Updates of $\mathbf{m}$ and $\mathcal{P}$

Input: Parent population: $\mathcal{P}$, offspring population: $\mathcal{Q}$, probability vector: $\mathbf{r}$, and tabu list: $Tlist$

Generate a random number $rand$ between [0,1];

if $0 \leq rand \leq r_1$ and $|\mathcal{P}| < n$ **then**

    Implement the insertion operator based on $\mathcal{P}$ and the ith individual in $\mathcal{Q}$ to obtain $\mathbf{m}'$ and $\mathcal{P}'$;

else if $r_1 < rand \leq (r_1 + r_2)$ and $|\mathcal{P}| > n_{min}$ **then**

    Implement the deletion operator based on $\mathcal{P}$ to obtain $\mathbf{m}'$ and $\mathcal{P}'$;

else

    Implement the replacement operator based on $\mathcal{P}$ and the ith individual in $\mathcal{Q}$ to obtain $\mathbf{m}'$ and $\mathcal{P}'$;

end if

Evaluate the objective function value and degree of constraint violation of $\mathcal{P}'$ based on (11);

$FEs = FEs + 1$;

if $\mathcal{P}'$ is better than $\mathcal{P}$ based on the feasibility rule (Deb, 2000) **then**

    $\mathbf{m} \leftarrow \mathbf{m}'$ and $\mathcal{P} \leftarrow \mathcal{P}'$;

end if

Output:

Output: $\mathbf{m}$, $\mathcal{P}$, and $\mathcal{P}'$

3.3. Updates of $\mathbf{m}$ *and* $\mathcal{P}$

In order to optimize the mining decision, we need to select miners to participate in mining and update the number of participating miners. To this end, the following two aspects are crucial toward the optimization of the mining decision:

- As stated in Section 3.1, each individual in $\mathcal{P}$ represents the resource allocation of a participating miner; therefore, $|\mathcal{P}|$ is equal to the number of participating miners. Since the optimal number of participating miners is unknown, $|\mathcal{P}|$ should be able to increase, decrease, or keep unchanged for updating the number of participating miners. Moreover, $|\mathcal{P}|$ should not change dramatically; otherwise, the search may be unstable.

- It is necessary to prevent unpromising miners from participating in mining to enhance the efficiency.

Considering the first aspect, we design three operators, namely, insertion, deletion, and replacement operators, to generate three new populations. Their population sizes are $|\mathcal{P}+1|$, $|\mathcal{P}-1|$, and $|\mathcal{P}|$, respectively. By using one of them to update $\mathcal{P}$, $|\mathcal{P}|$ can steadily increase, decrease, or keep unchanged. In addition, only one individual in $\mathcal{P}$ is different from each new population, which suggests that at most one individual in $\mathcal{P}$ is updated in each generation. Therefore, $\mathcal{P}$ can be updated in a stable way. Note, however, that if all the three operators are implemented in each update, three FEs are required for evaluating the three new populations. As a result, the overall evolutionary process requires a large number of FEs. To this end,

an operator is adaptively selected in each update based on $\mathbf{r}$. Regarding the second aspect, a tabu strategy is designed, in which unpromising miners are added to $Tlist$.

The update process of $\mathbf{m}$ and $\mathcal{P}$ is presented in **Algorithm 2**. First, the roulette wheel selection is adopted to select an operator from the insertion, deletion, and replacement operators based on $\mathbf{r}$ to update $\mathbf{m}$ and $\mathcal{P}$. The updated $\mathbf{m}$ and $\mathcal{P}$ are denoted as $\mathbf{m}'$ and $\mathcal{P}'$, respectively. Note that if $|\mathcal{P}|$ is equal to n or n_{min}, the insertion or deletion operator cannot be implemented, respectively. Based on $\mathcal{P}$ and the ith ($i = 1, \ldots, |\mathcal{Q}|$) individual in $\mathcal{Q}$, the insertion, deletion, and replacement operators are described as follows:

- Insertion operator: A miner, who does not participate in mining and is not in $Tlist$, is randomly selected to participate in mining. Then, the ith individual in $\mathcal{Q}$ is used to represent the resource allocation of this miner and added into $\mathcal{P}$.

- Deletion operator: A participating miner is randomly selected to give up mining and its resource allocation is removed from $\mathcal{P}$.

- Replacement operator: The ith individual in $\mathcal{Q}$ is used to randomly replace the resource allocation of a participating miner (i.e., a random individual in $\mathcal{P}$). Since $\mathbf{m}$ is not changed, $\mathbf{m}'$ is the same with $\mathbf{m}$.

Then, the objective function value and degree of constraint violation of $\mathcal{P}'$ are evaluated based on (11). If $\mathcal{P}'$ is better than $\mathcal{P}$ based on the commonly used feasibility rule (Deb, 2000), $\mathbf{m}$ and $\mathcal{P}$ are replaced with $\mathbf{m}'$ and $\mathcal{P}'$, respectively. Note that, in the updating of $\mathcal{P}$, the used selection operator is different from the conventional selection operator of DE (Zhou et al., 2019;

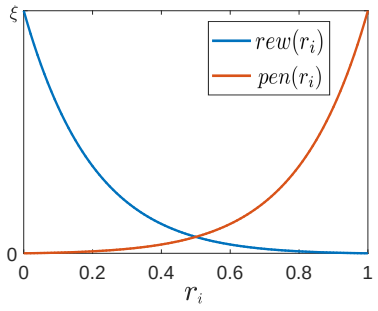

Figure 4: Trends of $rew(r_i)$ and $pen(r_i)$.

Chu et al., 2019) since a better population is selected rather than a better individual.

3.4. Updates of $\mathbf{r}$ *and Tlist*

3.4.1. Update of $\mathbf{r}$

In this paper, $\mathbf{r}$ is updated by collecting information on which operator is used and the performance of $\mathcal{P}'$. To be specific, if an operator is implemented and $\mathcal{P}'$ is better than $\mathcal{P}$, the selection probability of this operator will be increased by $rew(r_i)$, where $rew(r_i)$ is a reward function. In contrast, if an operator is implemented and $\mathcal{P}'$ is worse than $\mathcal{P}$, the selection probability of this operator will be decreased by $pen(r_i)$, where $pen(r_i)$ is a penalty function. $rew(r_i)$ and $pen(r_i)$ are designed as

$$rew(r_i) = \xi(1 - r_i)e^{-4r_i},\ i = 1, 2, 3 \tag{12}$$

and

$$pen(r_i) = \xi r_i e^{-4(1-r_i)},\ i = 1, 2, 3. \tag{13}$$

The reasons for designing the reward and penalty functions in (12) and (13) are as follows. As shown in Fig. 4, the value of $rew(r_i)$ changes from ξ

to 0 when the value of r_i changes from 0 to 1. In this way, a large reward is given when the value of r_i is small and a small reward is given when the value of r_i is big. It is reasonable since when the value of r_i is big, it means that the corresponding operator already has a high probability to be selected. In this case, a small reward can avoid a dramatic increase in the value of $rew(r_i)$. In contrast, a large penalty is given when the value of r_i is big, and a small penalty is given when the value of r_i is small. As a result, as r_i increases, the values of $rew(r_i)$ and $pen(r_i)$ have the opposite trends.

It is worth noting that in the initialization, the number of participating miners is small. Therefore, big selection probabilities are set for the insertion and replacement operators while a smaller selection probability for the deletion operator. Specifically, $\mathbf{r}$ is initialized as $\mathbf{r} = [0.45, 0.10, 0.45]$. Moreover, after each update of $\mathbf{r}$, the normalization is performed to ensure that $r_1 + r_2 + r_3 = 1$. The normalization of $\mathbf{r}$ is: $r_i = r_i/(r_1 + r_2 + r_3),\ i = 1, 2, 3$.

3.4.2. Update of Tlist

If the performance of the population cannot be improved after a miner participates in mining or if the performance of the population can be improved after a miner gives up participating in mining, both miners are considered to be unpromising miners. The reason is that if these miners are immediately selected to participate in mining again, the performance of the population is unlikely to be improved. As a result, we record these miners in *Tlist* and prevent them from participating in mining.

Specifically, *Tlist* is updated in the following two cases:

- If the insertion operator is implemented and $\mathcal{P}'$ is not better than $\mathcal{P}$, the miner who intends to participate in mining is added into *Tlist*.

Algorithm 3 Updates of **r** and $Tlist$

Input: Parent population: $\mathcal{P}$, new population: $\mathcal{P}'$, probability vector: **r**, and tabu list: $Tlist$
if the insertion operator is implemented **then**
  if $\mathcal{P}'$ is better than $\mathcal{P}$ **then**
    $r_1 = r_1 + rew(r_1)$;
  else
    $r_1 = r_1 - pen(r_1)$;
    Add the miner who intends to participate in mining into $Tlist$;
  end if
else if the deletion operator is implemented **then**
  if $\mathcal{P}'$ is better than $\mathcal{P}$ **then**
    $r_2 = r_2 + rew(r_2)$;
    Add the miner who gives up participating in mining into $Tlist$;
  else
    $r_2 = r_2 - pen(r_2)$;
  end if
else if the replacement operator is implemented **then**
  if $\mathcal{P}'$ is better than $\mathcal{P}$ **then**
    $r_3 = r_3 + rew(r_3)$;
  else
    $r_3 = r_3 - pen(r_3)$;
  end if
end if
Normalize **r**;
Output: **r** and $Tlist$

- If the deletion operator is implemented and $\mathcal{P}'$ is better than $\mathcal{P}$, the miner who gives up participating in mining is added into $Tlist$.

In this paper, the length of $Tlist$ is limited (e.g., $0.05n$). When $Tlist$ is full, in order to add a new unpromising miner, the earliest added miner will be removed. After removed from $Tlist$, this miner can be selected to participate in mining again. The updates of $\mathbf{r}$ and $Tlist$ are given in **Algorithm 3**.

3.5. Discussion

The major characteristics of DEMiDRA are discussed below.

- Although DEMiDRA shares some similarity to the methods in (Wang et al., 2018b; Huang et al., 2020) in terms of the encoding scheme, there are still several differences. To be specific, DEMiDRA uses a variable-size population, while the method in (Wang et al., 2018b) uses a fixed-size population. In addition, DEMiDRA is used to solve the mixed-variable optimization problem while the method in (Huang et al., 2020) is proposed for the variable-length optimization problem. Further, DEMiDRA is different from the method in (Huang et al., 2020) by designing an adaptive strategy to adjust the population size.

- DEMiDRA treats the whole population as a solution and changes the miner decision and resource allocation of one miner at most in each update. Therefore, DEMiDRA is similar to local search methods. Note, however, that local search methods typically randomly changes the miner decision and resource allocation of one miner. However, DEMiDRA adopts the crossover and mutation operators of DE

to changes the miner decision and resource allocation. In this case, DEMiDRA can use the information of other miners.

4. Experimental Studies

4.1. Algorithms for Comparison

The following algorithms were under our consideration for performance comparison.

- DE (Das and Suganthan, 2011): DE is originally designed for continuous-variable optimization problems and needs to be modified when dealing with mixed-variable optimization problems. To satisfy integer restrictions in the mining decision, the rounding operator is adopted after the crossover operator, which can transform a continuous value to its closest integer.

- ACO_{MV} (Liao et al., 2014): ACO_{MV} already has categorical and continuous-variable optimization mechanisms, which can deal with the mining decision and resource allocation, respectively.

- BOToP (Liu et al., 2020): BOToP first solves a transformed constrained biobjective optimization problem, the purpose of which is to approach the optimal solution of the mixed-variable optimization problem, and then DE is used to solve the original mixed-variable optimization problem to obtain the final optimal solution.

Note that these three competitors adopt the commonly used encoding scheme.

Table 1: Parameter Settings of the Studied Blockchain Network

Parameter	Value	Parameter	Value	Parameter	Value
X	$1.8e+5$ Cycles/bit	z	$1.00e-4$	c_3	3 Token/J
σ^2	-174dBm/Hz	B	10 MHz	D_i	[1, 2] Kbit
c_2	10 Token/GCycles	$T_{i,max}$	4s	f^{total}	800 GCycles/s
α	0.005 Token/bit	c_1	20 Token/J	w	2
f_i	[0.1, 2] GCycles/s	E_0	70 J	p_i	[0.01, 1] W
λ	1/600	k_1	1e-27	k_2	1

4.2. Parameter Settings

The parameter settings of the three competitors and DEMiDRA are presented as follows:

- Population size: The population sizes of DE, ACO_{MV}, and BOToP were set to 100, 60, and 30, respectively, which are consistent with the original studies (Das and Suganthan, 2011; Liao et al., 2014; Liu et al., 2020). In addition, there is no need to set the population size for DEMiDRA since its population size is equal to the number of participating miners.

- Specific parameters: For DE, the scaling factor F and the crossover control parameter CR were set to 0.9 and 0.5, respectively. For ACO_{MV}, the influence of the best-quality solution was set to 0.05099 and the search width was set to 0.6795. For BOToP, F and CR were selected from two parameter pools: $\{0.6, 0.8, 1.0\}$ and $\{0.1, 0.2, 1.0\}$, respectively. For DEMiDRA, F and CR were set to 0.9 and 0.5, respectively. Besides, ξ in (12) and (13) was set to 0.005.

- Stopping criterion: Each algorithm terminated when 10,000 FEs was

Table 2: Results (in Token) of DEMiDRA and the Three Competitors.

n	DE Mean±Std Dev	ACO_{MV} Mean±Std Dev	BOToP Mean±Std Dev	DEMiDRA Mean±Std Dev
50	1.42e+3±3.41e+1 ↑ [59.86%]	1.87e+3±1.33e+1 ↑ [21.39%]	1.61e+3±7.18e+1 ↑ [40.99%]	2.27e+3±2.27e+2
100	2.27e+3±4.78e+1 ↑ [108.81%]	2.86e+3±1.97e+1 ↑ [65.73%]	2.50e+3±8.14e+1 ↑ [89.60%]	4.74e+3±3.60e+2
150	2.91e+3±5.69e+1 ↑ [157.73%]	3.81e+3±1.27e+2 ↑ [96.85%]	3.12e+3±1.14e+2 ↑ [140.38%]	7.50e+3±1.25e+1
200	3.59e+3±8.47e+1 ↑ [178.55%]	5.11e+3±2.57e+2 ↑ [95.88%]	3.78e+3±1.70e+2 ↑ [164.55%]	1.00e+4±1.86e+1
250	4.24e+3±8.73e+1 ↑ [194.81%]	6.23e+3±2.67e+2 ↑ [100.64%]	4.35e+3±1.27e+2 ↑ [187.36%]	1.25e+4±2.56e+2
300	4.54e+3±7.93e+2 ↑ [225.99%]	6.81e+3±2.55e+2 ↑ [117.33%]	4.60e+3±1.57e+2 ↑ [221.74%]	1.48e+4±4.18e+1
350	5.04e+3±1.28e+2 ↑ [241.27%]	7.68e+3±3.38e+2 ↑ [123.96%]	5.23e+3±1.44e+2 ↑ [228.87%]	1.72e+4±5.82e+1
400	5.30e+3±1.19e+2 ↑ [267.92%]	8.12e+3±3.09e+2 ↑ [140.15%]	5.67e+3±2.50e+2 ↑ [243.92%]	1.95e+4±1.03e+2
450	5.82e+3±1.35e+2 ↑ [241.92%]	9.08e+3±4.64e+2 ↑ [119.16%]	6.09e+3±2.37e+2 ↑ [226.77%]	1.99e+4±2.02e+2
500	5.92e+3±1.98e+2 ↑ [236.15%]	9.38e+3±4.80e+2 ↑ [112.15%]	6.60e+3±2.96e+2 ↑ [201.52%]	1.99e+4±1.69e+2
↑ / ↓ / ≈	10/0/0	10/0/0	10/0/0	

reached (i.e., $MaxFE = 10,000$). Note that, in BOToP, the first half of $MaxFEs$ was allocated to the first phase and the rest was allocated to the second phase.

The parameter settings of the studied blockchain network are given as follows. All miners were randomly distributed in a 1,000m $\times$1,000m square area and the MEC server was located in the center of this area. Ten instances with different numbers of miners (i.e., $n = 50, 100, \ldots, 500$) were considered to test the performance of DEMiDRA. The other parameter settings are summarized in Table 1.

All the experiments are implemented in MATLAB and are tested on a personal computer running with an Intel Core i5-7500 CPU @3.40 GHz and 8 GB of RAM.

4.3. Comparison With Three Competitors

The results of DEMiDRA and the three competitors are given in Table 2, where "Mean" and "Std Dev" indicate the average and standard deviation of the total profits of all miners over 30 independent runs, respectively, and percentages in the square brackets indicate the performance improvement of DEMiDRA against the three competitors. In addition, the Wilcoxon's rank-sum test at a 0.05 significance level was executed to test the statistical significance between DEMiDRA and each competitor. At the bottom of Table 2, "$\uparrow$", "$\downarrow$", and "$\approx$" indicate that DEMiDRA performs better than, worse than, and similar to each competitor, respectively.

From Table 2, we can observe that DEMiDRA provides a better average total profit than the three competitors on each instance. It is worth noting

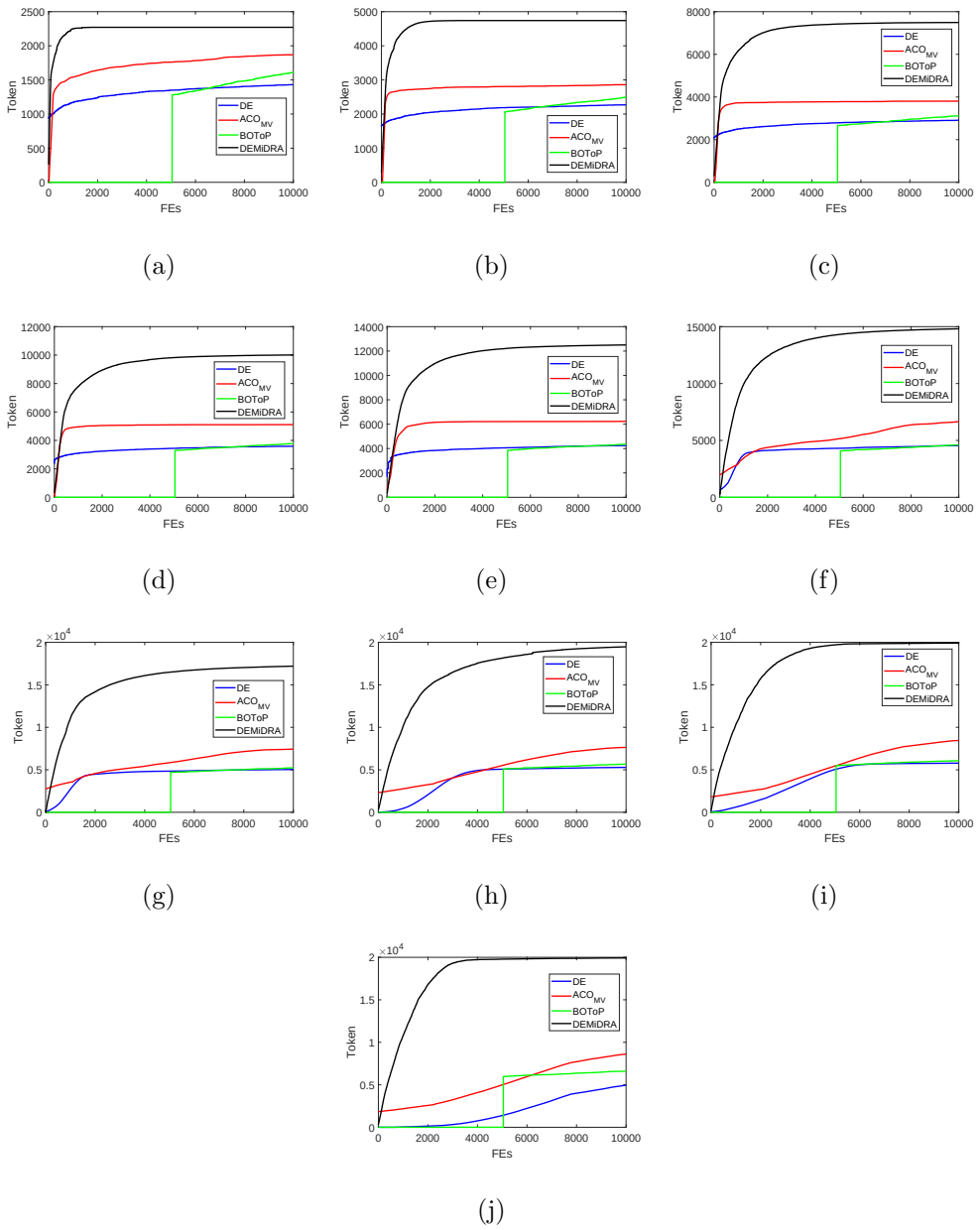

Figure 5: Evolution of the average total profits of all miners provided by DEMiDRA and the three competitors.

that, when $n \geq 200$, the average total profits obtained by DEMiDRA are an order of magnitude higher than those derived from the three competitors. In terms of the performance improvement, DEMiDRA has an advantage over the three competitors on each instance. Specifically, compared with DE, DEMiDRA provides more than 100% performance improvement on all instances except $n = 50$. When $n \geq 300$, the performance improvement of DEMiDRA is over 200%. Against ACO_{MV}, when $n \geq 250$, DEMiDRA achieves more than 100% performance improvement. In addition, when $150 \leq n \leq 250$ and $n \geq 300$, the performance improvement of DEMiDRA is more than 100% and 200% against BOToP, respectively. Besides, DEMiDRA is statistically better than the three competitors on each instance according to the Wilcoxon's rank-sum test at a 0.05 significance level. Fig. 5 depicts the evolution of the average total profits of all miners provided by DE, ACO_{MV}, BOToP, and DEMiDRA[3]. As shown in Fig. 5, DEMiDRA achieves the best performance among all algorithms.

For the above observations, we would like to give the following comments. DE, ACO_{MV}, and BOToP face the challenges caused by both mixed variables and the large search space. However, since the individuals in DEMiDRA contain only continuous variables and the length of each individual is two, DEMiDRA does not suffer from the above-mentioned challenges. More importantly, although DE, ACO_{MV}, and BOToP can generate different individuals, some of them may have the same performance since there may exist a lot of redundant variables. However, this phenomenon does not occur in

[3]Note that, when an algorithm cannot produce any feasible solution, the average total profit of all miners was set to 0.

DEMiDRA due to the fact that DEMiDRA does not contain any redundant variable.

4.4. Effect of the Adaptive Population Update Strategy

To study the effect of the adaptive population update strategy, a variant of DEMiDRA was designed, called DEMiDRA-WoA, which did not employ the adaptive population update strategy. In DEMiDRA-WoA, the selection probability of each of the insertion, deletion, and replacement operators was 1/3 and kept unchanged in the evolution. The results of DEMiDRA and DEMiDRA-WoA are presented in Table 3. Clearly, in terms of the average total profit of all miners, DEMiDRA is better than DEMiDRA-WoA on all instances. Moreover, DEMiDRA performs significantly better than DEMiDRA-WoA on nine out of ten instances according to the Wilcoxon's rank-sum test at a 0.05 significance level. The comparison reveals that the adaptive population update strategy is able to improve the performance of DEMiDRA.

4.5. Effect of the Tabu Strategy

To study the effect of the tabu strategy, another variant of DEMiDRA was designed, called DEMiDRA-WoT, in which the tabu strategy was removed. The results of DEMiDRA-WoT and DEMiDRA are presented in Table 3. As shown in Table 3, on each instance, DEMiDRA provides a better average total profit of all miners and is statistically better than DEMiDRA-WoT according to the Wilcoxon's rank-sum test at a 0.05 significance level, which can verify the effectiveness of the tabu strategy.

Table 3: Results (in Token) of DEMiDRA and Its Two Variants.

n	DEMiDRA-WoA Mean±Std Dev	DEMiDRA-WoT Mean±Std Dev	DEMiDRA Mean±Std Dev
50	2.18e+3±2.49e+2 ≈ [4.13%]	2.14e+3±1.71e+2 ↑ [6.07%]	2.27e+3±2.27e+2
100	4.52e+3±2.08e+2 ↑ [4.87%]	4.50e+3±2.16e+1 ↑ [5.33%]	4.74e+3±3.60e+2
150	7.40e+3±1.96e+1 ↑ [1.35%]	7.39e+3±2.20e+1 ↑ [1.49%]	7.50e+3±1.25e+1
200	9.89e+3±2.43e+1 ↑ [1.11%]	9.85e+3±3.31e+1 ↑ [1.52%]	1.00e+4±1.86e+1
250	1.23e+4±4.82e+1 ↑ [1.63%]	1.23e+4±3.38e+1 ↑ [1.63%]	1.25e+4±2.56e+2
300	1.46e+4±6.92e+1 ↑ [1.37%]	1.46e+4±5.07e+1 ↑ [1.37%]	1.48e+4±4.18e+1
350	1.69e+4±1.24e+2 ↑ [1.78%]	1.70e+4±7.84e+1 ↑ [1.18%]	1.72e+4±5.82e+1
400	1.92e+4±1.04e+2 ↑ [1.56%]	1.93e+4±8.05e+1 ↑ [1.04%]	1.95e+4±1.03e+2
450	1.94e+4±2.21e+2 ↑ [2.58%]	1.95e+4±1.67e+2 ↑ [2.05%]	1.99e+4±2.02e+2
500	1.95e+4±1.56e+2 ↑ [2.05%]	1.96e+4±1.60e+2 ↑ [1.53%]	1.99e+4±1.69e+2
↑ / ↓ / ≈	9/0/1	10/0/0	

5. Conclusion

In this paper, a new DE algorithm (called DEMiDRA) was proposed to jointly optimize the mining decision and resource allocation for a MEC-enabled wireless blockchain network. The main characteristics of DEMiDRA can be summarized as follows:

- First, a new encoding scheme was proposed, in which each individual represents the resource allocation of a participating miner and the whole population represents the resource allocations of all participating miners. As a result, the redundant search space can be removed and the dimension of the search space is reduced to two.

- Afterward, an adaptive population update strategy was proposed to optimize the number of participating miners and a tabu strategy was designed to prevent unpromising miners from being selected. By using these two strategies, the mining decision can be optimized.

- DE was utilized to optimize the resource allocations of participating miners.

DEMiDRA was applied to a set of instances with different scales and compared with DE, ACO_{MV}, and BOToP. The results demonstrated the effectiveness of DEMiDRA. It is worth noting that this paper assumes that IoTDs are homogeneous in the studied blockchain network. In the future, we will study the heterogeneous blockchain network.

Appendix A. Appendix

To ensure that $\mathcal{C}5$ in (11) is satisfied, at least $n_{min} = \lfloor \frac{4c_3^2 E_0 k_1 \min_{i\in\mathcal{N}}(D_i X_i)}{c_2^2} \rfloor$ miners are required to participate in mining.

Proof: From (10) and $\mathcal{C}5$ in (11), we have

$$\sum_{i\in\mathcal{N}'} \left(c_2 f_i - c_3 k_1 f_i^2 D_i X_i\right) \geq c_3 E_0. \tag{A.1}$$

Then, let $g_i(f_i) = c_2 f_i - c_3 k_1 f_i^2 D_i X_i$. By substituting $g_i(f_i)$ to (A.1), we have

$$\sum_{i\in\mathcal{N}'} g_i(f_i) \geq c_3 E_0. \tag{A.2}$$

The first-order and second-order derivatives of $g_i(f_i)$ are $g_i'(f_i) = c_2 - 2c_3 k_1 D_i X_i f_i$ and $g_i''(f_i) = -2c_3 k_1 D_i X_i$, respectively. When $g_i'(f_i) = 0$, we have

$$f_i^* = \frac{c_2}{2c_3 k_1 D_i X_i}. \tag{A.3}$$

Due to $g_i''(f_i) < 0$, $g_i(f_i) \leq g_i(f_i^*) = \frac{c_2^2}{4c_3 k_1 D_i X_i}$. Thus, we have

$$\sum_{i\in\mathcal{N}'} g_i(f_i) \leq \sum_{i\in\mathcal{N}'} \frac{c_2^2}{4c_3 k_1 D_i X_i} \quad \leq \frac{n' c_2^2}{4c_3 k_1 \min_{i\in\mathcal{N}}(D_i X_i)}. \tag{A.4}$$

According to (A.2) and (A.4), $c_3 E_0 \leq \frac{n' c_2^2}{4c_3 k_1 \min_{i\in\mathcal{N}}(D_i X_i)}$. Re-arranging this inequality, we can obtain

$$n' \geq \frac{4c_3^2 E_0 k_1 \min_{i\in\mathcal{N}}(D_i X_i)}{c_2^2} \quad \geq \lfloor \frac{4c_3^2 E_0 k_1 \min_{i\in\mathcal{N}}(D_i X_i)}{c_2^2} \rfloor \tag{A.5}$$

where $\lfloor \cdot \rfloor$ denotes the rounding down operator. □

References

Ali, M.S., Vecchio, M., Pincheira, M., Dolui, K., Antonelli, F., Rehmani, M.H., 2019. Applications of blockchains in the Internet of Things: A comprehensive survey. IEEE Communications Surveys Tutorials 21, 1676–1717.

Azzi, R., Chamoun, R.K., Sokhn, M., 2019. The power of a blockchain-based supply chain. Computers & Industrial Engineering 135, 582 – 592.

Böhme, R., Christin, N., Edelman, B., Moore, T., 2015. Bitcoin: Economics, technology, and governance. Journal of Economic Perspectives 29, 213–38.

Chen, X., 2015. Decentralized computation offloading game for mobile cloud computing. IEEE Transactions on Parallel and Distributed Systems 26, 974–983.

Chu, X., Gao, D., Cheng, S., Wu, L., Chen, J., Shi, Y., Qin, Q., 2019. Worker assignment with learning-forgetting effect in cellular manufacturing system using adaptive memetic differential search algorithm. Computers & Industrial Engineering 136, 381 – 396.

Das, S., Suganthan, P.N., 2011. Differential evolution: A survey of the state-of-the-art. IEEE Transactions on Evolutionary Computation 15, 4–31.

Deb, K., 2000. An efficient constraint handling method for genetic algorithms. Computer Methods in Applied Mechanics and Engineering 186, 311 – 338.

Guo, F., Zhang, H., Ji, H., Li, X., Leung, V.C.M., 2018. An efficient computation offloading management scheme in the densely deployed small cell networks with mobile edge computing. IEEE/ACM Transactions on Networking 26, 2651–2664.

Houy, N., 2016. The bitcoin mining game. Ledger 1, 53–68.

Huang, P., Wang, Y., Wang, K., Liu, Z., 2019. A bilevel optimization approach for joint offloading decision and resource allocation in cooperative mobile edge computing. IEEE Transactions on Cybernetics to be published, doi:10.1109/TCYB.2019.2916728.

Huang, P., Wang, Y., Wang, K., Yang, K., 2020. Differential evolution with a variable population size for deployment optimization in a uav-assisted iot data collection system. IEEE Transactions on Emerging Topics in Computational Intelligence 4, 324–335.

Jiang, F., Wang, K., Dong, L., Pan, C., Xu, W., Yang, K., 2019. Deep learning based joint resource scheduling algorithms for hybrid MEC networks. IEEE Internet of Things Journal to be published, doi:10.1109/JIOT.2019.2954503.

Jiao, Y., Wang, P., Niyato, D., Suankaewmanee, K., 2019. Auction mechanisms in cloud/fog computing resource allocation for public blockchain networks. IEEE Transactions on Parallel and Distributed Systems 30, 1975–1989.

Kiayias, A., Koutsoupias, E., Kyropoulou, M., Tselekounis, Y., 2016. Blockchain mining games, in: Proceedings of the 2016 ACM Conference

on Economics and Computation, Association for Computing Machinery, New York, USA. pp. 365–382.

Leng, J., Yan, D., Liu, Q., Xu, K., Zhao, J.L., Shi, R., Wei, L., Zhang, D., Chen, X., 2020. Manuchain: Combining permissioned blockchain with a holistic optimization model as bi-level intelligence for smart manufacturing. IEEE Transactions on Systems, Man, and Cybernetics: Systems 50, 182–192.

Liao, T., Socha, K., Montes de Oca, M.A., Stützle, T., Dorigo, M., 2014. Ant colony optimization for mixed-variable optimization problems. IEEE Transactions on Evolutionary Computation 18, 503–518.

Liu, J., Wang, Y., Xin, B., Wang, L., 2020. A biobjective perspective for mixed-integer programming. IEEE Transactions on Systems, Man, and Cybernetics: Systems to be published, doi:10.1109/TSMC.2020.3043642.

Liu, M., Yu, F.R., Teng, Y., Leung, V.C.M., Song, M., 2018a. Computation offloading and content caching in wireless blockchain networks with mobile edge computing. IEEE Transactions on Vehicular Technology 67, 11008–11021.

Liu, X., Wang, W., Niyato, D., Zhao, N., Wang, P., 2018b. Evolutionary game for mining pool selection in blockchain networks. IEEE Wireless Communications Letters 7, 760–763.

Luo, S., Liang, W., Zhao, G., 2020. Hybrid pso-wdba method for the site selection of tailings pond. Computers & Industrial Engineering 143, 106429.

Luong, N.C., Xiong, Z., Wang, P., Niyato, D., 2018. Optimal auction for edge computing resource management in mobile blockchain networks: A deep learning approach, in: 2018 IEEE International Conference on Communications (ICC), pp. 1–6.

Tang, C., Li, C., Yu, X., Zheng, Z., Chen, Z., 2019. Cooperative mining in blockchain networks with zero-determinant strategies. IEEE Transactions on Cybernetics to be published, doi:10.1109/TCYB.2019.2915253.

Wang, B., Li, H., Li, J., Wang, Y., 2019a. Composite differential evolution for constrained evolutionary optimization. IEEE Transactions on Systems, Man, and Cybernetics: Systems 49, 1482–1495.

Wang, K., Huang, P., Yang, K., Pan, C., Wang, J., 2019b. Unified offloading decision making and resource allocation in me-ran. IEEE Transactions on Vehicular Technology 68, 8159–8172.

Wang, S., Taha, A.F., Wang, J., Kvaternik, K., Hahn, A., 2019c. Energy crowdsourcing and peer-to-peer energy trading in blockchain-enabled smart grids. IEEE Transactions on Systems, Man, and Cybernetics: Systems 49, 1612–1623.

Wang, X., Dong, Z., Tang, L., 2018a. Multiobjective differential evolution with personal archive and biased self-adaptive mutation selection. IEEE Transactions on Systems, Man, and Cybernetics: Systems to be published, doi:10.1109/TSMC.2018.2875043.

Wang, Y., Liu, H., Long, H., Zhang, Z., Yang, S., 2018b. Differential evo-

lution with a new encoding mechanism for optimizing wind farm layout. IEEE Transactions on Industrial Informatics 14, 1040–1054.

Xiang, S., Xing, L., Wang, L., Zou, K., 2019. Comprehensive learning pigeon-inspired optimizationwith tabu list. SCIENCE CHINA Information Sciences 62, 070208.

Xiong, Z., Feng, S., Wang, W., Niyato, D., Wang, P., Han, Z., 2019. Cloud/fog computing resource management and pricing for blockchain networks. IEEE Internet of Things Journal 6, 4585–4600.

Xiong, Z., Zhang, Y., Niyato, D., Wang, P., Han, Z., 2018. When mobile blockchain meets edge computing. IEEE Communications Magazine 56, 33–39.

Yi, J.H., Xing, L.N., Wang, G.G., Dong, J., Vasilakos, A.V., Alavi, A.H., Wang, L., 2020. Behavior of crossover operators in nsga-iii for large-scale optimization problems. Information Sciences 509, 470 – 487.

Yuan, Y., Wang, F., 2018. Blockchain and cryptocurrencies: Model, techniques, and applications. IEEE Transactions on Systems, Man, and Cybernetics: Systems 48, 1421–1428.

Zhang, Y., Zhang, P., Tao, F., Liu, Y., Zuo, Y., 2019. Consensus aware manufacturing service collaboration optimization under blockchain based industrial internet platform. Computers & Industrial Engineering 135, 1025 – 1035.

Zhou, S., Xing, L., Zheng, X., Du, N., Wang, L., Zhang, Q., 2019. A

self-adaptive differential evolution algorithm for scheduling a single batch-processing machine with arbitrary job sizes and release times. IEEE Transactions on Cybernetics to be publised, doi:10.1109/TCYB.2019.2939219.