HOSTED BY

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

Optimizing jointly mining decision and resource allocation in a MEC-enabled blockchain networks

Mohamed Abdel-Basset [a], Reda Mohamed [a], Ibrahim M. Hezam [b,*], Karam M. Sallam [c,d,*], Ahmad M. Alshamrani [b], Ibrahim A. Hameed [e,*]

[a] Faculty of Computers and Informatics, Zagazig University, Zagazig, Sharqiyah 44519, Egypt
[b] Department of Statistics & Operations Research, College of Sciences, King Saud University, Riyadh, Saudi Arabia
[c] Department of Computer Science, University of Sharjah, Sharjah, United Arab Emirates
[d] Faculty of Science and Technology, School of IT and Systems, University of Canberra, Canberra, ACT 2601, Australia
[e] Department of ICT and Natural Sciences, Norwegian University of Science and Technology (NTNU), Ålesund, Norway

A R T I C L E I N F O

Article history:
Received 20 July 2023
Revised 30 August 2023
Accepted 20 September 2023
Available online 26 September 2023

Keywords:
Optimization
Blockchain
Metaheuristics
Swarm intelligence
Resource allocation
Real-world applications

A B S T R A C T

In this paper, several recently published metaheuristic algorithms are adapted to optimize the NP-hard problem of jointly mining decision and resource allocation in mobile edge computing (MEC) enabled blockchain networks under two different encoding schemes. The first scheme represents individuals in a way that incorporates the mining decisions, transmission power, and computing resources of all miners for each individual, with mining decisions determined by a binary vector whose values indicate whether miners partake in mining or not. While, the second scheme makes each individual accountable for the transmission power and computing resources of each participant miner, treating all individuals as a singular solution to the problem. Then the Nutcracker optimization algorithm and gradient-based optimizer are modified to propose two robust variants, MNOA and MGBO, respectively. We then combine MNOA and MGBO to create HNOA, which further optimizes the mining decision and resource allocation in this problem. HNOA and other variants are validated using nine instances with a range of 150 to 600 miners. HNOA is also compared to several competing optimizers to demonstrate its efficacy in terms of several performance metrics. The experimental findings show the superiority of the proposed algorithm.

1. Introduction

Over the past decade, blockchain networks have seen a meteoric rise in popularity due to their capacity to provide immutable ledgers in a decentralized manner and also serve as platforms for data-driven autonomous organizations (Wang et al., 2019). The blockchain networks were first used as the backbone of a public, distributed ledger system to handle asset transactions in the form of digital cryptocurrencies between Peer-to-Peer clients (Nakamoto, 2008). These networks are peer-to-peer and decentralized, thus, no central authority is needed to ensure a trustworthy ledger and safe financial dealings (Yuan and Wang, 2018). Because of this feature, the transactions that take place on the blockchain are secure, inexpensive, constant, fast, and tamper-proof (Yuan and Wang, 2018). These benefits have led to the use of blockchain networks in a variety of domains, including smart manufacturing, the Internet of Things (IoT), supply chain, and smart grid (Wang et al., 2021).

The elimination of all of the drawbacks associated with centralized systems is one of the primary reasons why the blockchain technology has the ability to make the world a smaller place. The mixed blockchain construction offers the potential for a large reduction in the required amount of storage space (Tian et al., 2019). Blockchain was designed to function as a platform for distributed storage to facilitate decentralized registries and to provide many methods for securing data through authentication, authorization, and verification (Malik and Sharma, 2023). The most

* Corresponding authors at: Department of Computer Science, University of Sharjah, Sharjah, United Arab Emirates (Karam M. Sallam).

E-mail addresses: mohamedbasset@ieee.org (M. Abdel-Basset), redamoh@zu.edu.eg (R. Mohamed), ialmishnanah@ksu.edu.sa (I.M. Hezam), KSallam@sharjah.ac.ae, karam.sallam@canberra.edu.au (K.M. Sallam), ahmadm@ksu.edu.sa (A.M. Alshamrani), ibib@ntnu.no (I.A. Hameed).

Peer review under responsibility of King Saud University. Production and hosting by Elsevier.

ELSEVIER Production and hosting by Elsevier

https://doi.org/10.1016/j.jksuci.2023.101779
1319-1578/© 2023 The Author(s). Published by Elsevier B.V. on behalf of King Saud University.

successful applications of blockchain technology to far are Ethereum and Bitcoin (Malik and Sharma, 2023).

Blockchain networks employ a data storage mechanism that utilizes interconnected blocks arranged in a linked list structure. Each block is comprised of a body and a header, where the header includes several components, such as Nonce, merkle tree, hash of the previous block, and timestamp, while the block body includes the transaction and transaction counter (Malik and Sharma, 2023; Abdelsalam et al., 2023). To add a new block to the chain, participants known as miners must undertake the Proof-of-Work (PoW) process, which involves generating a hash value that binds the new block to the preceding blocks in the chain. Upon completing the PoW puzzle, miners broadcast the solution to other participants on the network for verification (Wang et al., 2021). The new block is added to the chain if a majority of participants concur on the validity of the solution (Xiong et al., 2018). Notably, the mining process is computationally taxing and necessitates substantial computational resources due to the arduous computations required to generate anti-collision hash values. Despite this, miners often have limited access to resources (Wang et al., 2021).

Many studies have recently focused on the topic of blockchain network resource management. They pay special attention to accurately optimizing the mining decision to determine if a miner takes part in mining or not and allocating the resources to the participating miners (Wang et al., 2021). Over the last few decades, wireless blockchain networks that run on Internet of Things devices (IoTDs) have garnered a lot of attention (Ali et al., 2018). IoTDs, on the other hand, are unable to allow mining on local devices because of their restricted computational capabilities (Jiang et al., 2019). According to (Wang et al., 2019), mobile edge computing (MEC) is an exciting new technology that has the potential to improve the computational capabilities of miners/IoTDs by offloading the mining tasks to a MEC server. Several studies and experiments have been carried out on wireless blockchain networks integrated with MEC. For example, Wang, K., et al. (Liu et al., 2018) proposed a wireless blockchain framework enabled by MEC to offload computation-intensive mining tasks to nearby edge computing nodes to achieve the computational capabilities required during the mining process. Despite this, this framework did not take into account the MEC service provider (MSP)'s profit. Luong, N.C., et al. (Luong et al.) developed a deep learning model for resource allocation in the wireless blockchain networks enabled by MEC. Du, J., et al. (Du et al., 2021) suggested a deep reinforcement learning algorithm to determine resource allocation and price for maximizing all miners' total profit. Several other works in MEC-supported blockchain networks are extensively discussed in (Wang et al., 2021).

The resource allocations and mining decision in the wireless blockchain network enabled by MEC could be formulated as an optimization problem that could be optimized by the metaheuristic and evolutionary algorithms for maximizing the profits of all miners. There are several metaheuristic optimization algorithms proposed for tackling several optimization problems, some of them are the dwarf mongoose optimization algorithm (Agushaka et al., 2022), the gazelle optimization algorithm (Agushaka et al., 2023), the adaptive hybrid dandelion optimizer (Hu et al., 2023), and the opposition-based artificial hummingbird algorithm (Laith et al., 2023). Those algorithms in the literature could achieve outstanding outcomes for several optimization problems, so some researchers have recently paid attention to applying them for jointly optimizing the decision mining, transmission power, and computing resources of all miners (Wang et al., 2021). For instance, in (Wang et al., 2021), a new variant of the differential evolution technique known as DEMiDRA was proposed for optimizing resource allocation in the blockchain network. In DEMiDRA, each solution symbolizes the resource allocation of one of the participant miners, and the population as a whole is comprised of the resource allocations of all participant miners. Regarding optimizing the mining decision to enhance efficiency, the population size is updated based on three operators (Insertion, deletion, and replacement) within the optimization process to adjust the number of participating miners. In (Hussien et al., 2023), the Henry single gas solubility optimization algorithm was improved using the chaotic maps to present a new variant, namely CHSGSO, with better exploration and exploitation operator. This variant was applied for optimizing mining decisions and resource allocation in the blockchain network integrated with MEC to improve the efficiency of the mining process. In addition, Shijing Yuan et al. (Yuan et al., 2022) developed an optimization approach for an edge video streaming system that makes use of blockchain technology. The approach seeks to find the best balance between precision and power consumption by optimizing the offloading mode and allocating resources accordingly.

To the best of our knowledge, no study in the literature, with the exception of (Wang et al., 2021), has been proposed to optimize both the MSP's profit and the resource allocation in the wireless blockchain network enabled by MEC technology. Even though DEMiDRA was proposed to optimize those two issues, it still suffers from some shortcomings, like falling into local minima and low convergence speed, which prevent it from achieving highly accurate outcomes. In addition, a few studies have employed the metaheuristics algorithm for solving this problem. Therefore, in this paper, several recently published metaheuristic algorithms are adapted to tackle this problem under two different encoding schemes. The first scheme represents the individuals in a form that makes each individual includes the computing resources, transmission power, and mining decision of all miners. On the contrary, the second scheme makes each individual responsible for the computing resources and transmission power of a participant miner; hence, all individuals are considered a single solution to the tackled problem. Regarding optimizing mining decisions under this scheme, three operators are employed to adjust the population size to insert, delete, and replace a participating miner at each function evaluation in the hope of finding a near-optimal number of participating miners that could improve efficiency. Under the first scheme, the mining decision is based on a binary variable to determine if a miner partakes in mining or not. Since the majority of metaheuristics were designed to deal only with continuous problems, they could not be directly applied to tackle this problem. Therefore, various V-shaped and S-shaped transfer functions were investigated to find the best-performing one with the metaheuristic algorithms. However, we found that the performance of the classical metaheuristics still needs robust improvements. Therefore, both Nutcracker optimization algorithm and gradient-based optimizer are modified to propose two robust variants, namely MNOA and MGBO, respectively. Those variants have better exploration and exploitation operators. To further optimize the resource allocation and mining decision, both MNOA and MGBO is effectively integrated to present a new variant called HNOA. Under two encoding schemes, the proposed HNOA and the other variants are validated using nine instances with several miners ranging between 150 and 600. In addition, HNOA is compared to several rival optimizers to reveal its effectiveness in terms of several performance metrics. According to the experimental findings, the performance of HNOA under the second encoding scheme is significantly better than its performance under the first encoding scheme and the rival optimizers, like DEMiDRA. The main contributions of this study are as follows:

- Investigating the performance of several recently published metaheuristic algorithms for jointly optimizing resource allocation and mining decisions.

- Investigating the performance of two different encoding schemes with various adapted algorithms.
- Investigating the performance of various S-shaped and V-shaped transfer functions to pick the most effective one.
- Modifying both NOA and GBO to propose robust variants, namely MGBO and MNOA, with better exploration and exploitation operators for achieving better outcomes for the considered problem.
- To further enhance the outcomes, both MGBO and MNOA are effectively integrated to propose a better variant called HNOA.
- Validating HNOA and the other variants using nine instances with a number of miners ranging between 150 and 600
- Comparing HNOA to several rival optimizers, like DEMiDRA, to reveal its effectiveness in terms of several performance metrics
- Under the second encoding scheme, the experimental findings disclose the HNOA's superiority in comparison to all the rival optimizers.

The remaining sections of this paper are structured as that: Section 2: Briefly discusses the problem formulation; Section 3: overviews the classical NOA and GBO; Section 4: explains the proposed algorithm; Section 5: reports the experimental settings; Section 6: shows results and discussion; Section 7: presents the conclusion and future work.

2. System model and problem formulation

In Fig. 1, wireless blockchain network with MEC is depicted. This figure includes a set of m miners, denoted as $N = [1, 2, \cdots, m]$, to take part in the mining process (Liu et al., 2018). The miners, who decide to participate in the mining process, need to acquire computational resources from the MSP. After that, they need to offload their tasks to the MEC server. In this study, the mining task of the *ith* miner is represented by two factors:BS_i and Z_i, where they indicate the block size and computing intensity of the task allocated to the *ith* miner, respectively. In the blockchain network that was herein studied, the mining task is not deemed to be finished until all following three steps: offloading, mining, and propagation processes have been successfully performed. Only after all three steps have been successfully completed is the miner eligible to get a reward. In the event that a miner chooses not to partake in mining or is unable to successfully finish its mining task, the miner will not be eligible for any reward (Liu et al., 2018).

In this paper, a vector $D = [D_1, D_2, D_3, \cdots, D_m]$ is used to represent the mining decisions of m miners; each *ith* dimension in this vector includes either 0 to indicate that the *ith* miner does not participate in the mining or 1 to indicate that it decides to participate. Not all miners will participate in the mining process, so we represent the participating miners with a set $N\prime$, while the total miners are represented with a set N. The number $n\prime$ of participating miners could be easily computed as follows: $n' = \sum_{i \in N} D_i$. Additionally, each miner, who intends to partake in the mining process, needs two resources, including the MEC server's computing resources (CPU Cycles/s) and the IoTD's transmission power. Those resources are represented in this paper using two vectors: the first vector is represented as $p = [p_1, p_2, p_3, \cdots, p_{n'}]$ to include the transmission power allocated to the participant miners $(p_i(i \in N'))$, and the second vector is represented as $f = [f_1, f_2, f_3, \cdots, f_{n'}]$ to include the computing resources allocated to the participating miners $(f_i(i \in N'))$.

2.1. Offloading step

During this stage of the process, all the participating miners send their tasks simultaneously to the MEC server. The formula for expressing the rate of transmission of the ith participant miner is computed as follows:

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

where W represents the channel bandwidth, H_i indicates the channel state data for the ith participant miner, p_i refers to the transmission power allocated to ith participant miner, σ^2 represents the power of the background noise. Then, the amount of time needed for transmission and the amount of energy needed to complete the task for the ith participant miner can be computed respectively according to the following formula:

$$T_i^t = \frac{BS_i}{R_i}, \forall i \in N' \quad (2)$$

$$E_i^t = p_i T_i^t, \forall i \in N' \quad (3)$$

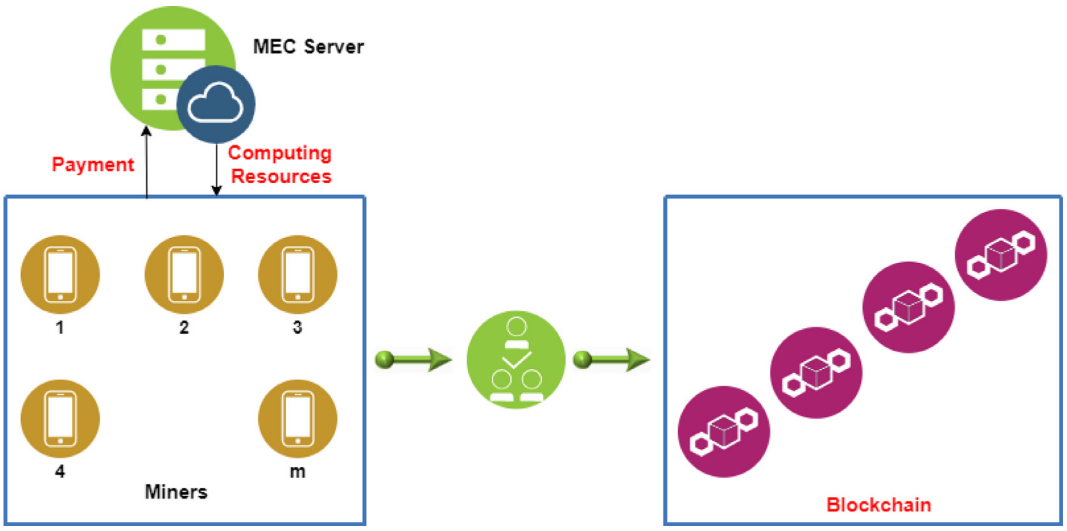

Fig. 1. Wireless blockchain network with mobile edge computing.

2.2. Mining step

During this stage, the responsibility of the MEC server is to carry out the mining tasks that have been communicated to the participants. The mining duration and the amount of energy required to complete the task for the ith participant miner are respectively stated as:

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

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

where k_1 represents the coefficient of effective capacitance. According to (Xiong et al., 2018), a miner on a blockchain network who completes a mining task in a shorter amount of time has a greater chance of being rewarded. As a result, the assumption has been made that the probability of a miner obtaining the reward is negatively proportional to its mining duration, which can be modeled using the following formula (Wang et al., 2021):

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

where k_2 represents the scaling factor.

2.3. Propagating step

Even if the miner successfully completes the mining phase, they will not be eligible for a reward if the result of the mining step takes a long time to spread throughout the network. The reasoning behind this is based on the fact that under these conditions, it is possible that the consensus is going to fail, and the block may be abandoned, which is referred to as orphaning (Liu et al., 2018). In blockchain networks, blocks are generated according to a Poisson process with a constant mean rate, and their propagation time T_i^o is proportional to the block size BS_i (Xiong et al., 2018). The orphaning likelihood of the ith miner is computed according to the following formula (Wang et al., 2021):

$$P_i^o = 1 - e^{-\lambda\left(T_i^o + T_i^s\right)}, \forall i \in N' \tag{7}$$

$$T_i^o = zBS_i \tag{8}$$

where z represents a specific delay factor and T_i^s indicates the time at which the ith participant miner begins mining its block. In this study, the mining task assigned to the ith participant miner will be carried out as soon as it is obtained by the MEC server; hence, $T_i^s = T_i^t$.

2.4. Profit model

The reward received by a miner is comprised of two parts: a variable reward and a fixed reward. Furthermore, a miner needs to consume some costs related to communication and computation. The profit earned by the ith miner who participates is determined by the following formula:

$$F_i^m = (w + \alpha BS_i)P_i^m(1 - P_i^o) - c_1 E_i^t - c_2 f_i \tag{9}$$

where w stands for the fixed reward, αBS_i stands for the variable reward, α stands for the variable reward factor, and c2 and c1 stand for the costs of the computation resources and transmission energy, respectively. Finally, the profit of all miners could be estimated using the following formula:

$$F^T = \sum_{i \in N'} F_i^m \tag{10}$$

Furthermore, the MSP generates revenue through the sale of computing resources to miners. However, it is imperative for the MSP to cover its energy consumption costs, which encompass both the energy consumed during mining and the energy consumed at no-load. Hence, the profitability of the MSP can be mathematically represented as follows:

$$F^{MSP} = \sum_{i \in N'} \left(c_2 f_i - c_3 E_i^m\right) - c_3 E_0 \tag{11}$$

where c_3 represents the cost of energy that is required by the MSP, while E_0 indicates the amount of energy that is consumed by the MSP at no-load.

2.5. Problem formulation

In the blockchain network that is studied in this paper, optimizing jointly both mining decisions and resource allocations represented in the transmission power and computing resource is considered a crucial problem that needs to be accurately tackled to optimize the aggregate profit of all miners. According to (Wang et al., 2021), the mathematical model of this problem is described as follows:

$$MaxF^T = \sum_{i \in N'} F_i^m \tag{12}$$

Subject to the following constraints:

$$C_1 : D_i \in \{0, 1\}, \forall i \in N$$

$$C_2 : p^{min} \le p_i \le p^{max}, \forall i \in N'$$

$$C_3 : f^{min} \le f_i \le f^{max}, \forall i \in N'$$

$$C4 : \sum_{i \in N'} f_i < f^{total}$$

$$C5 : F^{MSP} \ge 0$$

$$C_6 = T_i^t + T_i^m + T_i^o \le T_i^{max}, \forall i \in N'$$

We assume that all IoTDs in the examined blockchain network have the same minimum and maximum transmit power $\left[p^{min}, p^{max}\right]$ and compute resources $\left[f^{min}, f^{max}\right]$, respectively.

3. Methods

3.1. Gradient-based optimizer

Recently, a population-based optimization algorithm, referred to as the gradient-based optimizer (GBO), was introduced for solving global optimization and engineering optimization problems (Ahmadianfar et al., 2020). This algorithm makes use of the Newton method to guide the individuals participating in the optimization process to find the near-optimal solutions to several optimization problems. The GBO approach consists of two parts, each of which is discussed in greater depth below.

- **Gradient search rule (GSR) phase**

The GBO algorithm achieves rapid convergence and avoids entrapment in local optima by using gradient-based directions to ease agent motions during solution hunting. In this step, a vital variable designated ρ_1 is used to strike a balance between exploitation and exploration, allowing the algorithm to more swiftly and effectively avoid local minima and arrive at near-optimal solutions.

ρ_1 could be computed according to the following mathematical equation:

$$\rho_1 = 2 \times r \times \alpha - \alpha \tag{13}$$

$$\alpha = \left| \beta \times \sin\left(\frac{3\pi}{2} + \sin\left(\beta \times \frac{3\pi}{2}\right)\right) \right| \tag{14}$$

$$\beta = \beta_{min} + (\beta_{max} - \beta_{min}) \times \left(1 - \left(\frac{t}{t_{max}}\right)^3\right)^2 \tag{15}$$

where β_{max} and β_{min} includes constant numbers of 1.2 and 0.2, respectively, t represents the current function evaluation, and t_{max} indicates the maximum number of function evaluations. Afterward, The GSR based on ρ_1 could be computed according to the following mathematical equation to balance between exploration and exploitation of GBO:

$$GSR = rn \times \rho_1 \times \frac{2\Delta x \times X_n}{\overrightarrow{X}^t_w - \overrightarrow{X^*} + \varepsilon} \tag{16}$$

where rn is a numerical value generated at rondom using the normal distribution, ε is a so small value to avoid division by zero, $\overrightarrow{X}^t_w$ represents the worst solution, and $\overrightarrow{X^*}$ is the best-so-far solution. Δx is computed according to the following formula:

$$\Delta x = \overrightarrow{r} \times |S| \tag{17}$$

$$S = \frac{\left(X_*^{(t-1)} - X_a^{(t-1)}\right) + \delta}{2} \tag{18}$$

$$\delta = 2 \times r_2 \times \left(\left|\frac{X_a^{(t-1)} + X_b^{(t-1)} + X_c^{(t-1)} + X_d^{(t-1)}}{4}\right| - X_i^{(t-1)}\right) \tag{19}$$

where r_2 is a number selected randomly in the interval [0, 1] according to the uniform distribution; a, b, c, and d are the indices of four solutions selected at random from the current population. In accordance with GSR, a new solution $X1_i^{(t)}$ for the *ith* individual could be generated as described in the following mathematical formula:

$$X1_i^{(t)} = X_i^{(t-1)} - GSR \tag{20}$$

However, this newly generated solution still needs an enhancement to exploit the search space extensively. Therefore, the movement direction (DM) was utilized to improve the convergence speed of GBO for achieving the desired solutions in a less number of function evaluations. $X1_i^{(t)}$ based on both GSR and DM could be computed using the following formula:

$$\overrightarrow{X1}^{t+1}_i = \overrightarrow{X}^t_i - GSR + DM \tag{21}$$

$$DM = r \times \rho_2 \times \left(\overrightarrow{X^*} - \overrightarrow{X}^t_i\right) \tag{22}$$

$$\rho_2 = 2 \times r \times \alpha - \alpha \tag{23}$$

However, in a new attempt to improve the exploration capabilities of GBO, according to (Ahmadianfar et al., 2020); $X1_i^{(t)}$ could be related with the newton method according to the following formula:

$$\overrightarrow{X1}^t_i = \overrightarrow{X}^t_i - rn \times \rho_1 \times \frac{2\Delta x * X_n}{yp_i^t - yq_i^t + \varepsilon} + r \times \rho_2 \times \left(\overrightarrow{X^*} - \overrightarrow{X}^t_i\right) \tag{24}$$

where $yq_i^{(t-1)}$ and $yp_i^{(t-1)}$ could be computed using the following formula:

$$yp_i^{(t-1)} = r\left(\frac{\left[z_i^{(t-1)} + x_i^{(t-1)}\right]}{2} + r * \Delta x\right) \tag{25}$$

$$yq_i^{(t-1)} = r\left(\frac{\left[z_i^{(t-1)} + x_i^{(t-1)}\right]}{2} - r * \Delta x\right) \tag{26}$$

where $z_i^{(t-1)}$ is mathematically defined as follows:

$$z_i^{(t-1)} = x_i - \overrightarrow{rn} \times \frac{2\Delta x \times x_n}{\overrightarrow{X}^t_w - \overrightarrow{X^*} + \varepsilon} \tag{27}$$

where $\overrightarrow{rn}$ is a vector assigned with numbers that are randomly obtained according to the normal distribution. $\overrightarrow{X1}^{t+1}_i$ is employed for exploring the search space as much as possible to avoid getting stuck into local minima. To strength the GBO's exploitation operator, an additional solution $\overrightarrow{X2}^{t+1}_i$ for the *ith* individual is generated to exploit the regions around the best-so-far solution in an attempt to accelerate the convergence speed towards the near-optimal solution. The mathematical formula that could be used for computing $\overrightarrow{X2}^{t+1}_i$ is defined as follows:

$$\overrightarrow{X2}^{t+1}_i = \overrightarrow{X^*} - r \times \rho_1 \times \frac{2\Delta x \times X_n}{yp_i^t - yq_i^t + \varepsilon} + r \times \rho_2 \times \left(\overrightarrow{X^*} - \overrightarrow{X}^t_i\right) \tag{28}$$

Ultimately, in accordance with (Ahmadianfar et al., 2020), the new solution of each individual could be generated based on three vectors, $X1_i^{(t)}$, $X2_i^{(t)}$, and $X3_i^{(t)}$, as shown in the following mathematical formula:

$$\overrightarrow{X}^{t+1}_i = r_a\left(r_b \times \overrightarrow{X1}^{t+1}_i + (1 - r_b) \times \overrightarrow{X2}^{t+1}_i\right) + (1 - r_a) \times \overrightarrow{X3}^{t+1}_i \tag{29}$$

$$\overrightarrow{X3}^{t+1}_i = \overrightarrow{X}^t_i - \rho_1 \times \left(\overrightarrow{X2}^{t+1}_i - \overrightarrow{X1}^{t+1}_i\right) \tag{30}$$

where r_a and r_b are two numerical values generated at random in the range [0, 1] based on the uniform distribution.

- **Local escaping operator '**

The GBO algorithm is combined with a unique operator called the local escaping operator (LEO) to further avoid being stuck in local minima and to speed up convergence towards the near-optimal solution. The mathematical model of this operator is defined as follows:

$$\overrightarrow{X}^{t+1}_i = \begin{cases} \overrightarrow{X}^t_i + f_1\left(u_1 \overrightarrow{x}^t_* - u_2 \overrightarrow{x}^{t+1}_k\right) \\ +f_2\rho_1\left(u_3\left(\overrightarrow{X2}^{t+1}_i - \overrightarrow{X1}^{t+1}_i\right)\right) \\ + \frac{u_2\left(\overrightarrow{x}^t_a - \overrightarrow{x}^t_b\right)}{2}, r < 0.5 and r_1 < pr \\ \overrightarrow{X^*} + f_1\left(u_1 \overrightarrow{X^*} - u_2 \overrightarrow{x}^{t+1}_k\right) \\ +f_2\rho_1\left(u_3\left(\overrightarrow{X2}^{t+1}_i - \overrightarrow{X1}^{t+1}_i\right)\right) \\ + \frac{u_2\left(\overrightarrow{x}^t_a - \overrightarrow{x}^t_b\right)}{2}, r \geq 0.5 and r_1 < pr \end{cases} \tag{31}$$

where *pr* is a predefined probability between 0 and 1 to determine the percent of applying LSO within the optimization process, f_1, and f_2 are two random numbers chosen randomly in the range [1, −1] according to the uniform distribution. u_2, u_1, and u_3 are computed according to the following formula:

$$u_1 = \begin{cases} 2r_1 if \mu_1 < 0.5 \\ 1 otherwise \end{cases} \tag{32}$$

$$u_2 = \begin{cases} r_1, if \mu_1 < 0.5 \\ 1, otherwise \end{cases} \tag{33}$$

$$u_3 = \begin{cases} r_1, if \mu_1 < 0.5 \\ 1, otherwise \end{cases} \tag{34}$$

where r_1 and μ_1 are numerical values generated randomly between 0 and 1 according to the uniform distribution. x_k^{t+1} is computed using the following mathematical equation:

$$\overrightarrow{x}_k^{t+1} = \begin{cases} \overrightarrow{x}_r, if \mu_2 < 0.5 \\ \overrightarrow{x}_a^t, otherwise \end{cases} \tag{35}$$

where x_r is an individual chosen at random within the lower bound and upper bound of the tackled optimization problems, as defined in the following formula:

$$\overrightarrow{x}_r = \overrightarrow{x}_l + \overrightarrow{r}\left(\overrightarrow{x}_u - \overrightarrow{x}_l\right) \tag{36}$$

where μ_2 is a numerical value generated randomly between 0 and 1 according to the uniform distribution, $\overrightarrow{x}_l$ and $\overrightarrow{x}_u$ represent the lower and upper bound of the tackled optimization problem. The steps of GBO is described in Algorithm 1.

Algorithm 1 (*GBO's Steps*).

```
Input: population size PS, and $T_{max}$;
  Output: $\overrightarrow{X^*}$
  1. Initialize N individuals, $X_i (i \in N)$.
  2. Evaluate each individual and identify the best and
     worst solutions
  3. t = 1; %% Function evaluation counter
  4. while (t < $T_{max}$)
  5.   for each i individual
  6.     Select randomly $a \neq b \neq c \neq d \neq i$.
  7.     // GSR strategy.
  8.     Generate $\overrightarrow{X}_i^{t+1}$ by (29)
  9.     // LEO Strategy
  10.    Generate $X_i^{t+1}$ by (31)
  11.    t++ %% Increment the current function evaluation
  12.  End
  13.    Update $\overrightarrow{X^*}$ and $\overrightarrow{X}_w^t$
  14.  End
```

3.2. Nutcracker optimization algorithm

In order to solve global optimization and technical optimization challenges, the Nutcracker optimization Algorithm (NOA) was recently introduced (Abdel-Basset, 2023). The nutcracker's foraging, storing, and retrieval of pine seeds are all modeled in NOA. There are two main types of nutcracker behavior, and they both happen at different times. The NOA's steps are described in Algorithm 2. Following is a detailed explanation of the two primary strategies (Foraging and storage strategy and Cache-search and recovery strategy) upon which the mathematical model of NOA is built.

- Foraging and storage strategy
- Foraging stage: Exploration phase 1

This is considered the initial exploration stage in NOA. Initially, the positions of the nutcrackers within the search space are determined at random. Afterwards, each nutcracker begins by examining the seed cone's initial positions. If the nutcracker discovers viable seeds, it will transfer them to a storage location and conceal them in a cache. If it is unable to find any viable seeds in one pine cone, the nutcracker will move on to another area with trees of a different species. This pattern of behavior can be modeled as follows:

$$\overrightarrow{X}_i^{t+1} = \begin{cases} X_{i,j}^t if \tau_1 < \tau_2 \\ \begin{cases} X_m^t + \gamma \cdot \left(X_{a,j}^t - X_{b,j}^t\right) \\ +\mu.\left(r^2.x_{u,j} - x_{l,j}\right), if t \leq T_{max}/2.0 \\ X_{C,j}^t + \mu \cdot \left(X_{a,j}^t - X_{b,j}^t\right) \\ +\mu.(r_1 < \delta).\left(r^2.x_{u,j} - x_{l,j}\right), Otherwise \end{cases} otherwise \end{cases} \tag{37}$$

where $X_{i,j}^t$ indicates the *jth* decision variable of the *ith* solution at function evaluation *t*; γ includes a numerical value selected at random according to the levy-flight; *a*, *b*, and *c* are the indices of three solutions selected at random from the current population; τ_1, τ_2, *r*, and r_1 are numbers chosen at random in the interval [0, 1] according to the uniform distribution; $X_{m,j}^t$ is the mean of the *jth* dimensions in all individuals; and μ is a number selected at random according to the following formula:

$$\mu = \begin{cases} \tau_3 if r_1 < r_2 \\ \tau_4 if r_2 < r_3 \\ \tau_5 if r_1 < r_3 \end{cases} \tag{38}$$

where r_2, r_3, and τ_4 are three numerical values chosen at random in the interval [0, 1] according to the uniform distribution. τ_4 is a numerical value selected at random based on the normal distribution, and τ_5 is a numerical value selected at random based on the levy-flight.

- **Storage stage: Exploitation phase 1**

In this stage, the nutcrackers gather and store pine seed harvests. This behavior is represented by the equation listed below:

$$\overrightarrow{X}_i^{t+1} = \begin{cases} \overrightarrow{X}_i^t + \mu \cdot \left(\overrightarrow{X^*} - \overrightarrow{X}_i^t\right) \cdot |\lambda| + r_1 \cdot \left(\overrightarrow{X}_a^t - \overrightarrow{X}_b^t\right) if \tau_1 < \tau_2 \\ \overrightarrow{X^*} + \mu \cdot \left(\overrightarrow{X}_a^t - \overrightarrow{X}_b^t\right) if \tau_1 < \tau_3 \\ \overrightarrow{X^*} \cdot l Otherwise \end{cases} \tag{39}$$

where λ includes a numerical value randomly selected based on the lévy flight. *l* represents a controlling factor that includes numerical value reducing linearly from 1 to 0. The following formula balances exploitation and exploration operators through a tradeoff between the first exploitation and exploration operators of NOA during the optimization process:

$$\overrightarrow{X}_i^{t+1} = \begin{cases} Eq.(37), if \varphi < P_{a_1} \\ Eq.(39), otherwise \end{cases} \tag{40}$$

where φ is a numerical value chosen at random between 0 and 1 according to the uniform distribution, and P_{a_1} is estimated using the following formula:

$$P_{a_1} = 1 - \frac{t}{T_{max}} \tag{41}$$

- Cache-search and recovery strategy
- Cache-search stage: Exploration phase 2

As soon as winter approaches and the trees lose their leaves, nutcrackers begin searching for their hidden food stores. This is the second exploration stage for the NOA. The nutcrackers use a system that taps into their spatial memories to lead them to their hidden stores of nuts. Many objects may be used by nutcrackers as signals for a single cache. To keep things straightforward, the authors assumed that each cache has only two markers or signals. As shown in the matrix below, each nutcracker in NOA uses two reference points (RPs) for each cache as signals:

$$\text{RPs} = \begin{bmatrix} \overrightarrow{RP}^{t}_{1,1} & \overrightarrow{RP}^{t}_{1,2} \\ \vdots & \vdots \\ \overrightarrow{RP}^{t}_{i,1} & \overrightarrow{RP}^{t}_{i,2} \\ \vdots & \vdots \\ \overrightarrow{RP}^{t}_{N,1} & \overrightarrow{RP}^{t}_{N,1} \end{bmatrix} \tag{42}$$

where $\overrightarrow{RP}^{t}_{i,1}$ stands for the first RP for the i th solution at function evaluation t. Two separate equations were devised to generate the first and second RPs, respectively, in order to enhance the nutcracker searching process for hidden caches. The first RP can be derived using (43), and the second RP can be derived using (44).

$$\overrightarrow{RP}^{t}_{i,1} = \begin{cases} \overrightarrow{X}^{t}_{i} + \alpha \cdot \cos(\theta) \cdot \left(\left(\overrightarrow{X}^{t}_{a} - \overrightarrow{X}^{t}_{b} \right) \right) + \alpha \cdot RP, if\theta = \pi/2 \\ \overrightarrow{X}^{t}_{i} + \alpha \cdot \cos(\theta) \cdot \left(\left(\overrightarrow{X}^{t}_{a} - \overrightarrow{X}^{t}_{b} \right) \right), otherwise \end{cases} \tag{43}$$

$$\overrightarrow{RP}^{t}_{i,2} = \begin{cases} \overrightarrow{X}^{t}_{i} + \left(\alpha \cdot cos(\theta) \cdot \left(\left(\overrightarrow{x_u} - \overrightarrow{x_l} \right) \cdot \tau_3 + \overrightarrow{x_l} \right) + \alpha \cdot RP \right) \\ \cdot \overrightarrow{U_2}, if\theta = \pi/2 \\ \overrightarrow{X}^{t}_{i} + \alpha \cdot cos(\theta) \cdot \left(\left(\overrightarrow{x_u} - \overrightarrow{x_l} \right) \cdot \tau_3 + \overrightarrow{x_l} \right) \cdot \overrightarrow{U_2}, otherwise \end{cases} \tag{44}$$

$$\overrightarrow{U_1} = \begin{cases} 1 \overrightarrow{r_2} < P_{rp} \\ 0 otherwise \end{cases} \tag{45}$$

where $\overrightarrow{r_2}$ represents a vector including numerical values generated at random between 0 and 1 based on the uniform distribution; θ is a number chosen at random in the range [0, 1] according to the uniform distribution; and P_{rp} represents the exploration rate and is predefined between 0 and 1; and α is estimated by the following equation:

$$\alpha = \begin{cases} \left(1 - \frac{t}{T_{max}}\right)^{2\frac{t}{T_{max}}}, if r_1 > r_2 \\ \left(\frac{t}{T_{max}}\right)^{\frac{2}{t}}, otherwise \end{cases} \tag{46}$$

NOA employs the following formula to activate the spatial memory of the *ith* nutcracker for the first RP:

$$\overrightarrow{X}^{t+1}_{i} = \begin{cases} \overrightarrow{X}^{t}_{i}, iff\left(\overrightarrow{X}^{t}_{i}\right) < if\left(\overrightarrow{RP}^{t}_{i,1}\right) \\ \overrightarrow{RP}^{t}_{i,1}, otherwise \end{cases} \tag{47}$$

- **Recovery stage: Exploitation phase 2**

In the first scenario, a nutcracker recalls the cache's location using the initial RP. The following mathematical model illustrates this behavior:

$$X^{t+1}_{ij} = \begin{cases} X^{t}_{ij}, if \tau_3 < \tau_4 \\ X^{t}_{ij} + r_1 \cdot \left(X^{*}_{j} - X^{t}_{i,j} \right) + r_2 \cdot \left(\overrightarrow{RP}^{t}_{i,1} - X^{t}_{c,j} \right), otherwise \end{cases} \tag{48}$$

where r_1, r_2, τ_3andτ_4 are numerical values chosen at random between 0 and 1 according to the uniform distribution. If a nutcracker is unable to access its cache using the first RP, it will use its spatial memory to move to the second RP, as shown in the following equation:

$$\overrightarrow{X}^{t+1}_{i} = \begin{cases} \overrightarrow{X}^{t}_{i}, iff\left(\overrightarrow{X}^{t}_{i}\right) < if\left(\overrightarrow{RP}^{t}_{i,2}\right) \\ \overrightarrow{RP}^{t}_{i,2}, otherwise \end{cases} \tag{49}$$

Eq. (48) is altered in NOA to search for the cache's location using the second RP, as defined in the following equation:

$$X^{t+1}_{ij} = \begin{cases} X^{t}_{ij}, if \tau_5 < \tau_6 \\ X^{t}_{ij} + r_1 \cdot \left(X^{*}_{j} - X^{t}_{ij} \right) + r_2 \cdot \left(\overrightarrow{RP}^{t}_{i,2} - X^{t}_{cj} \right), otherwise \end{cases} \tag{50}$$

where $\tau_5$andτ_6 are two numbers selected at random between 0 and 1 according to the uniform distribution. The following equation depicts the tradeoff between the second and first RPs in recovery behavior:

$$\overrightarrow{X}^{t+1}_{i} = \begin{cases} Eq.(48), if \tau_7 < \tau_8 \\ Eq.(50), otherwise \end{cases} \tag{51}$$

where τ_7 and τ_8 are two numbers selected at random between 0 and 1 according to the uniform distribution. In order to locate the cache, the spatial memory between the second RP, the first RP, and the current position is activated using the following equation:

$$\overrightarrow{X}^{t+1}_{i} = \begin{cases} Eq.(47), iff(Eq.(47)) < f(Eq.(49)) \\ Eq.(49), otherwise \end{cases} \tag{52}$$

The recovery and cache-search phases are then randomly switched using the following formula in order to strike a balance between exploitation and exploration operators:

$$\overrightarrow{X}^{t+1}_{i} = \begin{cases} Eq.(51), if \phi < P_{a_2} \\ Eq.(52), otherwise \end{cases} \tag{53}$$

where P_{a_2} is a predetermined value in the range between 0 and 1 that represents the likelihood of reaching the exploitation stage throughout the optimization process. However, in the NOA algorithm, the nutcracker will remain in its existing position if the quality of its current solution is better than that of the new solution; otherwise, the new solution is used in the next generation instead of the current solution, as defined in the following equation:

$$\vec{X}_i^{t+1} = \begin{cases} \vec{X}_i^{t+1}, iff\left(\vec{X}_i^{t+1}\right) < f\left(\vec{X}_i^{t}\right) \\ \vec{X}_i^{t}, otherwise \end{cases} \tag{54}$$

Algorithm 2 (*Steps of NOA*).

```
Input: population size PS, $\vec{x_l}$, $\vec{x_u}$, and $T_{max}$;
  Output: $\vec{X^*}$
  1. Initialize $PS$ individuals, $X_i (i \in N)$.
  2. Evaluate each individual and identify the best and worst solutions
  3. t = 1; %% Function evaluation counter
  4. while $(t < T_{max})$
  5.   $\sigma$, $\sigma_1$, and $\varphi$: three numerical values generated at random in the interval 0 and 1.
  6.   If $\sigma < \sigma_1$ //* Foraging and storage strategy*//
  7.     for i = 1:N
  8.       if $\varphi < P_{a_1}$ /* Exploration phase1*/
  9.         Generate $\vec{X}_i^{t+1}$ using (37) and (54)
  10.      else /*Exploitation phase1*/
  11.        Generate $\vec{X}_i^{t+1}$ using (39) and (54)
  12.      end if
  13.      $t++$
  14.    end for
  15.  else //* Cache-search and recovery strategy *//
  16.    Calculate RP matrix using (42), (43) and (44).
  17.    for i = 1:N
  18.      if $\phi < P_{a_2}$ /*Exploitation phase2*/
  19.        Generate $\vec{X}_i^{t+1}$ using (51) and (54)
  20.      else /*Exploration phase2*/
  21.        Generate $\vec{X}_i^{t+1}$ using (52) and (54)
  22.      end if
  23.    $t++$
  24.  end for
  25. end while
```

4. Proposed algorithm

4.1. Solution representation

The first step for solving the majority of the optimization problems is based on picking the optimal encoding scheme that could efficiently represent the solutions inside the population in the hope of fulfilling two purposes: (1) Alleviating the high dimensionality problem and (2) Making all variables in the solution homogenous to avoid mixed-variable problems that might deteriorate the performance of the optimization technique. The problem tackled in this paper is considered a mixed-variable optimization problem because its solutions include binary and continuous variables. The binary variables symbolized as D aim to determine whether miners take part in the mining process or not, while the continuous variables represent the computing resources and transmission power for each participant miner. In the literature, two encoding schemes were proposed to represent the solutions to this problem. The first scheme is widely used, while the second scheme is recently proposed and still needs further investigation to confirm its efficiency with the recently published metaheuristics. In detail, those two encoding schemes are discussed in the next section.

- **Commonly-used encoding scheme** (Wang, 2021)**:** This scheme has been widely considered in the literature for solving this problem, but it is not considered optimal because it suffers from high dimensionality problems in addition to mixing both binary and continuous variables in the same solution. In a more sense, it integrates the computing resources, transmission power, and mining decision for each miner in each individual inside the population. Hence, if the number of miners is m, the length of dimensions for each solution must be $3m$, as depicted in Fig. 2. This is not considered a feasible manner, especially with increasing the number of miners. Furthermore, this scheme mixes both binary and continuous variables in the same solution, and hence the metaheuristic algorithms could not be directly adopted for solving this problem because their majority is only designed for continuous optimization problems. In our study, we investigate the performance of this scheme with some recently-published metaheuristic algorithms to experimentally show its poor performance. To adopt these algorithms for the binary variables, one of the S-shaped and V-shaped transfer functions, which are modeled in Table 1 and depicted in Figs. 3 and 4, is used to normalize the continuous values of

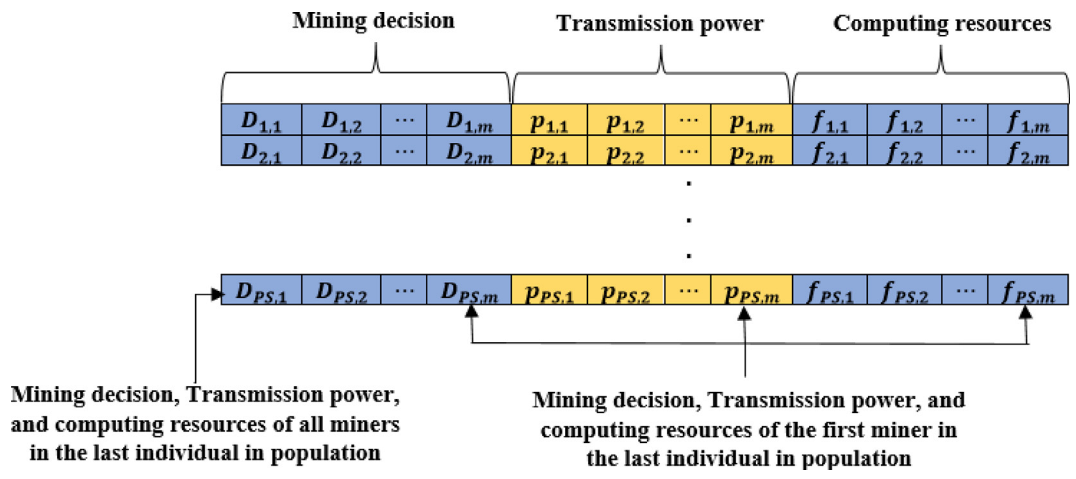
Mining decision
Transmission power
Computing resources
$D_{1,1}$ $D_{1,2}$ ⋯ $D_{1,m}$ $p_{1,1}$ $p_{1,2}$ ⋯ $p_{1,m}$ $f_{1,1}$ $f_{1,2}$ ⋯ $f_{1,m}$
$D_{2,1}$ $D_{2,2}$ ⋯ $D_{2,m}$ $p_{2,1}$ $p_{2,2}$ ⋯ $p_{2,m}$ $f_{2,1}$ $f_{2,2}$ ⋯ $f_{2,m}$
$D_{PS,1}$ $D_{PS,2}$ ⋯ $D_{PS,m}$ $p_{PS,1}$ $p_{PS,2}$ ⋯ $p_{PS,m}$ $f_{PS,1}$ $f_{PS,2}$ ⋯ $f_{PS,m}$
Mining decision, Transmission power, and computing resources of all miners in the last individual in population
Mining decision, Transmission power, and computing resources of the first miner in the last individual in population

Fig. 2. Solutions representation using the widely-used encoding scheme.

Table 1
Mathematical model of S-shaped and V-shaped.

S-Shaped	V-Shaped
S1 $F\left(\overrightarrow{X}\right) = \frac{1}{1+e^{-\overrightarrow{X}}}$	V1 $F\left(\overrightarrow{X}\right) = \left\lvert \frac{2}{\pi}\,\mathrm{arcTan}\left(\frac{\pi}{2}\,\overrightarrow{X}\right) \right\rvert$
S2 $F\left(\overrightarrow{X}\right) = \frac{1}{1+e^{-2\overrightarrow{X}}}$	V2 $F\left(\overrightarrow{X}\right) = \left\lvert \tanh\left(\overrightarrow{X}\right) \right\rvert$
S3 $F\left(\overrightarrow{X}\right) = \frac{1}{1+e^{-\frac{\overrightarrow{X}}{2}}}$	V3 $F\left(\overrightarrow{X}\right) = \left\lvert \frac{a}{\sqrt{1+\overrightarrow{X}^2}} \right\rvert$
S4 $F\left(\overrightarrow{X}\right) = \frac{1}{1+e^{-\frac{\overrightarrow{X}}{3}}}$	V4 $F\left(\overrightarrow{X}\right) = \left\lvert \mathrm{erf}\left(\frac{\sqrt{\pi}}{2}\overrightarrow{X}\right) \right\rvert$

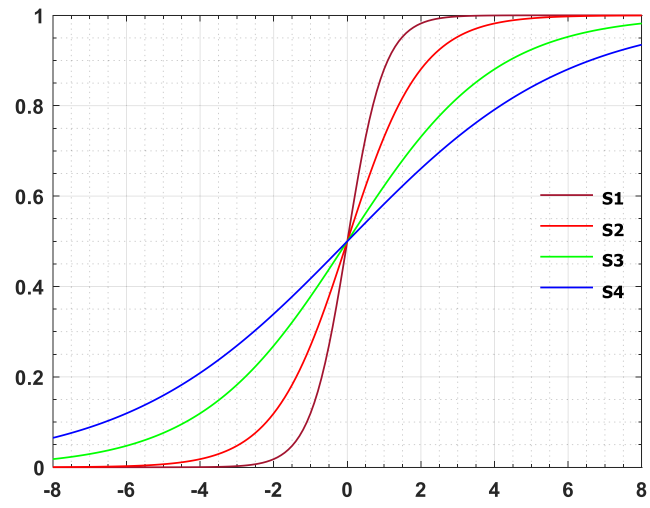

Fig. 3. S-shaped transfer functions.

the decision mining between 0 and 1. Then, they are compared to random value (*rand*) to convert them into binary values 0 and 1, as shown in (55). Those transfer functions are investigated with HNOA to pick the most effective one that could maximize its performance when jointly optimizing the resource allocation and mining decision.

$$\overrightarrow{X}_{bin}\left(\overrightarrow{X}\right) = \begin{cases} 1 \ if \ F\left(\overrightarrow{X}\right) \geq r \ and \\ 0 \ otherwise \end{cases} \tag{55}$$

- **Recently-proposed encoding scheme** (Wang, 2021)**:** This scheme has been recently proposed to overcome the drawbacks of the former scheme. In this scheme, the transmission power and computing resources for each miner are represented by a unique individual in the population; hence, if the number of miners is m, the number of dimensions in each solution is fixed to 2 and the population size (PS) is of m to make each individual responsible for only a miner, as depicted in Fig. 5. This scheme avoids the dimensionality problem because it fixes the number of dimensions to 2 and extends the individuals in the population according to the length of participating miners. In the end, all individuals are considered a solution to the tackled problem, and the profit of each individual or miner is computed and summed with the others to compute the total profit that represents the objective value to all individuals in the population. Notably, this encoding scheme takes mining decisions into account even though they are not explicitly encoded into population.

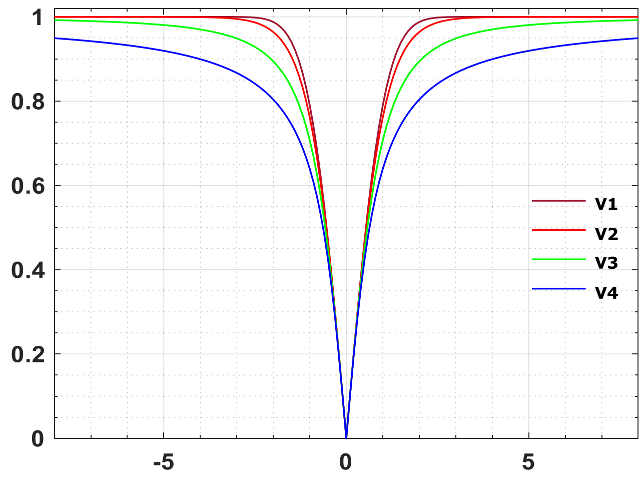

Fig. 4. V-shaped transfer functions.

4.2. Modified gradient-based optimizer

In GBO, a new solution for each individual could be generated based on relating three vectors, $X1_i^{t+1}$, $X2_i^{t+1}$, and $X3_i^{t+1}$ using random numbers as modeled in (29). However, using random numbers might deteriorate the stability of the algorithm, and hence the algorithm might lose its ability to find better solutions within the rest of the optimization process. To avoid this problem, in our proposed, Eq. (29) is modified to alleviate the randomization process as an attempt to aid GBO in achieving better outcomes within the optimization process; this new variant of GBO is called MGBO. This modification is based on removing the random number r_b from the equation and relate three vectors using only the random number r_a as shown in the following mathematical equation:

$$\overrightarrow{X}_i^{t+1} = r_a\left(\overrightarrow{X1}_i^{t+1} + \overrightarrow{X2}_i^{t+1}\right) + (1 - r_a) \times \overrightarrow{X3}_i^{t+1} \tag{56}$$

In this equation, if r_a is small, then the preference is toward the third vector $\overrightarrow{X3}_i^{t+1}$; otherwise, the preference is toward the addition of the other two vectors. This equation could achieve better outcomes than the original one because reducing the randomization numbers makes the optimization process dependent more on the individual's information in the population. For example, in the original equation, using two different random numbers might cause the algorithm to move in several directions within the optimization process and hence if the near-optimal solution is found in a specific direction. The algorithm could not follow it because the random numbers might differ. Therefore, reducing the random number in the equation aids the algorithm in following the experience of the individuals to achieve better outcomes. To further improve the MGBO's performance, the LEO operator is eliminated to minimize the population diversity within the optimization process as an attempt to accelerate the convergence speed towards the near-optimal solution. Finally, the MGBO's pseudocode is shown in Algorithm 3.

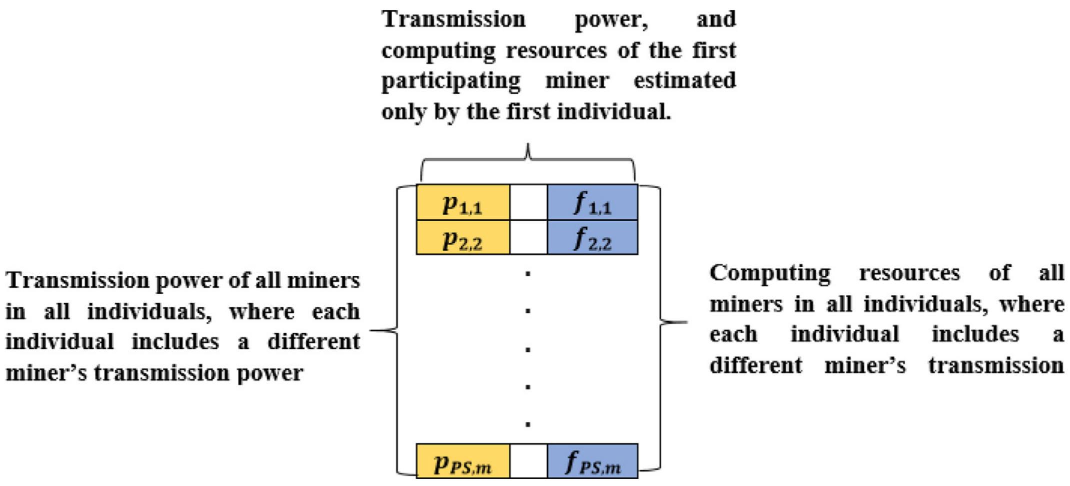
Transmission power, and computing resources of the first participating miner estimated only by the first individual.
$p_{1,1}$
$f_{1,1}$
$p_{2,2}$
$f_{2,2}$
$p_{PS,m}$
$f_{PS,m}$
Transmission power of all miners in all individuals, where each individual includes a different miner's transmission power
Computing resources of all miners in all individuals, where each individual includes a different miner's transmission

Fig. 5. Solution representation using the recently-proposed encoding scheme.

Algorithm 3 (*MGBO's Steps*).

```
Input: population size PS, and $T_{max}$;
Output: $\overrightarrow{X^*}$
1.  Initialize N individuals, $X_i (i \in N)$.
2.  Evaluate each individual and identify the best and worst solutions
3.  t = 1; %% Function evaluation counter
4.  while (t < $T_{max}$)
5.    for each i individual
6.      Select randomly $a \neq b \neq c \neq d \neq i$.
7.      // GSR strategy.
8.      Generate $\overrightarrow{X}_i^{t+1}$ by (56)
9.      t++ %% Increment the current function evaluation
10.   End
11.     Update $\overrightarrow{X^*}$ and $\overrightarrow{X}_w^t$
12.   End
```

4.3. Modified nutcracker optimization algorithm

NOA is based on two exploration operators and two exploitation operators. Those operators aid in achieving outstanding outcomes for several optimization problems. The no-free theorem says that whether the algorithm performs well on some optimization problems, it is not essential that it have the same performance on other optimization problems. Therefore, in this paper, we try to improve the performance of NOA for jointly optimizing the resource allocation and mining process by dispensing for some equations and observing its performance under others. After observing all possible combinations of the NOA equations, we found that NOA could perform well in this problem when reformulated (37) into (57) and formulated (39) into (58). Those equations are implemented in the same order as the classical NOA to propose

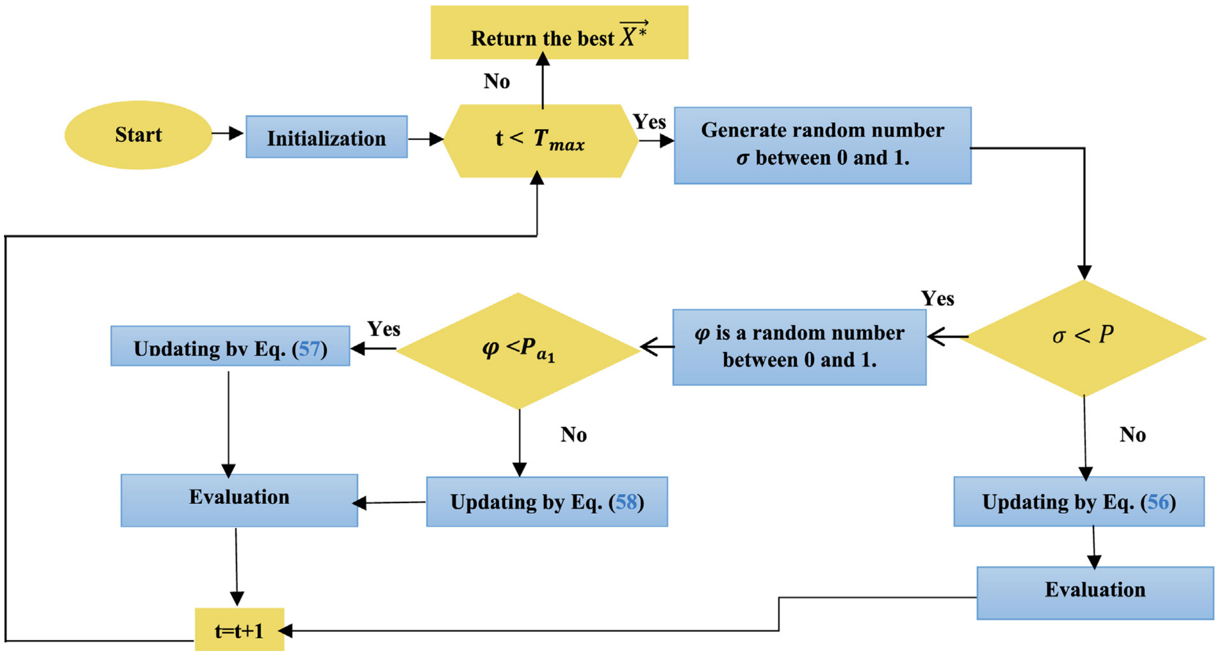

Fig. 6. Flowchart of HNOA.

a modified variant known as MNOA, as shown in algorithm 4. Regarding the other equations, we found that dispensing for them could significantly improve the performance of NOA

$$\overrightarrow{X}_i^{t+1} = \begin{cases} X_{i,j}^t if \tau_1 < \tau_2 \\ \left\{X_{C,j}^t + \mu \cdot \left(X_{a,j}^t - X_{b,j}^t\right) + \mu.(r_1 < \delta).(r^2.x_{u,j} - x_{l,j}\right), otherwise \end{cases} \quad (57)$$

$$\overrightarrow{X}_i^{t+1} = \begin{cases} \overrightarrow{X}_i^t + \mu \cdot \left(\overrightarrow{X^*} - \overrightarrow{X}_i^t\right) \cdot |\lambda| + r_1 \cdot \left(\overrightarrow{X}_a^t - \overrightarrow{X}_b^t\right) if \tau_1 < \tau_2 \\ \overrightarrow{X^*} + \mu \cdot \left(\overrightarrow{X}_a^t - \overrightarrow{X}_b^t\right) Otherwise \end{cases} \quad (58)$$

Algorithm 4 (*MNOA*).

```
Input: population size PS, $\overrightarrow{x_l}$, $\overrightarrow{x_u}$, and $T_{max}$;
  Output: $\overrightarrow{X^*}$
 1. Initialize N individuals, $X_i (i \in N)$.
 2. Evaluate each individual and identify the best and
    worst solutions
 3. t = 1; %% Function evaluation counter
 4. while ($t < T_{max}$)
 5.   $\sigma$, $\sigma_1$, and $\varphi$: three numerical values generated at
      random in the interval 0 and 1.
 6.   for i = 1:N
 7.     if $\varphi < P_{a_1}$ /* Exploration phase*/
 8.       Generate $\overrightarrow{X}_i^{t+1}$ using (57) and (54)
 9.     else /*Exploitation phase*/
10.       Generate $\overrightarrow{X}_i^{t+1}$ using (58) and (54)
11.     end if
12.     $t++$
13.   end for
14.   end while
```

4.4. Hybrid nutcracker optimization algorithm

MGBO has a robust exploration operator that enables it to extensively explore the search space for finding the most promising regions, which might involve a near-optimal solution. However, its ability to exploit the regions around the current solutions in the population is weak because its updating process for the current solution is based on the three vectors discussed before. On the contrary, MNOA has an outstanding exploitation operator, which enables accelerating the convergence speed by extensively focusing on the regions around the current solution and the best-so-far solutions. So, both MNOA and MGBO are effectively combined with a probability P to achieve a balance between the exploration and exploitation operators. This is done to avoid getting stuck in local optima and to speed up convergence so that the near-optimal solution can be found in a small number of function evaluations. Eq. (59) defines how both MGBO and MNOA are hybridized together to present a new strong variant called hybrid NOA (HNOA) for jointly optimizing resource allocation and mining decisions. The pseudocode of HNOA and its flowchart are presented in Algorithm 4 and Fig. 6, respectively.

$$\begin{cases} ExecuteMNOAif \sigma > P \\ ExecuteMGBO, otherwise \end{cases} \quad (53)$$

Algorithm 4 (*HNOA*).

```
Input: population size PS, $\overrightarrow{x_l}$, $\overrightarrow{x_u}$, and $T_{max}$;
  Output: $\overrightarrow{X^*}$
 1. Initialize N individuals, $X_i (i \in N)$.
 2. Evaluate each individual and identify the best and
    worst solutions
 3. t = 1; %% Function evaluation counter
 4. while ($t < T_{max}$)
 5.   $\sigma$, and $\varphi$: three numerical values at random in the
    interval 0 and 1.
 6.   if $\sigma$ <P %% Applying modified MNOA
 7.     for i = 1:N
 8.       if $\varphi < P_{a_1}$ /* Exploration phase*/
 9.         Generate $\overrightarrow{X}_i^{t+1}$ using (57) and (54)
10.       else /*Exploitation phase*/
11.         Generate $\overrightarrow{X}_i^{t+1}$ using (58) and (54)
12.       end if
13.       $t++$ %% Increment the current function evaluation
14.     end for
15.   Else %% Applying modified MGBO
16.     for each i individual
17.       Select randomly $a \neq b \neq c \neq d \neq i$.
18.       // GSR strategy.
19.       Generate $\overrightarrow{X}_i^{t+1}$ by (56)
20.       $t++$ %% Increment the current function evaluation
21.     End
22.   End
23. end while
```

4.5. Adaptation of HNOA under commonly used encoding scheme

Before starting the optimization process, PS individuals are created, where each individual is compounded of $3m$ dimensions. The first m dimensions represent the mining decision for each miner and are randomly initialized with 0 and 1 to decide whether the miner corresponding to the current initialized dimension takes part in the mining process or not; the following m dimensions represent the transmission power and are randomly initialized within their lower and upper bounds of 0.1 and 2, respectively; the last m dimensions represent the computing resources, which are also initialized at random in the range [0.01, 1] as its search boundaries, as used in (Wang, 2021). After encoding the solutions, they are assessed by (12) to determine their quality, and the solutions with the highest profit, also referred to as objective value, are identified as the best-so-far solution ($\overrightarrow{X^*}$), while that with the lowest objective value is considered the worst solution ($\overrightarrow{X}_w^t$). Afterward, the optimization process of HNOA starts to generate new solutions. Those new solutions are fixed to convert the first m dimensions in each solution into binary values, while the other dimensions that exceed their lower and upper bounds are brought back again. The fixed solutions are assessed using (12), and $\overrightarrow{X^*}$ and $\overrightarrow{X}_w^t$ are updated if there are better and worse solutions, respectively. This optimization process is continued until reaching T_{max}.

4.6. Adaptation of HNOA under recently proposed encoding scheme

Under this scheme, a number of PS individuals equivalent to the number of miners is generated, where each individual includes two dimensions, which represent the transmission power and computing resources, respectively. Those individuals are randomly initialized within the search boundary of each dimension. All individuals represent a solution to the tackled optimization problem, and its objective value is computed using (12). This solution is simultaneously considered the best-so-far solution and the worst solution. Afterwards, the optimization process of HNOA and the other rival optimizers is started to generate a new population in the hope that it can optimize resource allocation accurately.

However, the mining decision has not been considered so far, and hence the efficiency of the mining process might be worsened due to the participation of some unpromising miners. Therefore, in (Wang et al., 2021), the mining decision for the miners is considered by designing three operators: insertion, deletion, and replacement operators. Those operators generate three new populations, which could steadily increase, reduce, or remain unchanged. However, all operators are not applied in the same function evaluation in an attempt to exploit the maximum function evaluation as accurately as possible. According to (Wang et al., 2021), a single operator is selected in each function evaluation based on the random values found in a probability vector **r**. This vector includes three random numbers $[r_1, r_2, r_3]$; each number represents the selection probability of each operator. Regarding the unpromising miners, they are added to a tabu list, namely Tlist; this list is empty in the beginning. In the beginning, all miners participate in the mining process, so a vector $\boldsymbol{m}$ is generated and initialized with the number of participating miners. This vector in addition to the current population are updated based on the previous operators to manage the participating miners. More details for three operators are presented in the following list:

- **Selection operator**: This operator selects a miner at random from those who are not already mining (Existing in **m**) or in Tlist. Then, a new mining decision list $\boldsymbol{m}'$ is generated to include this miner, and then the resource allocation of the *ith* solution in the newly generated population X^{t+1} is set to this miner in a new population known as X'. Note that, before applying any operator, X' is equal to X^t, and $\mathbf{m}'$ is equal to **m**.
- **Deletion operator**: In the beginning, this operator set X^t to X', and **m to m**′. Then, it randomly selects a miner from X' and delete it from X' and $\mathbf{m}'$.
- **Replacement operator**: This operator randomly selects a miner from X' and assign it with the computing resources and transmission power of the *ith* individual in the newly generated population.

The objective value and constraint satisfaction of X' is computed and compared to that of the current population X^t. If the fitness value of X' is greater than that of X^t and its constraints are satisfied, $X^{t+1} = X'$; otherwise, $X^{t+1} = X^t$.

5. Experimental settings

In this paper, to validate the performance of HNOA, it is tested on several instances with several miners ranging between 50 and 600. The MEC server in the used blockchain network was placed in the center of $1000 \times 1000m$ square area where all miners were randomly dispersed. The other parameter settings of this network are listed in Table 2. Broadly speaking, in this study, two different experiments are conducted to investigate the performance of HNOA under two different encoding schemes discussed before. Under the commonly-used encoding schemes, nine metaheuristic algorithms, including generalized normal distribution optimization (GNDO) (Zhang et al., 2020); gradient-based optimizer (GBO) (Ahmadianfar et al., 2020); artificial gorilla troops optimizer (GTO) (Abdollahzadeh et al., 2021); differential evolution (DE) (Wang et al., 2021), prairie dog optimization (PDO) (Ezugwu et al., 2022); pelican optimization algorithm (POA) (Trojovský and Dehghani, 2022), dandelion optimizer (DO) (Zhao et al., 2022), NOA, and Kepler optimization algorithm (KOA) in addition to the proposed HNOA are applied to find the near-optimal solution that could optimize both the mining decision and resource allocation. However, this scheme suffers from the high dimensionality problem because it generates solutions of $3n$, where each one is divided into three parts: the first part, which is of the same length as the used miners, includes either 0 to identify that the miner does not participate in the mining or 1 to show that the miner will participate; the second and third parts includes the computing resources and transmission power of all used miners. Due to the poor performance of the metaheuristics under this encoding scheme, the authors in (Wang et al., 2021) proposed a novel encoding scheme that aids in solving the dimensionality problem in addition to considering the resource allocation for the participating miners only. To confirm that, those two encoding schemes are employed with some recently-published metaheuristic algorithms to find the most effective algorithm for optimizing the resource allocations of all miners.

All algorithms were compared using several performance metrics, including best fitness, average fitness, worst fitness, standard deviation (SD), p-value returned by the Wilcoxon rank sum test, Friedman mean rank (F-rank), convergence curve, and computational cost, to disclose their effectiveness and efficiency. The best fitness, average fitness, worst fitness, and F-rank show the outcomes' accuracy; SD shows the algorithms' stability; the computational cost reports the efficiency; the Wilcoxon rank-sum test is used to measure the difference between the outcomes of HNOA and those of each rival optimizer; and finally; and the convergence curve is used to reveal the convergence speed of each algorithm. Regarding the parameter of the rival optimizers, they are set as recommended in the cited papers (See Table 3). This study's experiments were executed on a computer with MATLAB R2019a, a 2.40 GHz Intel(R) Core(TM) i7-4700MQ processor, 32 GB of RAM, and a 64-bit version of Windows 10 Pro. Due to the fact that all algorithms examined in this study are stochastic, they are executed 30 times independently under a number of function evaluations up to 10,000, and the obtained outcomes are analyzed in terms of the previous performance metrics.

Table 2
The used blockchain network's parameter settings.

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

Table 3
Parameter settings of HNOA and rival optimizers.

Algorithms	Parameters	Value	Algorithms	Parameters	Value
GTO (Abdollahzadeh et al., 2021)	*p*	0.03	**DE** (Wang, 2021)	Scaling factor	0.9
	Beta	3		Crossover rate	0.5
	w	8			
	N'	120			
GBO (Ahmadianfar et al., 2020)	pr	0.5	**GTO** (Abdollahzadeh et al., 2021)	P	0.03
	$\beta_{min}\beta_{max}$N	0.2		Beta	3
		1.2		w	0.8
		50			
PDO (Ezugwu, 2022)	rho	0.005	**DO**(Zhao, 2022)	α	[0, 1]
	eps	0.1		k	[0, 1]
KOA (Abdel-Basset, 2023)	*T, M_0, λ*	3, 0.5, 10	**NOA**	Alpha	0.05
				Pa2	0.2
				Prb	0.2
POA (Trojovský and Dehghani, 2022)	*R*	0.2	**HNOA**	P	0.8

6. Results and discussions

Herein, the findings of the conducted experiments under each encoding scheme are reported to illustrate the outperformance of HNOA over the rival optimizers. The effectiveness of the recently-proposed encoding scheme over the commonly-used one is also discussed in this section.

6.1. Comparison under commonly utilized encoding scheme

This section first investigates the performance of various transfer functions to pick the most effective one that could improve HNOA for efficiently finding the participant miners that could maximize the total profits. HNOA under various transfer functions has been executed 30 independent times, and the average fitness value and F-rank are computed and presented in Table 4. This table discloses the superiority of V4 when the number of miners is greater than 100; otherwise, both S1 and S2 are competitive. To clearly show the effectiveness of each transfer function, the average of the F-rank values presented in the last-stated table is computed and presented in Fig. 7. Inspecting this figure discloses that V4 is the best, followed by S2, while S4 is the worst. Based on that, in the next experiments, V4 is integrated with some recently-published metaheuristic algorithms to investigate their performance against HKOA for jointly optimizing the resource allocation and mining decision when using the commonly-used encoding scheme.

Under the commonly-used encoding scheme, all algorithms with V4 and S2 have been executed 30 independent times using a maximum number of function evaluations and population size

Table 4
Comparison among various transfer functions with HNOA under commonly-used encoding scheme.

m	**Metrics**	**S1**	**S2**	**S3**	**S4**	**V1**	**V2**	**V3**	**V4**
50	Ave	2501.20	**2501.29**	2500.50	2499.65	2467.25	2469.00	2467.25	2464.01
	F-rank	2.40	**2.20**	2.76	3.04	6.28	6.40	6.48	6.80
100	Ave	**4895.85**	4894.20	4887.45	4883.82	4866.83	4862.34	4858.08	4867.68
	F-rank	**2.520**	2.960	3.440	4.120	5.600	5.760	6.080	5.52
150	Ave	6815.81	6950.28	6494.07	6125.52	7065.24	7105.16	7110.12	**7134.11**
	F-rank	5.84	4.76	7.16	7.80	3.32	2.76	2.36	**2.00**
200	Ave	8474.61	8999.52	7913.30	7507.80	9200.41	9260.03	9277.10	**9331.78**
	F-rank	6.12	4.84	7.00	7.84	3.16	2.64	2.40	**2.00**
250	Ave	9921.29	10939.04	9028.20	8206.43	11214.89	11309.43	11264.84	**11354.37**
	F-rank	6.160	4.680	7.040	7.800	3.160	2.240	2.960	**1.96**

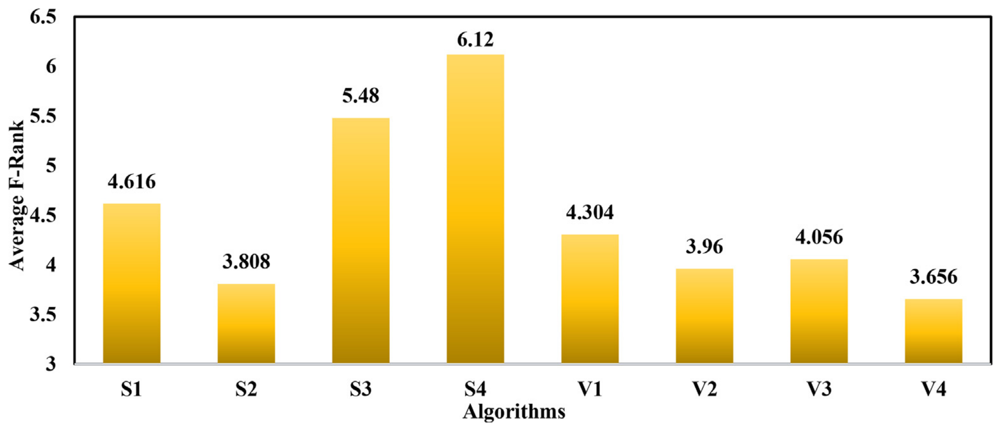

Fig. 7. Average F-rank of HNOA with various transfer functions under commonly-used encoding scheme.

Table 5
Comparison among algorithms with S2 under commonly-used encoding scheme.

m	Metrics	HNOA	NOA	GNDO	DO	GTO	PDO	KOA	GBO	POA	DE
50	Worst	**2498.68**	2476.66	1416.41	1942.51	2456.43	1312.51	2100.86	2358.80	1541.15	1968.21
	Ave	**2501.75**	2493.83	1505.30	2225.58	2491.83	1492.28	2307.94	2472.15	1862.54	2099.39
	Best	2504.43	2499.40	1629.25	2457.91	**2508.59**	1599.13	2470.03	2503.43	2346.76	2185.87
	SD	**1.630**	5.044	54.075	113.618	16.216	65.218	113.352	30.96	230.19	55.62
	p-value		2.E-09	1.E-09	1.E-09	5.E-01	1.E-09	1.E-09	1.E-07	1.E-09	1.E-09
	F-rank	**1.52**	2.84	9.44	5.88	2.20	9.52	5.40	3.52	7.72	6.96
100	Worst	4809.23	4789.92	2610.74	3499.80	**4855.78**	2431.76	2690.23	3899.39	2911.96	2167.36
	Ave	4891.15	4866.51	2716.04	3960.32	**4943.98**	2707.69	3798.94	4683.93	3578.06	2359.91
	Best	4929.00	4910.77	2864.96	4358.65	**4988.73**	2897.76	4802.20	4921.37	4439.05	2701.20
	SD	35.258	**29.397**	69.037	164.963	39.038	103.038	539.891	264.14	402.09	129.07
	p-value		1.E-03	1.E-09	1.E-09	2.E-05	1.E-09	1.E-09	1.E-05	1.E-09	1.E-09
	F-rank	2.28	2.96	8.36	5.64	**1.2**	8.64	5.88	3.68	6.44	9.92
150	Worst	**6782.44**	6410.08	3477.94	2822.51	6381.45	2695.35	3534.79	3860.34	3795.03	2030.40
	Ave	**6950.24**	6637.80	3633.14	3751.19	6947.59	3337.03	4318.74	5383.41	5092.00	2367.78
	Best	7039.46	6812.37	3779.53	4299.34	**7265.34**	3741.12	5890.35	6643.59	6073.59	2761.78
	SD	**64.823**	126.632	76.329	394.538	209.969	262.900	582.142	736.43	688.66	146.60
	p-value		2.E-09	1.E-09	1.E-09	7.E-01	1.E-09	1.E-09	1.E-09	1.E-09	1.E-09
	F-rank	**1.48**	2.96	7.68	7.32	1.60	8.68	5.96	4.36	4.96	10
200	Worst	**8709.30**	7815.87	4500.19	2785.66	8079.35	3064.56	3924.39	1919.07	4424.51	2246.06
	Ave	8982.91	8530.39	4669.73	3968.41	**9073.81**	3664.25	5518.86	5801.89	6139.21	2597.25
	Best	9158.75	8903.12	4910.69	4825.52	**9591.88**	4210.87	8067.33	8404.10	7807.01	3052.13
	SD	**107.143**	246.648	120.283	527.507	366.175	318.840	1107.481	1432.19	970.10	210.46
	p-value		6.E-09	1.E-09	1.E-09	4.E-01	1.E-09	1.E-09	1.E-09	1.E-09	1.E-09
	F-rank	**1.52**	2.88	6.48	8.12	1.6	8.56	5.64	5.44	4.84	9.92
250	Worst	**10433.65**	9156.46	5125.90	3812.35	8760.65	2656.88	4315.33	3539.57	4975.298	2192.621
	Ave	**10838.38**	10193.51	5609.13	4633.32	10727.24	4064.41	6093.01	6904.13	7119.363	2674.308
	Best	11143.14	10787.96	5976.05	5864.63	**12355.57**	4854.68	9686.32	9533.93	9030.397	3451.257
	SD	**214.19**	376.12	182.62	516.46	835.55	486.86	1387.20	1670.28	1125.496	267.0089
	p-value		3.E-08	1.E-09	1.E-09	9.E-01	1.E-09	1.E-09	1.E-09	1.E-09	1.E-09
	F-rank	**1.52**	2.84	6.12	7.88	1.68	8.64	6.08	5.16	5.08	10

Table 6
Comparison among algorithms with V4 under commonly-used encoding scheme.

m	Metrics	HNOA	NOA	GNDO	DO	GTO	PDO	KOA	GBO	POA	DE
50	Worst	**2446.65**	2461.90	1409.36	1217.15	2444.91	1377.53	2242.14	2214.04	1679.25	2178.33
	Ave	2472.67	2468.93	1519.06	1408.66	**2474.03**	1512.13	2339.23	2433.72	1911.53	2287.97
	Best	2497.13	2472.92	1618.48	1647.47	**2506.15**	1643.82	2422.11	2487.37	2318.80	2355.83
	SD	16.404	2.511	55.665	115.613	**14.728**	59.290	54.796	64.73	185.00	48.16
	p-value		8.E-01	1.E-09	1.E-09	5.E-01	1.E-09	1.E-09	1.E-02	1.E-09	1.E-09
	F-rank	2.36	2.68	8.64	9.48	**1.92**	8.88	5.08	3.16	6.96	5.84
100	Worst	**4790.70**	4807.32	2633.18	2188.48	4689.91	2598.58	3435.40	4301.31	3016.05	2598.53
	Ave	4850.84	4852.95	2729.58	2495.68	**4903.82**	2774.70	4020.69	4693.54	3400.73	2770.40
	Best	4897.61	4883.42	2917.56	2733.52	**4988.61**	2926.79	4675.62	4894.46	4048.38	3014.97
	SD	**30.198**	19.277	71.033	133.161	80.932	95.682	300.134	180.84	319.34	120.57
	p-value		7.E-01	1.E-09	1.E-09	2.E-04	1.E-09	1.E-09	3.E-04	1.E-09	1.E-09
	F-rank	2.64	2.56	8.12	9.8	**1.48**	8.04	5.08	3.36	5.88	8.04
150	Worst	**6872.59**	6875.57	3496.75	1718.01	6283.55	1113.84	3848.84	4577.26	3361.74	2246.919
	Ave	7009.34	6965.13	3707.52	2433.99	**7015.27**	3519.04	5110.86	5653.28	4460.64	2609.6
	Best	7204.18	7062.57	3866.74	2999.22	**7346.25**	3894.88	6048.92	6906.74	5673.26	2951.60
	SD	**76.09**	49.82	103.73	331.08	271.50	607.02	663.23	722.04	563.382	164.46
	p-value		3.E-02	1.E-09	1.E-09	4.E-01	1.E-09	1.E-09	2.E-09	1.E-09	1.E-09
	F-rank	1.96	2.32	7.36	9.56	**1.8**	7.72	4.96	4.32	5.72	9.28
200	Worst	**8786.46**	8997.63	4454.95	2372.58	8062.97	983.91	4974.82	5247.69	3955.64	2433.41
	Ave	9161.32	9098.55	4690.69	2820.76	**9265.50**	3782.53	6149.59	6941.23	5717.98	2820.51
	Best	9375.93	9180.08	4912.50	3399.68	**9990.20**	4975.48	7994.66	8692.74	7819.64	3352.6
	SD	**123.86**	47.06	123.81	260.58	398.90	1156.27	764.18	988.49	941.61	205.38
	p-value		4.E-03	1.E-09	1.E-09	3.E-02	1.E-09	1.E-09	1.E-09	1.E-09	1.E-09
	F-rank	1.88	2.52	7.12	9.24	**1.6**	8.16	5	4.48	5.72	9.28
250	Worst	**10861.36**	10858.88	5291.29	2667.84	9763.82	920.39	5609.71	4982.40	4828.12	2394.94
	Ave	11237.03	11095.92	5649.08	3357.35	**11247.44**	3767.19	7510.31	7807.95	6681.43	2870.13
	Best	**11573.14**	11263.71	6044.51	3730.38	12401.26	5751.49	9598.97	10569.17	9421.90	3397.68
	SD	166.81	115.19	148.82	299.83	555.65	1632.03	1084.14	1717.37	1368.37	241.93
	p-value		9.E-04	1.E-09	1.E-09	6.E-01	1.E-09	1.E-09	1.E-09	1.E-09	1.E-09
	F-rank	**1.56**	2.48	6.56	8.76	1.96	8.44	4.8	5.04	5.76	9.64

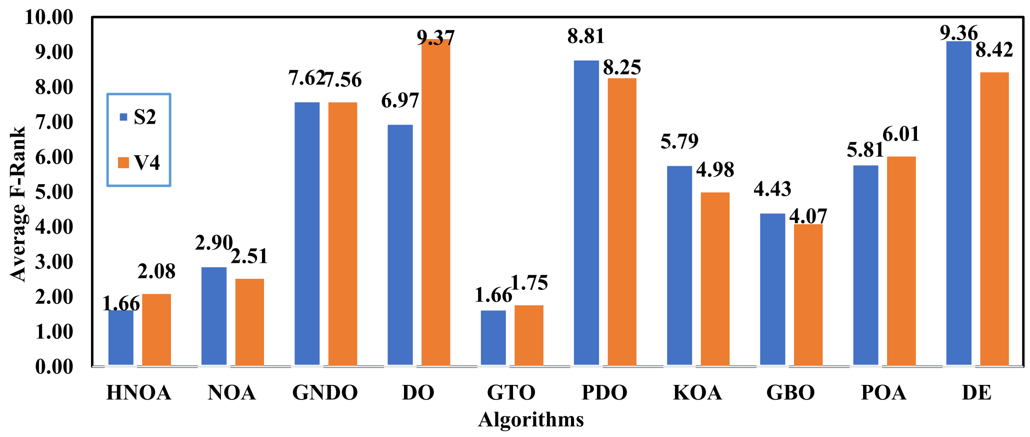

Fig. 8. Average Friedman mean rank of each algorithm under a common encoding scheme.

of 10,000 and 50, respectively. The outcomes of each algorithm under each transfer function within these runs have been analyzed in terms of six performance metrics (worst, ave, best, SD, p-value, and F-rank) and reported in Tables 5 and 6. Inspecting Table 5 discloses that HNOA under S2 has better values for the majority of the performance metrics under various utilized miners (M), while its performance for other metrics is somewhat competitive with the GTO. Under V4, the performance of NOA slightly degrades in comparison to GTO. To reveal the rank of each algorithm, the average of the F-rank values of each algorithm under each transfer is computed and presented in Fig. 8; this figure reveals that GTO and HNOA come in the first rank, followed by NOA, while DE is the worst algorithm. From that, it is concluded that the performance of HNOA relies on the used transfer function, where its performance is competitive with the others under S2 and inferior under V4. Also, the second thing that might deteriorate the performance of HNOA is that each solution includes both binary and continuous values, and hence it must have a high ability to accurately explore the continuous and discrete search spaces to reach the desired solutions. Therefore, this encoding scheme is replaced with a recently proposed one for two reasons:

- To create each solution with continuous values only, and hence the recently published metaheuristic algorithms could be directly adopted for tackling this problem.
- To avoid the high-dimensionality problem that might degrade the performance of several metaheuristic algorithms, like HNOA.

The outcomes under this encoding scheme are discussed in detail in the next section.

6.2. Comparison under recently-proposed encoding scheme

Under the most recent encoding scheme, six algorithms, which have robust performance under the commonly used encoding scheme in addition to the recently-proposed robust algorithm known as DEMiDRA, were executed 30 times independently with a maximum of 10,000 function evaluations and a population size equal to the number of miners. The outcomes of each algorithm derived from these runs were analyzed using seven performance metrics (worst, average, best, SD, p-value, F-rank, and time) and are presented in Table 7. This table reveals that HNOA has superior values for all performance metrics, except for time, under several miners ranging between 150 and 600. To determine the ranking of each algorithm, the average of the F-rank values reported in Table 7 for each algorithm has been computed and displayed in Fig. 9; this figure demonstrates that HNOA is the best algorithm, followed by GBO, DO, and NOA, respectively, while PDO is the worst performing algorithm. Regarding computational cost, Fig. 10 computes and reports the average computational cost incurred by each algorithm across all instances used. This figure reveals that HNOA comes in at the fourth rank in terms of the consumed computational cost, while GNDO is the most consumed algorithm. With the exception of M300, where the proposed algorithm slightly differs from GTO, the p-value from the Wilcoxon rank-sum test which is reported in the last-stated table, indicates that HKOA could produce significantly different results in comparison to all competing algorithms. In a nutshell, HNOA with the recently-proposed encoding scheme could outperform all others for the used miners, and this indicates the effective performance of this encoding scheme with the proposed HNOA.

Finally, we could conclude that the proposed HNOA with the recently-proposed encoding schme is considered a strong alternative for accurately optimizing the mining decision and resource allocation in a MEC-enabled blockchain networks. In addition, this algorithm could be applied to several real-world applications that are widely reliant on blockchain networks, like those in the financial sector, healthcare, the energy sector, cybersecurity, and IoT systems, for efficiently allocating the resources to the participating miners in addition to getting rid of the unpromising miners that might reduce the mining process efficiency.

6.3. Convergence speed analysis

This section depicts the convergence curves of the proposed HNOA in comparison to the other seven optimizers on six instances (M150, M200, M250, M300M, M350, and M400), as shown in Fig. 11. From this figure, it is evident that HNOA's performance is substantially superior to that of all competing algorithms for all used instances at the first half of the optimization process, as it was able to achieve the highest fitness value much more quickly than the competitors. However, at the end of the optimization process, certain algorithms could compete with the proposed algorithm in terms of solution quality. GNDO and PDO have the slowest convergence rates. This demonstrates that HNOA is not only effective for convergence speed but also for final quality.

6.4. HNOA' performance analysis under two encoding schemes

In Fig. 12, we display HNOA's performance under two different encoding schemes to illustrate how far the recently published

Table 7
Comparison among algorithms with recently-used encoding scheme.

m	**Metrics**	**HNOA**	**NOA**	**DEMiDRA**	**GNDO**	**DO**	**GTO**	**GBO**	**PDO**
150	Worst	**7514.77**	7513.54	7466.15	7137.93	7514.50	7501.03	7495.87	5280.73
	Ave	**7516.60**	7516.25	7491.78	7204.98	7516.18	7514.88	7515.77	7017.62
	Best	**7516.70**	7516.66	7505.12	7282.54	7516.59	7516.60	7516.67	7434.26
	SD	**0.381**	0.784	12.367	35.554	0.472	3.683	4.148	525.090
	p-value		**1.62E-07**	**1.42E-09**	**1.42E-09**	**2.87E-08**	**1.84E-08**	**1.65E-06**	**1.42E-09**
	F-rank	**1.24**	3.28	6.00	7.48	4.08	4.08	2.32	7.52
	Time	54.00	52.49	55.62	71.05	55.46	**51.75**	56.07	51.10
200	Worst	**10061.65**	10035.66	9979.28	9058.25	10056.39	9704.45	10060.71	6491.53
	Ave	**10062.97**	10057.68	10005.98	9478.40	10061.59	10042.46	10062.78	9104.32
	Best	**10063.11**	10062.73	10031.55	9631.09	10062.80	10062.73	10063.01	9816.23
	SD	**0.302**	7.137	14.947	134.624	1.377	72.037	0.452	963.622
	p-value		**5.85E-09**	**1.42E-09**	**1.42E-09**	**2.05E-08**	**9.29E-09**	**7.39E-06**	**1.42E-09**
	F-rank	**1.12**	4.16	5.92	7.48	3.68	4.16	1.96	7.52
	Time	58.88	58.34	61.00	80.92	60.54	**56.83**	62.57	56.15
250	Worst	12531.06	12563.22	12458.59	10953.48	12588.01	12428.00	**12593.00**	7713.77
	Ave	**12604.75**	12597.30	12500.64	11572.16	12604.03	12560.03	**12604.36**	11134.91
	Best	**12608.30**	12607.75	12541.23	11897.04	12607.17	12606.04	12607.87	12247.16
	SD	15.36	10.62	24.06	198.33	3.88	54.41	**7.26**	1260.05
	p-value		**1.99E-07**	**1.60E-09**	**1.42E-09**	**8.55E-08**	**1.31E-08**	**2.11E-04**	**1.42E-09**
	F-rank	**1.28**	3.76	5.88	7.48	3.28	4.6	2.2	7.52
	Time	63.46	64.65	67.52	90.32	67.03	**62.41**	68.60	60.42
300	Worst	15004.06	14839.07	14686.18	13029.97	14975.09	5828.62	**15011.28**	9294.91
	Ave	**15021.64**	14983.52	14793.38	13405.58	15009.16	14465.92	15020.61	12563.59
	Best	**15025.45**	15024.31	14889.58	13827.03	15022.53	15022.17	15024.48	14641.30
	SD	5.57	44.81	42.40	201.30	13.73	1820.44	**4.27**	1749.92
	p-value		**1.01E-06**	**1.42E-09**	**1.42E-09**	**5.12E-06**	**3.58E-08**	5.23E-02	**1.42E-09**
	F-rank	**1.64**	3.56	5.68	7.4	3.36	4.88	2	7.48
	Time	79.16	77.57	80.67	117.48	80.58	**74.42**	85.56	72.20
350	Worst	17470.32	17374.92	17066.26	13621.29	17487.01	16281.62	**17474.05**	5448.49
	Ave	**17529.10**	17470.87	17185.46	15089.47	17521.95	17314.57	17523.04	13433.30
	Best	**17540.13**	17538.32	17316.04	16081.80	17536.15	17530.37	17538.03	16412.04
	SD	15.88	51.97	69.51	639.00	13.88	311.54	**15.75**	2754.48
	p-value		**3.21E-06**	**1.42E-09**	**1.42E-09**	**3.84E-03**	**4.00E-08**	**2.20E-02**	**1.42E-09**
	F-rank	**1.68**	3.72	5.88	7.4	2.68	4.44	2.6	7.6
	Time	81.35	79.23	81.90	120.51	81.33	76.13	86.19	74.82
400	Worst	**19911.87**	19503.80	19336.25	15136.25	19761.07	17349.70	19780.18	6014.04
	Ave	**19977.88**	19778.00	19496.61	16514.96	19927.17	19347.64	19948.87	14970.00
	Best	**19995.20**	19958.93	19673.30	17413.09	19982.77	19917.79	19989.77	18651.98
	SD	**22.16**	128.23	92.55	625.36	62.11	693.16	46.25	3049.68
	p-value		**5.85E-09**	**1.42E-09**	**1.42E-09**	**2.55E-05**	**1.80E-09**	**6.38E-04**	**1.42E-09**
	F-rank	**1.44**	4.16	5.48	7.36	2.68	5.04	2.24	7.60
	Time	87.80	88.59	91.53	128.61	86.67	**81.61**	91.74	80.87
450	Worst	**19884.03**	19341.12	19105.91	17110.43	19931.37	17405.59	19812.35	8315.61
	Ave	**20280.79**	19940.48	19804.90	18242.98	20182.95	20011.10	20173.51	16432.80
	Best	**20393.97**	20292.36	20205.42	19294.54	20361.14	20338.12	20331.35	19106.86
	SD	**114.96**	248.86	259.78	662.20	126.69	591.38	137.36	3048.88
	p-value		**1.99E-07**	**1.04E-08**	**1.42E-09**	**1.04E-03**	**1.13E-04**	**1.95E-04**	**1.42E-09**
	F-rank	**1.76**	4.48	5.16	7.36	2.92	3.44	3.28	7.60
	Time	85.11	85.47	87.03	130.98	87.64	82.67	92.25	80.67
500	Worst	**20093.99**	19511.51	19443.26	17743.75	19936.71	12374.30	19027.26	6306.08
	Ave	**20311.78**	20065.38	19835.76	18886.11	20215.45	19770.48	20142.67	14292.61
	Best	**20423.21**	20332.41	20165.14	19749.81	20372.32	20350.36	20384.57	19098.36
	SD	**85.16**	218.19	166.71	393.60	119.91	1560.77	279.15	4522.12
	p-value		**1.23E-06**	**2.03E-09**	**1.42E-09**	**3.84E-04**	**9.70E-06**	**6.38E-04**	**1.42E-09**
	F-rank	**1.76**	3.92	5.48	7.00	2.84	3.92	3.16	7.92
	Time	87.51	86.77	90.11	139.88	88.89	**85.64**	93.77	84.38
600	Worst	**19923.95**	19145.87	19409.92	18506.78	19892.74	19260.51	19622.92	8081.86
	Ave	**20277.39**	19870.09	19792.44	19045.76	20194.16	20079.94	20081.64	16322.62
	Best	**20400.19**	20370.52	20160.94	19538.71	20426.61	20353.03	20352.78	19401.14
	SD	**104.18**	325.45	213.23	294.00	148.38	322.04	224.23	3260.90
	p-value		**3.88E-06**	**4.64E-09**	**1.42E-09**	**3.44E-02**	**4.34E-03**	**5.14E-04**	**1.42E-09**
	F-rank	**1.96**	4.4	5.24	7.04	2.76	3.32	3.4	7.88
	Time	98.97	99.21	104.45	161.34	101.97	**100.27**	112.08	94.94

scheme could improve HNOA. From this figure, it is obvious that the performance of HNOA is almost the same under the two studied encoding schemes when *M* is less than or equal to 200. Higher than that, the performance of HNOA with the recently-proposed scheme is dramatically improved, while its performance with the other scheme has substantially deteriorated. As a result, the proposed HNOA under the different encoding schemes almost has the same performance with the small dimensions However, when

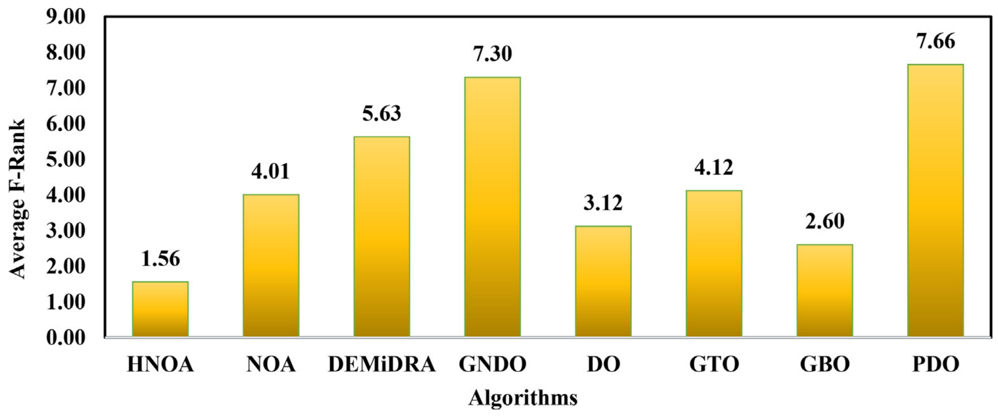

Fig. 9. Average Friedman mean rank.

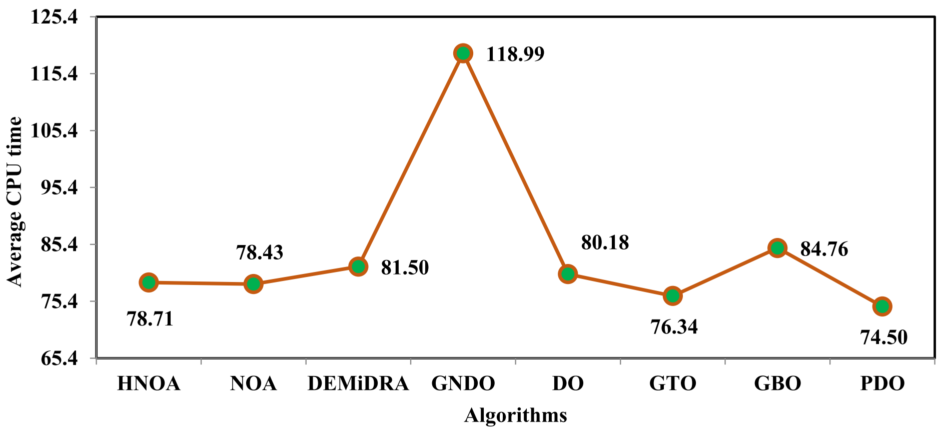

Fig. 10. Average Computational cost.

increasing the number of dimensions, the HNOA with the recently-proposed scheme is significantly improved, with an improvement percentage ranging between 2% and 10%.

6.5. Effects of various improvements

In this section, the influence of various algorithms used to design the proposed algorithm is investigated to disclose their importance. These algorithms include MGBO and MNOA, which are effectively integrated to present the proposed HNOA. In addition, the classical GBO and NOA are added to the comparison to show the difference between GBO and MGBO and between MNOA and NOA experimentally. The average total profits obtained by each algorithm under various miners ranging between 150 and 600 are presented in Fig. 13. This figure reveals that MGBO is better than GBO, with an improvement percentage ranging between 0.1 and 2, and HNOA is better than NOA, with an improvement rate reaching 1.5%, while HNOA is better than all the optimizers for the majority of utilized miners.

6.6. Parameter settings

The proposed HNOA has only one effective parameter that has to be accurately estimated to maximize its performance; this parameter is abbreviated as *P* and employed to determine the tradeoff probability between MGBO and HNOA. To estimate the near-optimal value for this parameter, extensive experiments under various values, including 0.01, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, and 0.9, are conducted to determine under any value the performance of HNOA is maximized. These experiments' outcomes are reported in Fig. 14, which shows that HNOA performs better when P = 0.8.

7. Conclusion and future work

Offloading mining tasks to a mobile edge computing (MEC) server has been used to increase the processing capacity of IoTDs. However, if many miners are trying to offload work to the MEC server at the same time, there may be significant transmission delays due to interference. To effectively solve the transmission delay problem in the MEC server, it is necessary to solve the combined resource allocation and mining decisions of all miners in a MEC-enabled blockchain network, which is an NP-hard optimization problem. In this study, we first adapt several recently published metaheuristic algorithms to test how well they perform when applied to this problem using two distinct encoding schemes. In the first encoding scheme, each solution is encoded in a manner that takes into account the mining decisions, transmission power, and computational resources of all miners. In contrast, the second scheme holds each individual responsible for the transmission power and computational resources of a participant miner, with the result that the whole population is treated as a single solution

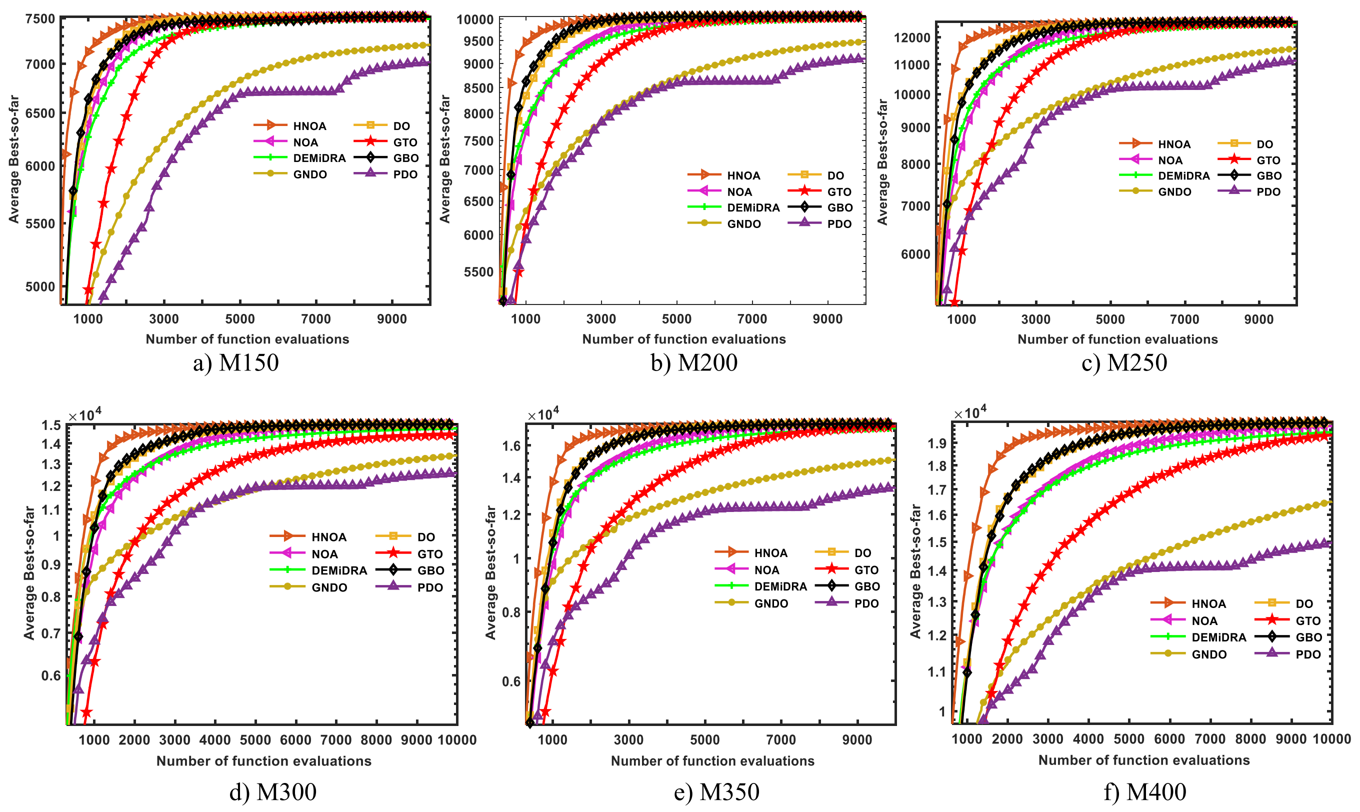

Fig. 11. Comparison among algorithms for convergence speed.

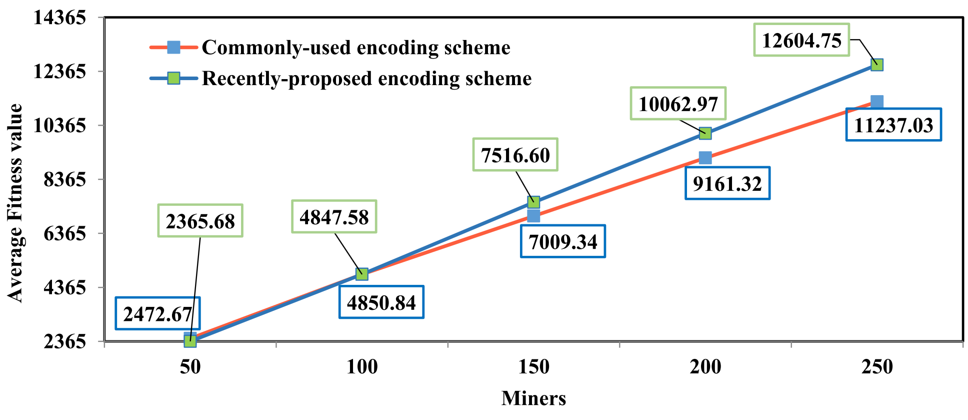

Fig. 12. HNOA's performance under different encoding schemes.

to the problem at hand. The first encoding technique deals with the mining decisions of all miners by a binary vector whose values indicate whether or not the miners participate in mining. In order to make the solutions of the metaheuristic algorithms applicable to this problem under the first encoding scheme, we used various V-shaped and S-shaped transfer functions. Those encoding schemes are employed with two robust modified variants of NOA and GBO, namely MNOA and MGBO, which are herein proposed for optimizing mining decisions and resource allocation in a blockchain network. Furthermore, both MNOA and MGBO are efficiently combined to present a new variant, termed HNOA, with better balancing between exploration and exploitation operators to aid in avoiding getting stuck in local minima and accelerating the convergence speed. Validation of the proposed HNOA using nine instances with 150–600 miners is performed under two encoding techniques. In addition, it is compared against a number of different optimizers so that its effectiveness in terms of a variety of performance measures can be demonstrated. According to the findings of the experiments, the performance of HNOA using the second encoding scheme is noticeably superior to both its performance using the first encoding scheme and the performance of competing optimizers. The main limitation of HNOA is that it consumes a slightly higher computation cost than some compared algorithms.

Our future work is presented in the following list:

- We will investigate the performance of HNOA when using heterogeneous miners in the used blockchain network.

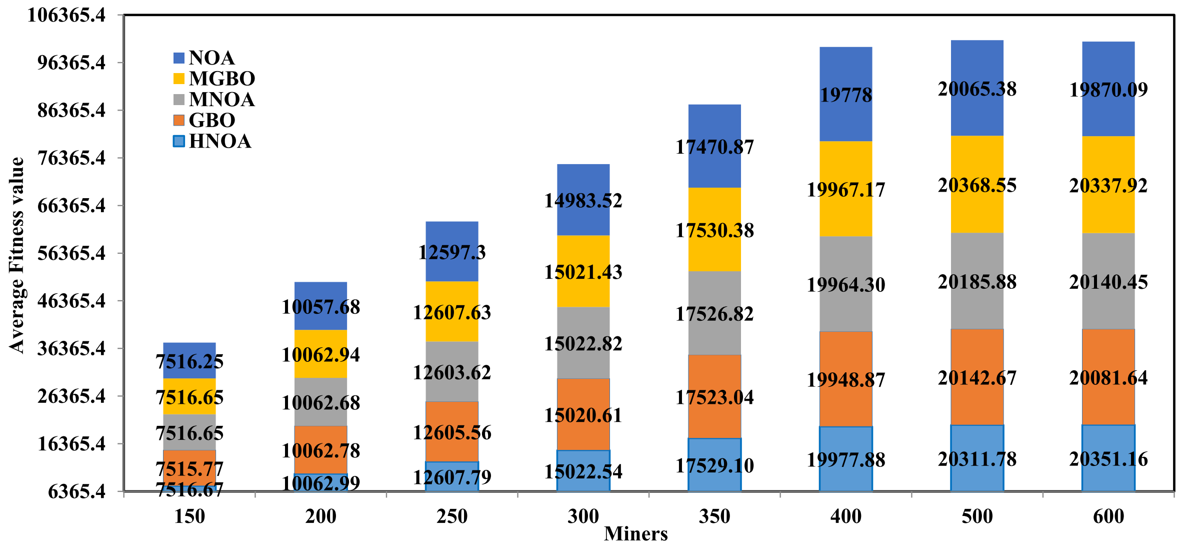

Fig. 13. HNOA's performance under different encoding schemes.

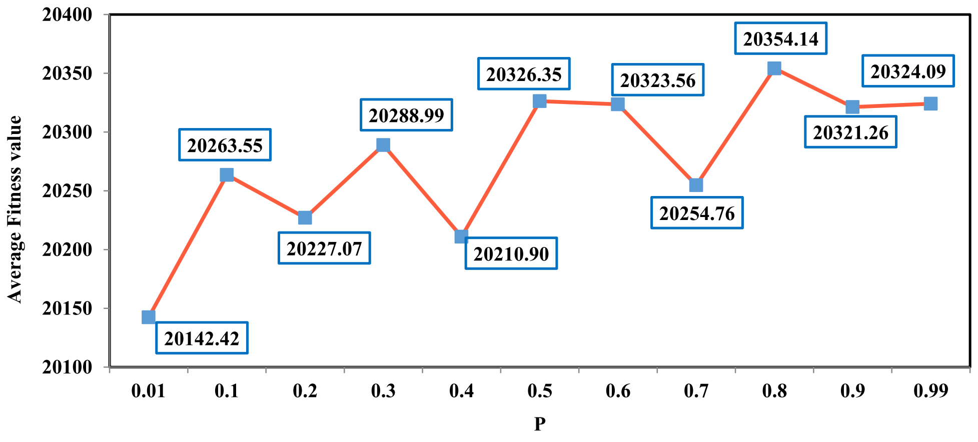

Fig. 14. Tuning the parameter *P*.

- Investigating the performance of HNOA for solving several other optimization problems, like task scheduling challenge in fog and cloud computing, multilevel thresholding image segmentation problem, and several else
- Assessing the performance of some recently proposed binary optimization algorithms for jointly optimizing the mining decisions of all miners in the studied blockchain network.
- Observing the performance of integrating HNOA with some mechanisms like Levy flight, chaotic maps, quantum computing, and ranking-based update strategy.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Funding

This research is supported by the Researchers Supporting Project number (RSP2023R389), King Saud University, Riyadh, Saudi Arabia.

References

Abdel-Basset, M. et al., 2023. Kepler optimization algorithm: A new metaheuristic algorithm inspired by Kepler's laws of planetary motion. Knowl.-Based Syst. 268, 110454.

Abdel-Basset, M., et al., 2023. Nutcracker optimizer: A novel nature-inspired metaheuristic algorithm for global optimization and engineering design problems. p. 110248.

Abdelsalam, M., Idrees, A.M., and Shokry, M. 2023. A Proposed Model for Improving the Reliability of Online Exam Results Using Blockchain. IEEE Access.

Abdollahzadeh, B., Soleimanian Gharehchopogh, F., Mirjalili, S., 2021. Artificial gorilla troops optimizer: A new nature-inspired metaheuristic algorithm for global optimization problems. Int. J. Intell. Syst. 36 (10), 5887–5958.

Agushaka, J.O., Ezugwu, A.E., Abualigah, L., 2022. Dwarf mongoose optimization algorithm. Comput. Methods Appl. Mech. Eng. 391, 114570.

Agushaka, J.O., Ezugwu, A.E., Abualigah, L., 2023. Gazelle optimization algorithm: a novel nature-inspired metaheuristic optimizer. Neural Comput. & Applic. 35 (5), 4099–4131.

Ahmadianfar, I., Bozorg-Haddad, O., Chu, X., 2020. Gradient-based optimizer: A new metaheuristic optimization algorithm. Inf. Sci. 540, 131–159.

Ali, M.S. et al., 2018. Applications of blockchains in the Internet of Things: A comprehensive survey. IEEE Commun. Surv. Tutorials 21 (2), 1676–1717.

Du, J. et al., 2021. Resource pricing and allocation in MEC enabled blockchain systems: An A3C deep reinforcement learning approach. IEEE Trans. Network Sci. Eng. 9 (1), 33–44.

Ezugwu, A.E. et al., 2022. Prairie dog optimization algorithm. Neural Comput. & Applic. 34 (22), 20017–20065.

Hu, G. et al., 2023. DETDO: An adaptive hybrid dandelion optimizer for engineering optimization. Adv. Eng. Inf. 57, 102004.

Hussien, R.M. et al., 2023. An improved Henry gas optimization algorithm for joint mining decision and resource allocation in a MEC-enabled blockchain networks. Neural Comput. & Applic., 1–16

Jiang, F. et al., 2019. Deep-learning-based joint resource scheduling algorithms for hybrid MEC networks. IEEE Internet Things J. 7 (7), 6252–6265.

Laith, A. et al., 2023. Modified Elite Opposition-Based Artificial Hummingbird Algorithm for Designing FOPID Controlled Cruise Control System. Intelligent Automation & Soft Computing.

Liu, M. et al., 2018. Computation offloading and content caching in wireless blockchain networks with mobile edge computing. IEEE Trans. Veh. Technol. 67 (11), 11008–11021.

Luong, N.C., et al. Optimal auction for edge computing resource management in mobile blockchain networks: A deep learning approach. IEEE.

Malik, A., Sharma, A.K., 2023. Blockchain-based digital chain of custody multimedia evidence preservation framework for internet-of-things. Journal of Information Security and Applications 77, 103579.

Nakamoto, S., 2008. Bitcoin: A peer-to-peer electronic cash system. Decentralized business review, 21260.

Tian, Z. et al., 2019. Block-DEF: A secure digital evidence framework using blockchain. Inf. Sci. 491, 151–165.

Trojovský, P. and Dehghani, M.J.S. 2022. Pelican optimization algorithm: A novel nature-inspired algorithm for engineering applications. 22(3), 855.

Wang, K. et al., 2019. Unified offloading decision making and resource allocation in ME-RAN. IEEE Trans. Veh. Technol. 68 (8), 8159–8172.

Wang, W. et al., 2019. A survey on consensus mechanisms and mining strategy management in blockchain networks. IEEE Access 7, 22328–22370.

Wang, Y. et al., 2021. A new differential evolution algorithm for joint mining decision and resource allocation in a MEC-enabled wireless blockchain network. Comput. Ind. Eng. 155, 107186.

Xiong, Z. et al., 2018. Cloud/fog computing resource management and pricing for blockchain networks. IEEE Internet Things J. 6 (3), 4585–4600.

Xiong, Z. et al., 2018. When mobile blockchain meets edge computing. IEEE Commun. Mag. 56 (8), 33–39.

Yuan, S., Li, J., Wu, C., 2022. JORA: Blockchain-based efficient joint computing offloading and resource allocation for edge video streaming systems. J. Syst. Archit. 133, 102740.

Yuan, Y., Wang, F.-Y., 2018. Blockchain and cryptocurrencies: Model, techniques, and applications. IEEE Transactions on Systems, Man, and Cybernetics: Systems 48 (9), 1421–1428.

Zhang, Y., Jin, Z., Mirjalili, S., 2020. Generalized normal distribution optimization and its applications in parameter extraction of photovoltaic models. Energ. Conver. Manage. 224, 113301.

Zhao, S., et al. 2022. Dandelion Optimizer: A nature-inspired metaheuristic algorithm for engineering applications. 114, p. 105075.