Received 26 March 2023, accepted 26 April 2023, date of publication 3 May 2023, date of current version 25 May 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3272732

RESEARCH ARTICLE

Evolutionary Game Theory-Based Optimal Scheduling Strategy for Heterogeneous Computing

RUI SHE [1] **AND WEI ZHAO** [2], **(Member, IEEE)**

[1]Research Institute of China Telecom Company Ltd., Beijing 102209, China
[2]North China Electric Power University, Baoding 071003, China

Corresponding author: Rui She (sher@chinatelecom.cn)

This work was supported in part by the National Key Research and Development Plan Project: New Data Stream Heterogeneous Processor Architecture and Computing System under Grant 2022YFB4501405.

ABSTRACT With the development of intelligent applications, simply relying on traditional single type of computing unit cannot efficiently satisfy diverse cloud requirements. The emergence of heterogeneous computing can efficiently achieve the adaptation of these intelligent applications by using different types of processing units such as Graphics Processing Unit (GPU) and Field Programmable Gate Array (FPGA). However, the trade-off between profit and costs in the process of scheduling heterogeneous computing resources is also an issue worthy of attention. To address this challenge, this work establishes a heterogeneous computing resource scheduling model based on Stackelberg differential game, which includes three roles Computing Power Trading Platforms (CPTPs), Heterogeneous Computing Service Providers (HCSPs), and Heterogeneous Computing Application Providers (HCAPs). The objective is to maximize utility function of CPTPs and HCSPs subject to rental ratio, pricing strategy and energy consumption of resource scheduling, which has proved that there exists a Stackelberg Nash Equilibrium (NE) solution. The Support Vector Machine based on Artificial Fish (SVM-AF) is proposed to predict the access times of heterogeneous computing applications. In addition, the distributed iteration method and Cauchy distribution is adopted to optimize the computing price strategy and improve its convergence performance. The simulation results show that compared with other strategies, the proposed strategy can effectively improve computing revenue of user experience and while reducing energy consumption in the process of resource scheduling.

INDEX TERMS Heterogeneous computing, resource scheduling, game optimization, Stackelberg.

I. INTRODUCTION

In recent years, with the continuous development of demand for new cloud computing services such as AI and big data, especially the extensive application of deep learning, higher requirements have been put forward for the computing ability of servers deployed in the cloud [1], [2], [3]. Affected by the physical design limit and energy consumption control, the traditional single computing unit with CPU can no longer meet the growing demand for diverse cloud requirements [4], [5]. As the demand for computing power in various fields continues to increase, various hardware chip products have been launched for different computing scenario applications, such as GPUs for image processing [6], [7], FPGAs for high-performance computing [8], NPUs for neural network based training [9], and DPUs for data processing [10].

Nowadays, heterogeneous multi-core computing architecture has gradually become the mainstream processor, which base on different types of instruction set architecture and computing units form a new system enables to serve the most suitable business scenarios [11], [12], [13]. At present, it mainly includes GPU, FPGA, NPU and elastic accelerated computing [14], [15]. With these computing units equip with special capabilities showing more and more powerful performance in large-scale parallel computing. OpenCL by Apple [16] and CUDA by NVIDIA [17] are both heterogeneous

The associate editor coordinating the review of this manuscript and approving it for publication was Agostino Forestiero.

computing unit. In this structure, the CPU is responsible for making up for the lack of logic control capabilities of the GPU. The GPU compensates for the shortcomings of the traditional CPU multicore architecture when dealing with high parallel, large scale compute intensive applications. Make the whole system suitable for a wider range of application scenarios. However, this structure also faces many new challenges. For example, the difference in heterogeneous cores leads to a decrease in resource utilization, which makes the system performance less than expected. Therefore, improving re-source utilization in heterogeneous systems is an extremely important research objective in the field of heterogeneous multicore research.

A. RELATED WORKS

In recent years, various heterogeneous computing resources scheduling schemes to enhance system performance have been proposed by academia and industry [20], [21]. In order to minimize the energy consumption in the task processing of heterogeneous edge devices, which including CPU and GPU computing resources, Zeng et al. [22] propose a resource management method of edge nodes based on federated learning. Based on CPU-GPU heterogeneous computing, the authors in [23] propose an energy efficient resource management strategy for federated edge learning to promote cooperative cooperation between CPU, GPU, and other heterogeneous computing hard-ware resources to improve the utilization of resources. Wang et al. [24] propose a dynamic scheduling scheme for real-time tasks based on cloud data center virtualization, which classifies heterogeneous tasks and virtual machines based on historical scheduling records and merges them by scheduling jobs of similar classes to maximize the host's operational state utilization. To further improve the utilization of heterogeneous resources to reduce the cost overhead in the scheduling process, Zeng et al. [22] propose a dynamically reconfigurable task scheduling method based on the CPU-FPGA heterogeneous body system architecture to improve the scheduling efficiency by fully considering the task scheduling overhead and latency impact. In [25], a heuristic algorithm for CPU-GPU system utilization awareness and energy saving is proposed from the accuracy of computing scheduling.

However, the above heterogeneous computing resource scheduling schemes are all for static scenarios. In real scenarios, diverse computing tasks are constantly coming. Accurately predicting the upcoming heterogeneous resources and preparing in advance is also the key to improve the utilization of heterogeneous resources and increase the revenue of computing service providers. In [26], a recurrent neural network based traffic prediction model for piecewise access control is proposed to further improve the network resource utilization by constructing an improved variant with a closed loop parameter update mechanism. Reference [27] proposed an AlloX strategy for achieving efficient prediction of access-side machine learning business resources, thereby enabling rational utilization of GPU and CPU resources and reducing the cost of CPU/GPU data centers. The authors of [28] study an elastic multi-resource allocation strategy based on a coupled CPU-GPU shelf, which provides resource availability while better ensuring user fairness. In [29], a deep Q-learning resource prediction and scheduling algorithm for GPU is proposed, which designed three prototypes of resource management systems, the simulation results show significant improvements in resource utilization compared to ordinary heuristics. The authors in [30] proposed a deep learning based on multi-core CPU workload prediction by fusing GMM clustering with LSTM algorithm for phase prediction, which will produce the best phase-aware prediction results and reduces the average error.

Although the above studies through task prediction have optimized resource utilization, but not consider the revenue issue. It is also worthwhile for cloud computing resource operators to pay attention to improving the operational revenue of computing power providers. Through reasonably adjusting the resource allocation strategies while ensuring the computing power service demand of computing power application providers. Game theory has been widely used in resource allocation and optimization tasks, such as internet pricing and network slicing resource allocation. In [31], a cloud computing resource sharing mechanism based on Stackelberg's differential game is proposed to facilitate resource transactions between cloud computing service providers and different edge computing service providers. The authors in [32] investigate a spatial anti-interference scheme for IoT that minimizes the anti-interference routing cost through Stackelberg games and reinforcement learning. For the limited computing resources of MEC servers, [33] designed a reasonable resource pricing and task offloading strategy based on the Stackelberg game. The simulation results showed that the study could improve the profit of MEC servers and the utility of end users, thus achieving a win-win situation. Reference [34] proposes a non-cooperative Stackelberg game interaction algorithm for distributed scheduling of fog and cloud resources. A resource controller is initiated to manage the available fog resources to further maximize the service provider's profit and seamless resource provisioning.

B. MOTIVATION AND CONTRIBUTIONS

The heterogeneous computing resource scheduling has been studied in [22], [23], [24], and [25], but it mainly considers the static resource scheduling scenario, the temporary resource switching scenario is ignored. Meanwhile, although resource prediction has effectively improved the flexible scheduling ability of computing resources in [26], [27], [28], [29], and [30], but not consider actual costs and profit from the perspective of operators and how to maximize the benefits of computing operations. Besides, the current research on resource scheduling based on game theory is mainly oriented to cloud computing power pricing, network elements, and other fields [31], [32], [33], and [34]. To the best of our knowledge, there is no contribution to investigate the heterogeneous computing resource scheduling with Stackelberg. Furthermore, the resource competition behavior between computing resource service providers in a heterogeneous computing environment is similar to the free competition market in economics, and the game theory based on the method can build the competition relationship in resource management. Therefore, to obtain more computing benefits by competing for limited computing resources, the game theory is introduced to build the cooperation and competition relationship between computing service providers. Main contributions of this paper are summarized as follows.

The main contributions of this paper are summarized as follows:

- We establish the heterogeneous computing resource scheduling model based on distributed computing resource management, which includes three roles, Computing Power Trading Platforms (CPTPs), Heterogeneous Computing Service Providers (HCSPs), and Heterogeneous Computing Application Providers (HCAPs).
- A Stackelberg game is proposed to facilitate the computing resource trading between the CPTPs and HCSPs. The profit function of heterogeneous computing trading platforms is constructed based on income, user preferences, and energy consumption, which has proved that the Stackelberg Equilibrium (SE) is exit in the proposed game.
- An Support Vector Machine based on Artificial Fish (SVM-AF) is developed to predict the access times of heterogeneous computing applications.
- The distributed iteration method and Cauchy distribution are adopted to optimize the computing price strategy and improve its convergence performance.

The effectiveness of the scheme has been verified, simulation results show that the proposed strategy can effectively improve computing revenue of user experience and while reducing energy consumption in the process of resource scheduling.

C. ORGANIZATION

The remainder of this paper is organized as follows. The system model is presented in Section II. We formulate the optimization problem in detail and describe its solutions in Section III and Section IV, respectively. Simulation results are presented in Section V. Finally, we draw our conclusions in Section VI.

II. SYSTEM MODEL

The system model of heterogeneous computing power resource scheduling is shown in Figure 1. The system model mainly contains X CPTPs $N_{CPT}(i = 1, 2, \ldots, X)$ and Y HCSPs $N_{CSP}(j = 1, 2, \ldots, Y)$. HCSP develop hardware resources based on the business requirements of different dedicated capabilities and provide them to the upper layer of a trusted arithmetic trading platform to realize hardware resource virtualization and abstraction, which can realize the unified management of resources. As the user of computing resources, HCAPs submits computing requirements to the resource trading platform to obtain more suitable computing resources, such as AI model training, video rendering, security surveillance, and graph computing.

The computational resources obey a heterogeneous chi-square Poisson point distribution with density φ. The resource scheduling process for the heterogeneous computing power is as follows:

1) HCSPs gives the price of computing resources and informs the CPTPs.

2) Based on the performance and price of computing power, CPTPs determines the computing power rental proportion and issues computing power prices according to the request of HCAPs.

3) Based on the prices, preferences and satisfaction, HCAPs select appropriate heterogeneous computing resources to access.

A. COMPUTING APPLICATION POPULARITY

In reality, HCAP has different preferences depending on the requirements, so the popularity of other computing power varies is important. The higher the application popularity, the more frequently users visit, so as to lead more revenue. Therefore, introducing the application popularity metric can more reasonably determine the proportion of application computing power rented and improve the revenue of the computing power resource platform.

Assume that the applications are denoted by $A = A_1, A_1, \ldots, A_N$, the popularity of the n-th application is $D_n, n = 1, 2, \ldots, N$. According to the popularity of the applications, the set of applications for algorithmic resources is ranked in descending order. The probability of a user requesting service from an application is D_n, which obeys the Zipf distribution.

With the increasing popularity of AI model training, video rendering, security surveillance, and graph computation, some popular applications are becoming well-known. In this paper, regulators θ is introduced to adjust the popularity of arithmetic applications to improve the popularity of heterogeneous arithmetic and the gain of heterogeneous computing power networks. The calculation formula can be given by

$$D_n = \frac{\theta / n^a}{\sum\limits_{i=1}^{N} (1/i^a)}, \tag{1}$$

where, a is the popularity index factor for heterogeneous computing power, with the increase of a, the popularity of heterogeneous computing applications continues to increase.

B. USER PREFERENCES

The variability of the computing power services provided by varies HCAPs leads to different preferences. The preference of computing power users directly affects the revenue of HCAP. In general, if the user preference is low, the revenue of HCAP will be lower, and vice versa. Therefore, in the paper,

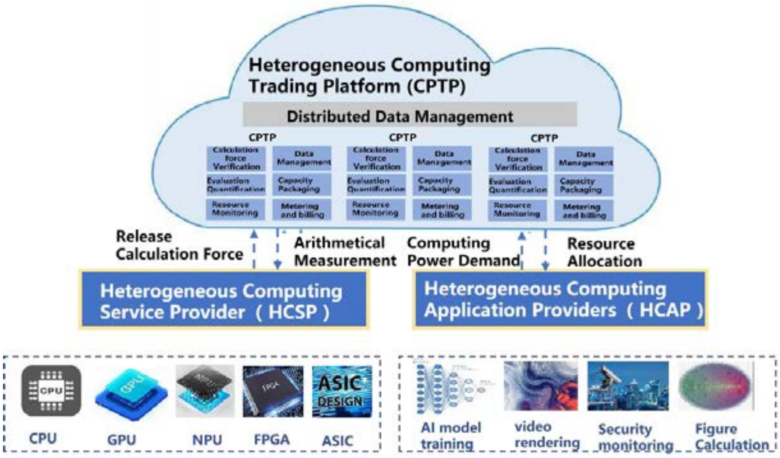

FIGURE 1. The heterogeneous computing resource scheduling model.

user preference metrics are introduced to measure the quality of computing services.

Various factors influence user preference, such as computing power performance, latency, energy consumption, etc. It assumed that each computing power has M evaluation metrics, and each evaluation metric has R discrete values. Each computing power resource corresponds to an computing power preference matrix $\mathbf{G}$, which can be given by

$$\mathbf{G} = \begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1R} \\ g_{21} & g_{22} & \cdots & g_{2R}, \\ \vdots & \vdots & \vdots & \vdots \\ g_{M1} & g_{M2} & \vdots & g_{MR} \end{bmatrix} \tag{2}$$

where g_{mr} denotes the degree of preference of HCAP for the m-th characteristic, m represents arithmetic aversion, and r is the computing preference $m \in 1, 2, \ldots, M, r \in 1, 2, \ldots, R$. $\mathbf{W} = W_1, W_2, \ldots, W_M$ is the matrix of weight values for M evaluation indicators, and the preference of users for the D-th computing power as shown in equation (3)

$$R_n = \frac{\lambda}{\lambda + \varepsilon} \frac{1}{R} \sum_{r=1}^{R} (\mathbf{WG}), \tag{3}$$

where ε is the influence factor of user preference. λ is the percentage of computing power resources rented by the CPTPs. R denotes the number of evaluation indicators. According to the law of large numbers, B_n converges to a constant value when it tends to infinity

$$\lim_{R \to \propto} B_n = \lim_{R \to \propto} \frac{\lambda}{\lambda + \varepsilon} \frac{1}{R} \sum_{r=1}^{R} (\mathbf{WG}) = \frac{\lambda}{\lambda + \varepsilon}. \tag{4}$$

C. CPTPs ENERGY CONSUMPTIONS

The energy consumption of CPTPs as the sum of energy consumption for resource scheduling E_r, maintenance E_m and idle time E_{idle}, the total energy consumption of the CPTP as

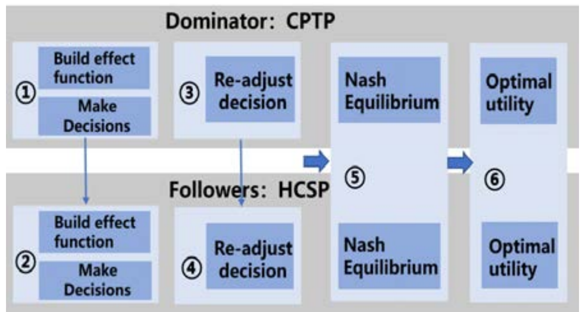

FIGURE 2. The process of the stackelberg game.

shown in equation (5)

$$\begin{aligned} E_x &= \sum_{y=1}^{Y} \lambda_{xy} E_r + E_m + E_{idle} \\ &= \sum_{y=1}^{Y} \lambda_{xy} t_r e_r + t_m e_m, \\ &\quad + (T_{total} - t_r - t_m) e_{idle} \end{aligned} \tag{5}$$

where λ_{xy} is the proportion of x-th CPTP renting y-th HCSP computing power, respectively. e_r, e_m and e_{idle} are the energy consumption per unit of time for resource scheduling, maintenance and idle time,. t_r, t_m are the time of resource scheduling and maintenance, respectively. T_{tot} is the total time.

This section proposes the Stackelberg to optimize the scheduling for the heterogeneous computing resources. The Stackelberg game model consists of three roles: parties, the strategy space, and utility function. The game parties generally consist of a dominant player and a follower. Both sides of the game have their utility functions and strategy spaces. In this paper, the process of the game can be shown as Fig. 2.

The Stackelberg game model contains three elements, which formulate as

1) Game player: CPTPs as the leader with a number of X. Followers are HCSPs with a number of Y.

2) Strategy space: Strategy space of the leader is the rent ratio λ_{xy} of computing resources given by the CPTPs. Follower's strategy space is the price $P = P_1, P_2, \ldots, P_Y$ of computing power service set by HCSPs.

3) Utility function: The utility functions of the leader and follower are the utility of CPTPs and HCSPs, respectively, given by U_x, U_y.

The utility function of the CPTP can be obtained can be expressed as

$$U_x^{proof} = \sum_{i=1}^{Y}\sum_{n=1}^{N} TSD_n B_n, \tag{6}$$

$$U_x^{rent} = \sum_{i=1}^{Y} \lambda_{xy}\varphi_y P_y,$$

$$U_x = U_x^{proof} - U_x^{rent} - \tau E_x \tag{7}$$

$$= \sum_{y=1}^{Y}\sum_{n=1}^{N} TSD_n B_n - \sum_{y=1}^{Y} \lambda_{xy}\varphi_y P_y,$$

$$- \sum_{y=1}^{Y} \tau(\lambda_{xy}E_r + E_m + E_{idle}) \tag{8}$$

where λ_{xy} is the proportion of x-th CPTP renting y-th HCSP computing power. U_x^{proof} is the proof from providing services for HCAP, φ_y is the total computing power of P_y. U_x^{rent}, E_x are the cost of computing resources rent and energy efficiency consumption, respectively. T denotes the number of times of the application is accessed per unit of time, τ denotes cost per unit energy, S is the unit price per service. The utility functions of HCSPs are the subtracting value of the fee charged for renting the computing resource service and the cost of managing the computing resource, which can be written as

$$U_y = \sum_{i=1}^{X} (P_y - C_y)\varphi_y \lambda_{xy}. \tag{9}$$

III. PROBLEM FORMULATE

A. PROBLEM FORMULATE

The optimization problem of the HCSPs is to maximize the utility function, which can be given by

$$MaxU_y(P, \lambda). \tag{10}$$

The optimization problem of CPTPs can also be denoted as maximizing the utility function, which can be formulated as

$$MaxU_x(T, P, \lambda). \tag{11}$$

Consequently, our goal is to maximize the utility function of CPTPs and HCSPs by optimizing the heterogeneous computing applications access times T, rental ratio of computing λ_{xy} and the pricing strategy of computing resources P_y, Which can be formulated as

$$\max_{T, P_y, \lambda_{xy}} \{U_x(T, P, \lambda), U_y(P, \lambda)\} \tag{12}$$

$$s.t.\ E_x = \sum_{y=1}^{Y} \tau(\lambda_{xy}E_r + E_m + E_{idle}) \leq E_{max}, \tag{12a}$$

$$P_y^{min} \leq P_y \leq P_y^{max}, \tag{12b}$$

$$\leq \lambda_{xy} \leq 1, \tag{12c}$$

$$F_x = \sum_{y=1}^{Y} \lambda_{xy}\varphi_y \leq F_{max}, \tag{12d}$$

where E_{max} is the thresholds of the maximum energy consumption, P_y^{min} and P_y^{max} are the minimum and maximum price constraint of computing resources, respectively. The constraint (12d) is given to guarantee number of computing resources rented, F_{max} is the maximum number of computing resources can be rented.

B. NASH EQUILIBRIUM

The Stackelberg game consists of multiple HCSPs and CPTPs, and there is a competitive relationship between multiple CPTPs because of the limited resources of HCSPs. Meanwhile, the rental strategy of each CPTPs affects the price of the computing resources of the HCSPs, which in turn affects the rental strategies of other HCSPs. Therefore, there is a non-cooperative game relationship among the CPTPs.

Nash equilibrium is the optimal solution of a non-cooperative game, which is a stable state of the strategy space among the game participants, i.e., there does not exist a game participant who can achieve more gains by changing the corresponding strategy space. The process of proving Nash equilibrium is as follows.

1) Let the dominant player (CPTP) is X, and the followers (HCSP) is Y, So the corresponding set is finite

2) It is obvious that in the Euclidean space, the strategy space of the game participants is a bounded non-empty closed set and the utility function is continuous on the strategy space.

The first-order and second-order partial derivatives of the effectiveness function are expressed as

$$\frac{\partial U_x}{\partial \lambda} = \frac{\varepsilon \sum\limits_{n=1}^{N} TSD_n \sum\limits_{r=1}^{R} \mathbf{WG}}{R(\lambda+\varepsilon)^2} - (\varphi_y P_y + \tau E_r), \tag{13}$$

$$\frac{\partial^2 U_x}{\partial^2 \lambda} = \frac{-2\varepsilon \sum\limits_{n=1}^{N} TSD_n \sum\limits_{r=1}^{R} \mathbf{WG}}{R(\lambda+\varepsilon)^2} < 0. \tag{14}$$

It can be seen that the second-order partial derivatives of the utility function are less than 0, which can be obtained that the utility function satisfies the strictly concave function property on the strategy space. For this part, the Nash equilibrium algorithm in the non-cooperative game is proposed as shown in TABLE 1.

TABLE 1. Algorithm 1 The Nash equilibrium algorithm in the non-cooperative game.

Algorithm 1 The Nash equilibrium algorithm in the non-cooperative game
1: Parameters set maximum iterations max, X, Y, N, T, S, computing services preference matrix $\mathbf{G}$, the matrix of weight values $\mathbf{W}$, the influence factor ε, tolerance error ϑ.
2: for $i = 1 : max$ do
3: $i = 1 : X$ do
4: Calculate the Euclidean distance for each CPTP $\phi_{ij}(x)$, if $\phi_{ij}(x) \leq \vartheta$, then.
5: Calculate utility function U_y by (11)
6: Calculate $U_x(\lambda)$ first and second order partial derivative of with respect to by (13) and (14)
7: if $\frac{\partial^2 U_x}{\partial \lambda^2} < 0$
8: Output: the Nash Equilibrium solution

IV. STRATEGY OPTIMIZATION

A. PREDICTED THE ACCESS TIMES

In order to further improve the revenue of CPTPs, ensure it can perceive the access times T of heterogeneous computing applications in real time, and achieve the effective deployment. In this section support vector machine algorithm based on artificial fish (SVM-AF) is proposed to optimize the penalty factors and kernel functions to improve the accuracy of model prediction. Therefore, the problem of (12) can be reformulated as

$$\begin{aligned}\max_{T} U_x &= U_x^{proof} - U_x^{rent} - \tau E_x \\ &= \sum_{y=1}^{Y}\sum_{n=1}^{N} TSD_n B_n - \sum_{y=1}^{Y}\lambda_{xy}\varphi_y P_y. \\ &\quad - \sum_{y=1}^{Y}\tau(\lambda_{xy}E_r + E_m + E_{idle})\end{aligned} \tag{15}$$

• Support Vector Machine Regression Prediction

For the sample set (a_i, b_i),$i = 1, 2, \ldots, I$, a_i is the i-th input vector of access times T,b_i is the output term of access times T, and $\phi(\cdot)$ is the nonlinear mapping of the sample mapping in the feature space. So the regression prediction of the least squares support vector machine can be expressed as:

$$f(a) = H^T\phi(a) + n, \tag{16}$$

where H and n are settable parameters, and their values can be determined by the following calculation

$$G = \frac{1}{2}\|H\|^2 + cG_1, \tag{17}$$

where G_1 is the loss function and c is the adjustment factor. The problem is expressed as

$$min(\frac{1}{2}\|H\|^2 + c\sum_{i=1}^{I} e_i^2), \tag{18}$$

$$s.t.\ H^T\phi(a_i) + n + e_i, \tag{18a}$$

$$L(H, n, e_i, \varrho_i) = \frac{1}{2}\|H\|^2 + c\sum_{i=1}^{I} e_i^2,$$

$$- \sum_{i=1}^{I}\varrho_i(H^T\phi(a_i) + n + e_i - b_i) \tag{19}$$

where ϱ_i is the Lagrangian multiplier, e_i is the error. According to the KKT condition

$$\begin{cases}\sum_{i=1}^{I}\varrho_i\phi(a_i) = H \\ ce_i = \varrho \\ \sum_{i=1}^{L}\varrho_i = 0, \\ H^T\phi(a_i) + n + e_i = b_i\end{cases} \tag{20}$$

The regression function is expressed as

$$f(a) = \sum_{i=1}^{I}\varrho_i K(a, a_i) + n, \tag{21}$$

where $K(a, a_i)$ is a kernel function satisfying mercer condition. Considering the characteristics of radial basis function such as wide convergence domain and strong generalization ability, this paper selects the kernel function of radial basis function, which is expressed as

$$K(a_i, a_j) = exp(-\frac{\|a_i - a_j\|^2}{2\sigma^2}), \tag{22}$$

where σ is the core width. In the LSSVM regression model, the kernel function and the adjustment factor c are two key parameters that affect the regression performance, and they are also important points to improve the accuracy of the model.

• Artificial fish swarm algorithm

The artificial fish swarm algorithm finds the optimal solution of the model based on the adaptiveness of each fish by imitating the characteristics of the freedom of fish foraging without adding centralized control factors. The artificial fish swarm algorithm has the advantages of less sensitivity and obvious robustness in the selection of initial values and parameters, which can overcome the local extremes to the highest extent and realize the global search of the optimal solution.

1) FORAGING BEHAVIOR

Fish foraging behavior refers to the process of fish finding food through their own perception and swimming to the location where food is abundant, and in the artificial fish swarming algorithm it is an iterative process that points to a better solution. In the foraging behavior, assume that the initial position of the artificial fish is X_i, in its swimmable range randomly choose a location X_j, where D_{ij} is the distance between X_j and X_i, V is the distance perceived by the individual artificial fish. If the food concentration at the location X_j is higher than X_i, which will be given

$$S_{ij}(i+1) = S_{ij}(i) + \frac{R(n)}{D_{ij}}[S_{ij}(j) - S_{ij}(i)]. \tag{23}$$

Instead, the artificial fish will swim one step at random, which will be have

$$S_{ij}(i+1) = S_{ij}(i) + R(n), \tag{24}$$

where $S_{ij}(i)$, $S_{ij}(j)$ are the elements of column i and j in the parameter matrix of artificial fish X_i, X_j, respectively. $S_{ij}(i+1)$ is the next states of X_i, $R(n)$ is a random number within $[1, n]$.

2) SCHOOLING BEHAVIOR

The schooling behavior refers to the natural aggregation of fish and collective foraging behavior. In this process, the school should avoid overcrowding with neighboring individuals. In the gathering behavior, the initial state is X_i, and the number of fish schools visible in its search domain is f_i, the set can be given by

$$J_{ik} = X_j | D_{ij} \leq V. \tag{25}$$

It is assumed that $J_{ik} \neq \varnothing$, the search center location is X_c, the food concentration at X_c is F_c, and the number of partners is N_c. If the food concentration at X_c is high and the crowding is not high, the artificial fish will swim toward X_c

$$X_{i+1} = X_i + \frac{(X_c - X_i)R_s}{D_{ic}}, \tag{26}$$

where, R_s denotes the random step size of artificial fish. Otherwise, artificial fish will perform foraging behavior. If $N_c = 0$ the artificial fish also performs the foraging behavior.

3) TAIL-CHASING BEHAVIOR

Tail-chasing behavior refers to the behavior of individuals in the school chasing the most active individuals nearby, and in the algorithm, it is the process of advancing to the neighboring optimal solutions. The current position of the artificial fish is X_i, the position with the highest food concentration visible in the search field is X_{max}, and the number of partners in the search area X_{max} is N_{max}. If the food concentration at X_{max} is high and not crowded $F_i < F_{max}$, the individual artificial fish will move towards X_{max}, which can be expressed as

$$X_{i+1} = X_i + \frac{(X_{max} - X_i)R_s}{D_{i,max}}. \tag{27}$$

On the contrary, artificial fish perform foraging behavior.

4) RANDOM BEHAVIOR

In the random behavior, artificial fish swim randomly in the water in order to find food and fish, which have

$$X_{i+1} = X_i + R_s. \tag{28}$$

In the random swimming of artificial fish, the food concentration at the location of each artificial fish individual shall be recorded and compared with the previous location. If it is superior, the previous location shall be replaced.

In order to overcome the shortcomings of support vector machines in solving large-scale random data, in this paper an improved support vector machine algorithm based on artificial fish swarm is proposed, which gives full play to the superior global search ability of the fish swarm algorithm and the efficient ''trans-ductive inference'' advantage of support vector machines. The optimal support vector machine penalty parameter c and kernel function parameter δ are found by the foraging behavior, clustering behavior and tail-chasing behavior of the fish swarm algorithm, which the optimal concentration food coordinates $(bestX_1, bestX_2)$ are found in the fish swarm algorithm. Algorithm 2 is proposed to solve the optimal power allocation in TABLE 2.

TABLE 2. Algorithm 2 Improved SVM algorithm based on artificial fish swarm.

1:Initialization $fish_{num} = 30$, $M_{max} = 40$, $V_s = 1$, $step = 0.3$, $de = 0.618$.
2: The input to the target is divided into a training set and a test set, and each set of data is an input vector $T_m = [T_{m1}, T_{m2}, T_{m3}, T_{m4}, T_{m5}]^T$
3: Normalize the data and use the normalized result as input to the support vector machine.
4: The fish swarm algorithm is introduced to optimize the penalty factor c and kernel function parameters of the SVM.
5: penalty factor $c \geq a_i \geq 0$, $(bestX_1, bestX_2) \in [-10, 10]$,$c = 2^{bestX_1}$,$\delta = 2^{bestX_2}$,$2fish_{num}$ is the fish Matrix, $fish_{num}$ is a whole for the optimization.
6: The fish's perceived distance Visual $V_s = 1$ and step length $p = 0.3$ are used as the parameters of random swimming. By iterative comparison, the maximum value of food concentration of artificial fish is found, both the optimal solution found and the parameters obtained by the search for the optimal are saved.
7: Determine whether the preset maximum number of fish iterations M_{max} is reached, and output the best parameters to the function (18) for prediction.
8: Predicting the number of access times of computing power based on (12), Output access times T.
9: Bring T into formula (15), output the utility function of CPTPs.

B. OPTIMAL RENTAL RATIO OF COMPUTING RESOURCES

In this section, we consider the sub-problem, which optimizes rental ratio of computing resources when access times and price strategy are fixed, the utility function of CPTPs can be written as

$$\max_{\lambda_{xy}} U_x = U_x^{proof} - U_x^{rent} - \tau E_x \tag{29}$$

$$s.t.\ E_x = \sum_{y=1}^{Y} \tau(\lambda_{xy}E_r + E_m + E_{idle}) \leq E_{max}, \tag{29a}$$

$$0 \leq \lambda_{xy} \leq 1, \tag{29b}$$

$$F_x = \sum_{y=1}^{Y} \lambda_{xy}\varphi_y \leq F_{max}, \tag{29c}$$

Assume that computing resources is sufficient for all HSAP. The HCSPs gives a set of computing resource rental prices, and the best rental ratio of CPTP can be obtained through derivation as follows

$$\frac{\partial U_x}{\partial \lambda} = \sum_{n=1}^{N} \frac{TSD_n \sum_{n=1}^{R} WGR\varepsilon}{R(\lambda + \varepsilon)^2} - \varphi_y P_y - \tau E_r, \tag{30}$$

$$\lambda = \left[\sqrt{\frac{\varepsilon \sum_{n=1}^{N} TSD_n \sum_{n=1}^{R} WG}{R(\varphi_y P_y + \tau E_r)} - \varepsilon}\right]^{\pm}. \quad (31)$$

From formula (30), it can be seen that when the rental ratio of the CPTP is 0, the CPTP will not rent computing services. HCSP and CPTP will not benefit from the computing power network, and the quality of computing power application providers cannot be improved. Therefore, the maximum value of computing power resource price exists, which can be obtained

$$P_y^{max} = \frac{1}{\varphi_y} \frac{\sum_{n=1}^{N} TSD_n \sum_{r=1}^{R} WG - \varepsilon R \tau E_r}{\varepsilon R}, \quad (32)$$

When $\lambda = 1$, the CPTP will rent all computing power. Due to the limited resources of HCSP, which will reduce the utility of HCSP. Therefore the minimum value of computing resource price exists, which can be given by

$$P_y^{min} = \frac{1}{\varphi_y} \frac{\varepsilon \sum_{n=1}^{N} TSD_n \sum_{n=1}^{R} WG - \tau E_r R(\varepsilon+1)^2}{R(\varepsilon+1)^2}. \quad (33)$$

In summary, the maximum and minimum computing power resource prices exist. When the price is less than the minimum value P_y^{min}, the current rental price should be raised by the CPTP. When the price is higher than the maximum value P_y^{max}, the current rental price should be lowered by the HCAP. Through multiple adjustments to achieve the best computing power service price, the two will ultimately maximize profits and improve the quality of user experience.

In practice, the service resources of HCAP will not be ignored, which should meet formula (34), and the rental proportion of computing resources should meet $0 \le \lambda \le 1$.

$$F_x = \sum_{y=1}^{Y} \lambda_{xy} \varphi_y \le F_{max}, \quad (34)$$

where, F_{max} is the maximum number of computing service resources.

Note that the optimal problem is convex and meet Slater condition. The optimal value of λ_{xy} can be seen the following dual problem.

$$\min_{\omega \ge 0, \xi \ge 0, \nu \ge 0, \theta \ge 0} \{\max_{\lambda_{xy}} U_x = U_x^{proof} - U_x^{rent} - \tau E_x)\}. \quad (35)$$

The Lagrangian multiplier method is proposed to optimize the game. The Lagrangian function as follows

$$\begin{aligned} L_x(\lambda_{xy}, \omega, \xi, \upsilon, \theta) = &\sum_{y=1}^{Y}\sum_{n=1}^{N} TSD_n B_n - \sum_{y=1}^{Y} \lambda_{xy}\varphi_y P_y \\ &- \sum_{y=1}^{Y} \tau(\lambda_{xy}E_r + E_m + E_{idle}). \\ &+ w(\sum_{y=1}^{Y}(\lambda_{xy}E_r + E_m + E_{idle}) - E_{max}) \\ &+ \xi(\sum_{y=1}^{Y} \lambda_{xy}\varphi_y - F_{max}) \\ &+ \upsilon\lambda_{xy} + \theta(1-\lambda_{xy}) \end{aligned} \quad (36)$$

The optimal rent ratio of computing power resources is shown can be given by

$$\begin{cases} \nabla_{\lambda_{xy}} L_x(\lambda_{xy}, \omega, \xi, \nu, \theta) = 0 \\ \nabla_{\omega} L_x(\lambda_{xy}, \omega, \xi, \nu, \theta) = 0 \\ \nabla_{\xi} L_x(\lambda_{xy}, \omega, \xi, \nu, \theta) = 0 \\ \nabla_{\nu} L_x(\lambda_{xy}, \omega, \xi, \nu, \theta) = 0 \\ \nabla_{\theta} L_x(\lambda_{xy}, \omega, \xi, \nu, \theta) = 0 \end{cases} \quad (37)$$

where ω, ξ and ν are the Lagrangian multipliers. The necessary and sufficient constraints of the Lagrangian function can be given by.

The best rental ratio of computing resources can be derived as follows

$$\begin{cases} 1, \gamma < \dfrac{\varepsilon \sum_{n=1}^{N} TSD_n \sum_{r=1}^{R} \mathbf{WG}}{R(\varphi_y P_y + E_d + E_{mr} - E_{tr})(1+\varepsilon)^2} \\ 0, \gamma < \dfrac{\varepsilon \sum_{n=1}^{N} TSD_n \sum_{r=1}^{R} \mathbf{WG}}{R(\varphi_y P_y + E_d + E_{mr} - E_{tr})} - 1, \\ \sqrt{[\dfrac{\varepsilon \sum_{n=1}^{N} TSD_n \sum_{r=1}^{R} \mathbf{WG}}{R(\varphi_y P_y + E_d + E_{mr} - E_{tr})(1+\gamma)} - \varepsilon]^{\pm}}, other \end{cases} \quad (38)$$

where, γ is the constraint of computing resources for HCSP, the expression of which can be given by

$$\gamma = (F_{max} - \sum_{y=1}^{Y} \lambda_{xy}\varphi_y)^2 - 1 \quad (39)$$

C. OPTIMAL PRICE STRATEGY OF COMPUTING RESOURCES

The optimal price of computing power resources can be obtained by calculating the maximum value of the utility function of HCSP. In this section, we consider the sub-problem, which optimizes price strategy of computing resources when access times and rental ratio are fixed, the utility function of HCSPs can be written as

$$\max_{P_y} U_y = \sum_{x=1}^{X} (P_y - C_y)\varphi_y \lambda_{xy}. \quad (40)$$

$$s.t.\ P_y^{min} \le P_y \le P_y^{max} \quad (40a)$$

The best value of the resource rental ratio of computing services is substituted into equation (40) for derivation. The derivation result is shown in equation (41). If the derivative is 0, it can be inversely solved. The price of HCSP for each update can be obtained as

$$\frac{\partial U_y}{\partial P_y} = \varphi_y \lambda_{xy}^* + \varphi_y (P_y - C_y) \frac{\partial \lambda_{xy}}{\partial P_y}, \tag{41}$$

$$P_y^* = C_y - \lambda_{xy}^* / \frac{\partial \lambda_{xy}}{\partial P_y}. \tag{42}$$

The optimal price of a single computing service resource is dynamic, and it is closely related to the price of computing services for other HCSPs. The iterative formula of computing power service is shown as follows

$$P_y^{t+1} = C_y^t - \frac{\rho \lambda_{xy}^*}{\partial \lambda_{xy} / \partial P_y}, \tag{43}$$

where, t is the number of iterations, P_y^t is the price of computing power service resources at iteration t. C_y^t denotes the cost price of computing power service re-source management at iteration t. ρ is the iteration step length, the value of which gradually decreases with the iteration number.

Because the decline parameter value is too small or too much and is not conducive to Nash equilibrium point approximation, the Cauchy distribution is introduced to optimize the system. The probability density function of one-dimensional Cauchy distribution is

$$f(x) = \frac{1}{\pi} \cdot \frac{t}{t + x^2}, (-\infty < x < +\infty), \tag{44}$$

When $t = 1$, (44) is the standard Cauchy distribution. The generating function of Cauchy distribution random variable can be given by

$$\eta = tan[(\varpi - 0.5)\pi], \tag{45}$$

where ϖ is a random variable on [0,1], where it represents the iteration step ρ. The optimized iteration step size can be obtained as

$$\rho^* = \frac{1}{2\pi} arctan \rho \tag{46}$$

When the number of iterations is $t + 1$, if the utility value of HCSP and CPTP reaches the maximum, the iteration process is terminated. On the contrary, enter the next cycle, and stop the iteration process until the utility value of both is maximum.

V. NUMERICAL RESULTS

A. EXPERIMENTAL SETTING

In this section, simulation results are given to evaluate the system performance and investigate the impact levels of the game optimization strategy of heterogeneous computing application resource scheduling. There are 5 HCSPs, Each HCSP contains multiple computing types, it can be expressed as $D_1 = (CPU, GPU, FPGA)$, $D_2 = (CPU, GPU, NPU)$, $D_3 = (CPU, NPU)$, $D_4 = (CPU, DSP)$, $D_5 = (CPU, ASIC)$. Simulation parameters are stated as Table 3. A single HCSP has 500 computing resources. In the simulation, this strategy is compared with ant colony algorithm (ACA), global optimization strategy (GOS), and QOS priority algorithm (QOS PA) [35], [38].

TABLE 3. Table of parameters for numerical results.

The number of CPTPs	$X = 6$
The number of HCSPs	$Y = 6$
Algorithm resource rental ratio	$\lambda = 0.6$
Number of discrete fetches	$R = 50$
Heterogeneous arithmetic service unit price	$S = 2$
Number of fish	$fish_{num} = 30$
Maximum number of iterations	$M_{max} = 30$
Field of view	$V_s = 40$
Step length	$step = 0.3$

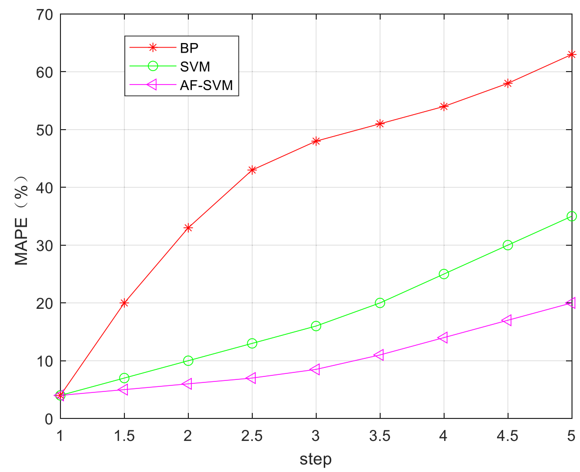

FIGURE 3. The number of applications used vs. reduction rate of energy consumption.

B. SIMULATION RESULT

In order to achieve fair comparison, we optimize the parameters of BP algorithm and SVM algorithm in a similar way. BP algorithm is a general algorithm for training neural networks, which combines optimization methods such as gradient descent and repeats two-stage cycle, propagation and weight update. SVM algorithm is a machine learning algorithm for analyzing data, which is used for classification and regression analysis. In addition, the mean absolute percentage error (MAPE) was applied to evaluate the accuracy of the model.

$$MAPE = \frac{1}{n} \sum_{i=1}^{N} | \frac{\hat{T}_i - T_i}{T_i} 100| \tag{47}$$

where, $\hat{T}_i$ is the number of accesses to real heterogeneous computing resources at the i-th time, T_i is the number of accesses to predicted computing resources at the same time, n is the number of predictions.

Fig.3 shows the MAPE values of three different algorithms BP, SVM, and SVM-AF. The calculation force access prediction process based on historical data is called step=1. New results can be obtained by pre setting. This process is called step=2, and so on. From the simulation results, it can be seen that MAPE increases with the increase of prediction step size. Therefore, we can draw the conclusion that the results become inaccurate as the step size increases. The simulation

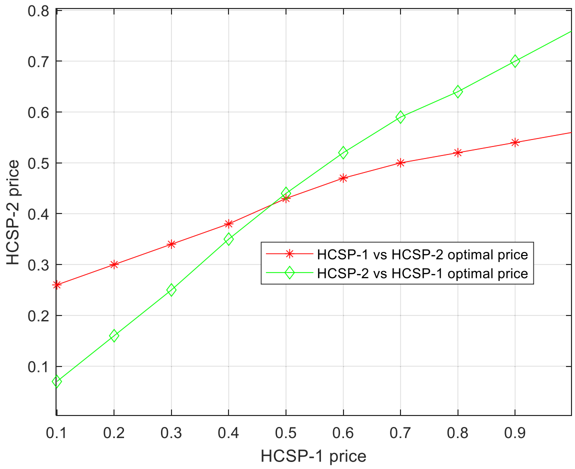

FIGURE 4. Computing price relationship between two HCSPs.

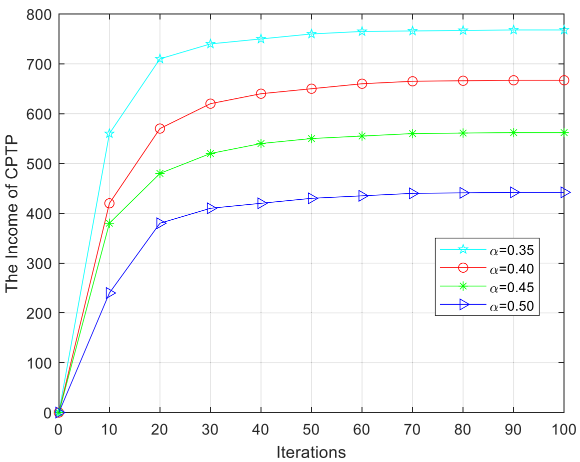

FIGURE 5. The revenue of CPTP vs. the number of iterations.

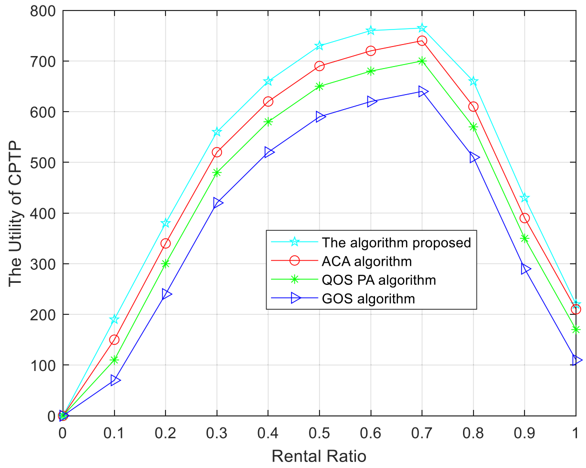

FIGURE 6. The utility of CPTP vs. the rental ratio of computing resources.

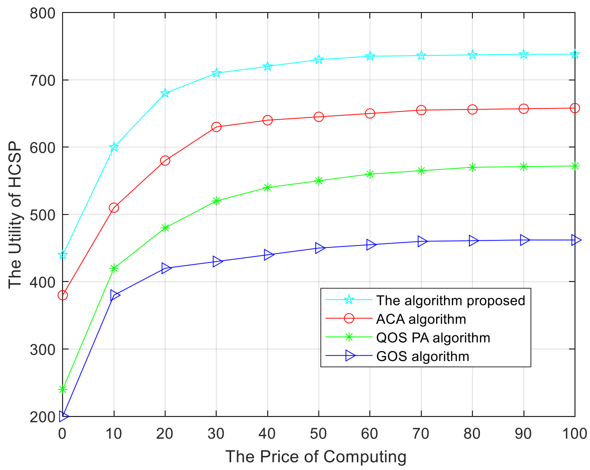

FIGURE 7. The utility of HCSP vs. the price of computing.

results show that the algorithm can achieve the minimum prediction error compared with the other two algorithms. Specifically, when step=5, compared with SVM algorithm, the absolute error of prediction is reduced by 7.3%, and compared with BP algorithm, the absolute error of prediction is reduced by 32.4%.

Fig.4 illustrates the computing price relationship between two HCSPs. The points on the curve represent the optimal pricing strategy of the current HCSPs relative to another HCSPs. It can be observed that the intersection point (0.49, 0.4) of the two curves is the Nash Equilibrium point, which represents that the pricing and utility of both parties reach the optimal level.

Fig.5 illustrates the income of CPTP versus the number of iterations. It can be observed that with the same number of iterations, the popularity of computing resources has a very strong impact on the CPTP income, i.e., as the popularity of heterogeneous computing resources increases, the CPTP income also increases. Besides, from figure 5, it can be seen the algorithm proposed in our paper has good convergence, which basically converges after about 65 iterations.

Fig.6 illustrates the utility of CPTP versus the rental ratio of computing resources. As can be seen, along with the rental ratio of computing resources increases, the utility of CPTP firstly increases and then decreases, which is explained by the fact that with the increase of rental ratio, more computing power rental costs will incur, and the energy consumption of CPTPs for computing power resource scheduling and mainte-nance will also increase, so lead to utility of CPTP decrease.

Fig.7 illustrates the utility of HCSP versus the price of computing. As can be observed from figure 7, along with the pricing of computing increases, the utility of HCSP increases. It can be explained as follows, since a higher price of com-puting can increase the income of HSCPs. However, when the rental price rises to a certain level, it will affect the rental ratio of computing power for HCSP, thus the utility of HCSP will decrease. Furthermore, we also observed that the strategy proposed is superior to other strategies and algorithms.

Fig.8 shows the utility of CPTP versus the number of access times. It can be seen from the figure the utility of

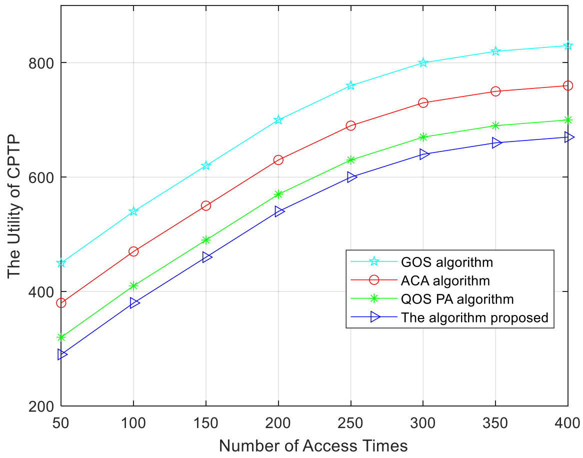

FIGURE 8. **The utility of CPTP vs. the number of access times.**

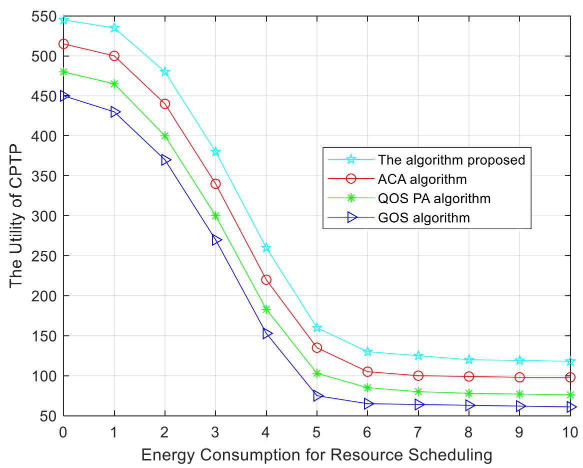

FIGURE 9. **The utility of CPTP vs. the energy consumption for resource scheduling.**

CPTP increases with the number of computing access times increases. And the utility for the optimization strategy in this paper is better than the other three strategies and algorithms.

Fig.9 shows the utility of CPTP versus the energy consumption for resource scheduling. As can be seen, along with the energy consumption for resource scheduling the utility of CPTP firstly decreases and then then stabilize. It can be explained by the fact that with the increase of energy consumption, the cost of energy consumption will increase during resource scheduling. However, when the energy consumption generated in the resource scheduling reaches a certain level, it will affect the rental ratio of computing resources, so the utility of CPTP tends to be stable.

VI. CONCLUSION

In order to solution the challenge of the trade-off between profit and costs in the process of scheduling heterogeneous computing resources, this work establishes a heterogeneous computing resource scheduling model based on Stackelberg differential game, which includes three roles Computing Power Trading Platforms (CPTPs), Heterogeneous Computing Service Providers (HCSPs), and Heterogeneous Computing Application Providers (HCAPs). The objective is to maximize utility function of CPTPs and HCSPs subject to rental ratio, pricing strategy and energy consumption of resource scheduling, which has proved that there exists a Stackelberg Nash Equilibrium (NE) solution. The Support Vector Machine based on an Artificial Fish swarm (SVM-AF) is proposed to predict the access times of heterogeneous computing applications. In addition, the distributed iteration method and Cauchy distribution is adopted to optimize the computing price strategy and improve its convergence performance. The simulation results show that compared with other strategies, the proposed strategy can effectively improve computing revenue of user experience and while reducing energy consumption in the process of resource scheduling.

VII. DISCUSSION

In this study, we establish a heterogeneous computing resource scheduling model based on Stackelberg differential game. The maximization of the system utility function is solved by optimizing the access times, rental ratio, pricing strategy. The SVM-AF is proposed to predict the access times of heterogeneous computing applications. When comparing our results to those of older studies, it must be pointed out that we propose a strategy that can effectively improve the computing revenue of user experience and reduce energy consumption. Moreover, on the basis of considering the efficient computing power provided by heterogeneous computing, we study the trade-off between the cost and profit of computing power resource leasing, so as to effectively improve the revenue of computing power suppliers and provide better guidance for the actual computing power operation. One important future direction is to research the differences in costs and profits of different computing resources and consider the role and significance of operation and maintenance capabilities in the entire heterogeneous computing resource trading process.

REFERENCES

[1] X. Zhou, W. Liang, K. I. Wang, and L. T. Yang, "Deep correlation mining based on hierarchical hybrid networks for heterogeneous big data recommendations," *IEEE Trans. Computat. Social Syst.*, vol. 8, no. 1, pp. 171–178, Feb. 2021.

[2] X. Zhou, X. Xu, W. Liang, Z. Zeng, and Z. Yan, "Deep-learning-enhanced multitarget detection for end–edge–cloud surveillance in smart IoT," *IEEE Internet Things J.*, vol. 8, no. 16, pp. 12588–12596, Aug. 2021.

[3] F. Zheng, C. Wang, Z. Zhou, Z. Pi, and D. Huang, "LEO laser microwave hybrid inter-satellite routing strategy based on modified *Q*-routing algorithm," *EURASIP J. Wireless Commun. Netw.*, vol. 2022, no. 1, Dec. 2022, Art. no. 36.

[4] S. Li, N. Zhang, H. Chen, S. Lin, O. A. Dobre, and H. Wang, "Joint road side units selection and resource allocation in vehicular edge computing," *IEEE Trans. Veh. Technol.*, vol. 70, no. 12, pp. 13190–13204, Dec. 2021.

[5] Z. Zhou, F. Qiu, F. Zheng, and M. Ye, "Neural network joint capacity-power control strategy based on NSGAII-BP for interference suppression in LEO satellite uplinks," *Comput. Elect. Eng.*, vol. 102, Sep. 2022, Art. no. 108093.

[6] C. Lin, P. Peitian, and R. Krithik, "A tensor processing framework for CPU-manycore heterogeneous systems," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 41, no. 6, pp. 1620–1635, Jun. 2022.

[7] C.-F. Juang, W.-Y. Chen, and C.-W. Liang, "Speedup of learning in interval type-2 neural fuzzy systems through graphic processing units," *IEEE Trans. Fuzzy Syst.*, vol. 23, no. 4, pp. 1286–1298, Aug. 2015.

[8] M. J. Cannon, A. M. Keller, H. C. Rowberry, C. A. Thurlow, A. Perez-Celis, and M. J. Wirthlin, "Strategies for removing common mode failures from TMR designs deployed on SRAM FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 66, no. 1, pp. 207–215, Jan. 2019.

[9] Y. Chen and S. Chien, "Configurable texture unit for convolutional neural networks on graphics processing units," in *Proc. IEEE Int. Conf. Artif. Intell. Circuits Syst. (AICAS)*, Mar. 2019, pp. 234–238.

[10] J. Zhu, L. Wang, H. Liu, S. Tian, Q. Deng, and J. Li, "An efficient task assignment framework to accelerate DPU-based convolutional neural network inference on FPGAs," *IEEE Access*, vol. 8, pp. 83224–83237, 2020.

[11] H. Liu, J. Xu, X. Liao, H. Jin, Y. Zhang, and F. Mao, "A simulation framework for memristor-based heterogeneous computing architectures," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 41, no. 12, pp. 5476–5488, Dec. 2022.

[12] S. Chen, Z. Li, B. Yang, and G. Rudolph, "Quantum-inspired hyper-heuristics for energy-aware scheduling on heterogeneous computing systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 6, pp. 1796–1810, Jun. 2016.

[13] Z. Zhang, T. Song, L. Lin, Y. Hua, X. He, Z. Xue, R. Ma, and H. Guan, "Towards ubiquitous intelligent computing: Heterogeneous distributed deep neural networks," *IEEE Trans. Big Data*, vol. 8, no. 3, pp. 644–657, Jun. 2022.

[14] H. Cho, J. Lee, and J. Lee, "FARNN: FPGA-GPU hybrid acceleration platform for recurrent neural networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 7, pp. 1725–1738, Jul. 2022.

[15] O. Ferraz, S. Subramaniyan, R. Chinthala, J. Andrade, J. R. Cavallaro, S. K. Nandy, V. Silva, X. Zhang, M. Purnaprajna, and G. Falcao, "A survey on high-throughput non-binary LDPC decoders: ASIC, FPGA, and GPU architectures," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 1, pp. 524–556, 1st Quart., 2022.

[16] F. Civerchia, M. Pelcat, L. Maggiani, K. Kondepu, P. Castoldi, and L. Valcarenghi, "Is OpenCL driven reconfigurable hardware suitable for virtualising 5G infrastructure?" *IEEE Trans. Netw. Service Manage.*, vol. 17, no. 2, pp. 849–863, Jun. 2020.

[17] Y. Cho, J. Kim, and D. Yu, "Comparative study of CUDA GPU implementations in Python with the fast iterative shrinkage-thresholding algorithm for LASSO," *IEEE Access*, vol. 10, pp. 53324–53343, 2022.

[18] X. Tang and Z. Fu, "CPU–GPU utilization aware energy-efficient scheduling algorithm on heterogeneous computing systems," *IEEE Access*, vol. 8, pp. 58948–58958, 2020.

[19] J. Liu, F. Zhang, H. Li, D. Wang, W. Wan, X. Fang, J. Zhai, and X. Du, "Exploring query processing on CPU-GPU integrated edge device," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 12, pp. 4057–4070, Dec. 2022.

[20] X. Zhou, Y. Li, and W. Liang, "CNN-RNN based intelligent recommendation for online medical pre-diagnosis support," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 18, no. 3, pp. 912–921, May 2021.

[21] Y. Liu, H. Wu, K. Rezaee, M. R. Khosravi, O. I. Khalaf, A. A. Khan, D. Ramesh, and L. Qi, "Interaction-enhanced and time-aware graph convolutional network for successive point-of-interest recommendation in traveling enterprises," *IEEE Trans. Ind. Informat.*, vol. 19, no. 1, pp. 635–643, Jan. 2023.

[22] Q. Zeng, Y. Du, K. Huang, and K. K. Leung, "Energy-efficient resource management for federated edge learning with CPU-GPU heterogeneous computing," *IEEE Trans. Wireless Commun.*, vol. 20, no. 12, pp. 7947–7962, Dec. 2021.

[23] A. Marahatta, S. Pirbhulal, F. Zhang, R. M. Parizi, K. R. Choo, and Z. Liu, "Classification-based and energy-efficient dynamic task scheduling scheme for virtualized cloud data center," *IEEE Trans. Cloud Comput.*, vol. 9, no. 4, pp. 1376–1390, Oct. 2021.

[24] Z. Zhu, J. Zhang, J. Zhao, J. Cao, D. Zhao, G. Jia, and Q. Meng, "A hardware and software task-scheduling framework based on CPU+FPGA heterogeneous architecture in edge computing," *IEEE Access*, vol. 7, pp. 148975–148988, 2019.

[25] H. Khaleghzadeh, M. Fahad, A. Shahid, R. R. Manumachu, and A. Lastovetsky, "Bi-objective optimization of data-parallel applications on heterogeneous HPC platforms for performance and energy through workload distribution," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 3, pp. 543–560, Mar. 2021.

[26] W. Jiang, Y. Zhan, G. Zeng, and J. Lu, "Probabilistic-forecasting-based admission control for network slicing in software-defined networks," *IEEE Internet Things J.*, vol. 9, no. 15, pp. 14030–14047, Aug. 2022.

[27] T. N. Le, X. Sun, M. Chowdhury, and Z. Liu, "AlloX: Allocation across computing resources for hybrid CPU/GPU clusters," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 46, no. 2, pp. 87–95, Jan. 2019.

[28] S. Tang, B. He, S. Zhang, and Z. Niu, "Elastic multi-resource fairness: Balancing fairness and efficiency in coupled CPU-GPU architectures," in *Proc. Int. Conf. High Perform. Comput., Netw., Storage Anal.*, Nov. 2016, pp. 875–886.

[29] X. Fan, Z. Rashidi, B. Ryu, H. Pourmodheji, A. An, J. Liu, and Y. Hu, "GPU resource scheduling using CNN-based deep reinforcement learning," in *Proc. 31st Annu. Int. Conf. Comput. Sci. Softw. Eng.*, Nov. 2021, vol. 96, no. 31, pp. 196–201.

[30] E. S. A. Lozano and A. Gerstlauer, "Learning-based phase-aware multi-core cpu workload forecasting," *ACM Trans. Design Automat. Electron. Syst.*, vol. 28, no. 15, pp. 918–931, May 2023.

[31] J. Du, C. Jiang, A. Benslimane, S. Guo, and Y. Ren, "SDN-based resource allocation in edge and cloud computing systems: An evolutionary Stackelberg differential game approach," *IEEE/ACM Trans. Netw.*, vol. 30, no. 4, pp. 1613–1628, Aug. 2022.

[32] C. Han, L. Huo, X. Tong, H. Wang, and X. Liu, "Spatial anti-jamming scheme for Internet of Satellites based on the deep reinforcement learning and Stackelberg game," *IEEE Trans. Veh. Technol.*, vol. 69, no. 5, pp. 5331–5342, May 2020.

[33] M. Tao, K. Ota, M. Dong, and H. Yuan, "Stackelberg game-based pricing and offloading in mobile edge computing," *IEEE Wireless Commun. Lett.*, vol. 11, no. 5, pp. 883–887, May 2022.

[34] A. Hazra, M. Adhikari, T. Amgoth, and S. N. Srirama, "Stackelberg game for service deployment of IoT-enabled applications in 6G-aware fog networks," *IEEE Internet Things J.*, vol. 8, no. 7, pp. 5185–5193, Apr. 2021.

[35] M.-C. Lee, A. F. Molisch, N. Sastry, and A. Raman, "Individual preference probability modeling and parameterization for video content in wireless caching networks," *IEEE/ACM Trans. Netw.*, vol. 27, no. 2, pp. 676–690, Apr. 2019.

[36] X. Chen, J. Xu, and D. Long, "Resource scheduling algorithm of cloud computing based on ant colony optimization-shuffled frog leading algorithm," *J. Comput. Appl.*, vol. 38, no. 6, pp. 1670–1674, Oct. 2018.

[37] J. Li, H. Chen, Y. Chen, Z. Lin, B. Vucetic, and L. Hanzo, "Pricing and resource allocation via game theory for a small-cell video caching system," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 8, pp. 2115–2129, Aug. 2016.

[38] P. Varshney and Y. Simmhan, "Characterizing application scheduling on edge, fog and cloud computing resources," 2019, *arXiv:1904.10125*.

RUI SHE received the M.S. degree in communication and information engineering from North China Electric Power University, China, in 2020. She is currently an Engineer with Research Institute of China Telecom Company Ltd., Beijing, China. Her research interests include cloud computing, heterogeneous computing, edge services, massive MIMO, NOMA, energy efficiency of wireless communications, physical layer security, wireless resource management, and optimization technology.

WEI ZHAO (Member, IEEE) received the B.Eng. degree from North China Electric Power University, China, in 2008, and the Ph.D. degree from the Beijing University of Posts and Telecommunications, Beijing, China, in 2011. In 2011, he joined the School of Electrical and Electronic Engineering, North China Electric Power University. From 2016 to 2017, he visited King's College London, U.K., as a Visiting Scholar. He holds many patents, three EI papers, an SCI, and multiple other types of articles. His research interests include wireless security, massive MIMO, NOMA, and the energy efficiency of wireless communications.