The Task Collaborative Migration Method for Marine IoT based on Edge Computing-RESTful Architecture

Yifan Hu*
School of information and control engineering,
Qingdao University of Technology
Qingdao, China
Shandong Computer Science Center and Shandong Provincial Key Laboratory of Computer Networks,
Qilu University of Technology (Shandong Academy of Sciences)
Jinan, China
huyif12@163.com

Fuqiang Wang
Shandong Computer Science Center and Shandong Provincial Key Laboratory of Computer Networks,
Qilu University of Technology (Shandong Academy of Sciences)
Jinan, China
wangfq@sdas.org

***Abstract*—Marine Internet of Things (Marine IoT) has garnered increasing interest in monitoring oceanic environments. But establishing a Marine IoT system for observing and processing marine environmental data presents various challenges. These include limited underwater communication bandwidth, and uneven distribution of tasks among data-sensing devices. This paper proposes an Edge Computing-RESTful (ECR) architecture to efficiently integrate and process the substantial volume of data gathered through marine observations. 1) The ECR method is introduced for task collaborative migration, aiming to schedule and manage the tasks within the Marine IoT system. 2) Through unified scheduling and management of tasks, the edge gateway can facilitate migration, scheduling, and interaction of task loads among various edge sensing devices. The proposed dynamic and adaptive approach matches computing tasks with edge sensing devices, minimizing energy consumption and migration delay within the Marine IoT system. Simulation results validate the superiority of the proposed method for Marine IoT, showcasing improved performance metrics such as reduced time delays and lower energy consumption during task collaborative migration.**

Keywords—Edge computing; RESTful; Collaborative migration; Marine IoT

I. INTRODUCTION

The Marine IoT System is designed to monitor diverse marine environmental data through several types of sensors, encompassing variables such as wind, temperature, humidity, pressure, waves, flow, nutrients, COD, dissolved oxygen, pH, and underwater video [1]. This comprehensive marine environmental observation network seamlessly integrates broadband, wireless, and underwater acoustic communication, along with maritime sensor networks [2]. The Marine IoT architecture diagram is shown in Fig. 1.

However, several challenges impede the development of an efficient Marine IoT system. The bandwidth for marine communication is constrained, posing difficulties in handling the extensive scale of observational data. Additionally, the heterogeneity of sensing devices introduces imbalances in resource allocation and computing loads. Hassan *et al.* [3] emphasized the necessity for a rapid prediction mechanism and efficient information dissemination within the marine sensing and communication network to address emergencies effectively. To mitigate issues such as data loss and transmission delays in sensor data, edge computing is proposed in the paper. This approach involves processing data close to the source, facilitating swift responses within the system [4]. Nevertheless, current solutions lack the flexibility required for scheduling and migrating task, hindering effective interaction between sensing devices and tasks.

Therefore, it is necessary to introduce edge computing mechanism into the Marine IoT system to reduce data transmission volumes [5] and enhance data observation efficiency [6]. Moreover, addressing the heterogeneity and disorder of sensing devices is paramount for efficient Marine IoT application development. A unified system structure is proposed to solve task load distribution issues. The implementation of a distributed RESTful (Representational State Transfer) mechanism aims to improve the edge computing performance of distributed sensing devices and establish a unified scheduling mechanism for task loads.

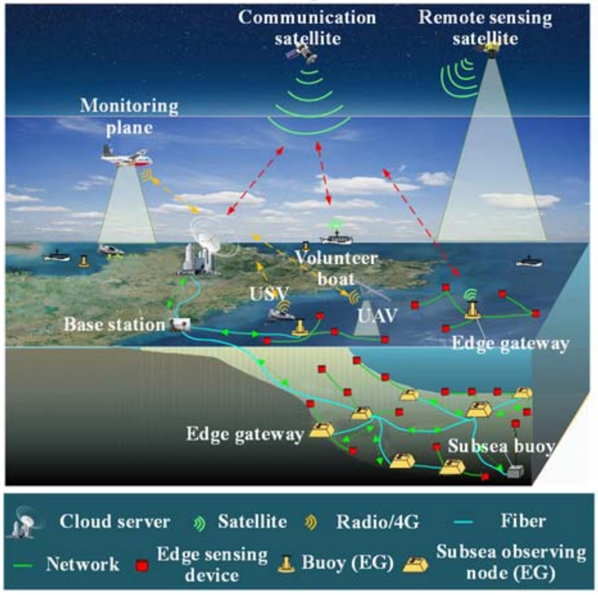


Fig. 1 Diagram of Marine IoT system architecture

The RESTful (Representational State Transfer) mechanism holds promise for enhancing the performance of edge computing within Marine IoT systems. It offers the potential to streamline network complexity and bolster the scalability of the network system. Among the protocols aligned with RESTful principles, HTTP stands out [7]. Cai and Qi [8] introduced an IoT overlay network architecture method based on RESTful principles. They designed a resource adaptation layer equipped with standardized interfaces to facilitate the fusion of heterogeneous subnets and cross-protocol communication. This approach shields differences in resource

---

This work is supported by Shandong Key Laboratory Foundation of Computer Networks (No. SDKLCN-2022-02), Shandong Provincial Natural Science Foundation (No. ZR2023MF052).

* Corresponding author: Yifan Hu (huyif12@163.com)

access among underlying physical devices and devises protocols for device access to support Internet-compatible communication. Xu and Chao [9] proposed a RESTful interface that enables easy external access to devices like smart homes, bridging the gap between smart home systems and traditional networks. Additionally, Xu et al. [10] presented a zone-oriented architecture for the IoT web, extending the foundational architecture of modern Web systems to establish a coordinated and unified RESTful system architecture. However, these methods did not integrate edge computing with RESTful mechanism. It is necessary to dynamically manage tasks on edge devices efficiently and balance task loads.

To address the characteristics and challenges specific to Marine IoT, this paper introduces an Edge Computing-RESTful (ECR) architecture for constructing the Marine IoT system. It proposes an ECR-based task collaborative migration method (ECR method) to schedule and migrate task loads, maintaining network device balance. Distributed devices such as edge sensing devices and gateways are supported through this unified and coordinated ECR model. In this model, resource-rich edge sensing devices undertake migrated tasks from the edge gateway. Tasks are dynamically scheduled based on resource types and the computing abilities of edge sensing devices, minimizing data transmission overhead and optimizing task execution efficiency. The contributions of this work are as below:

1) Establishing a unified adaptation interface for heterogeneous sensing devices within the Marine IoT system through a unified and coordinated distributed ECR architecture. This shields the impact of diversity of sensors on upper-layer applications. The data volume could also be reduced by EC preprocessing mechanisms of edge gateway and device.

2) Introducing a unified scheduling and management mechanism for task loads under the ECR method, addressing resource constraints of sensing devices. This enables the selection of optimal tasks for collaborative migration, achieving dynamic adaptive matching between computing tasks and edge sensing devices while reducing energy consumption during task migration.

The remainder of this paper is structured as follows: Section 2 presents the ECR-based Marine IoT model and task migration model. The task collaborative migration method is described in details in Section 3. Section 4 evaluates the simulation results of the method. Finally, Section 5 concludes the work.

II. MODEL DESCRIPTION

A. ECR based Marine IoT model

The proposed Marine IoT system encompasses two types of edge devices: Edge Sensing Device (ESD) and Edge Gateway (EG). The ESD is equipped to sense, pre-process, and transmit data. It also possesses the capability to migrate tasks and collaborate on task computing. Due to limited terminal resources and the dynamic marine observing environment, tasks need to be balanced and migrated from constrained devices to resource-rich edge sensing devices.

The EG has the capacity to pre-process, merge, and store various heterogeneous data sent from the ESD. It uploads the processed valid data to the cloud server and manages task migration between ESDs. There are edge gateways, cloud server, edge sensing devices, and tasks (migrated) in the multi-layer model of Marine IoT.

The ESDs transmit data to EGs via wired or wireless communication. EGs preprocess and fuse data, uploads data to storage layer for storage. EG also migrates part of applications, data materials, and services processed in the cloud server for local processing, benefiting from reduced transmission delay and data volume. Subsequently, EG transmits data to the cloud server via wired (optical cable) communication.

The edge computing model, as depicted in Fig. 2, utilizes a RESTful scheduling mechanism to facilitate the deployment and migration of resources between EG and ESD. This mechanism also enables interaction between different edge sensing devices. Resources are accessed and exchanged between ESDs, and between ESDs and EGs through REST rules. EG plays a pivotal role in task migration between ESDs, achieving cooperative processing of tasks and ensuring efficient utilization of resources.

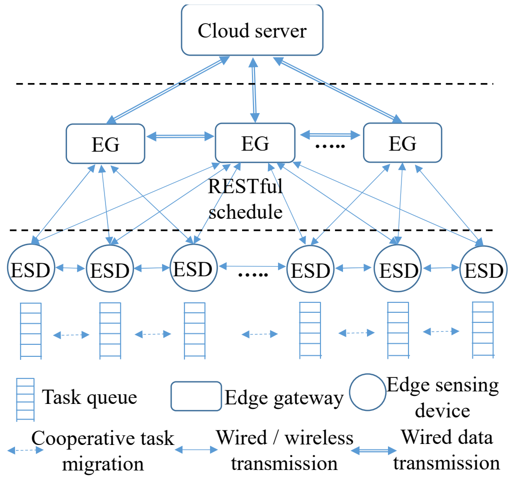


Fig. 2 Multi-layer model diagram of Marine IoT

B. Task migration model

The Marine IoT system consists of l EGs denoted as G $=\{G_1, G_2, G_3,\ldots, G_k,\ldots, G_l\}$, where $0<k\le l, l\in N^+$. $\{G_k\}=\{URI_{G_k}, P_{G_k}, A_{G_k}\}$ represents the specific parameter configuration set of each EG. URI_{G_k} represents URI address of G_k. P_{G_k} represents computing ability (in *Bit*/*s*) of G_k. A_{G_k} represents availability of G_k. $A_{G_k}=1$ is that G_k is in idle status and available for scheduling management.

The number of ESDs is assumed n， ESDs is denoted as D $=\{D_1, D_2, D_3,\ldots, D_i,\ldots, D_n\}$, where $0<i\le n, n\in N^+$. $\{D_i\}=\{URI_{D_i}, A_{D_i}, B_{D_i}^0, P_{D_i}^0, Q_{D_i}, CS_{D_i}, E_{D_i}^r\}$ represents the specific parameter configuration set of each edge sensing device. URI_{D_i} represents URI address of D_i. A_{D_i} represents availability of D_i, if D_i is in the status of idle and accepting task cooperation, $A_{D_i}=1$, else $A_{D_i}=0$. $B_{D_i}^0$ is shared transmission bandwidth of D_i, $E_{D_i}^r$ is remaining energy of D_i, $P_{D_i}^0$ means the computing ability of D_i. Load evaluation $Q_{D_i}=\begin{cases}1-v_{D_i}/B_{D_i}, v_{D_i}<B_{D_i}\\ 0, \quad v_{D_i}\ge B_{D_i}\end{cases}$, where v_{D_i} is current bandwidth transmission rate of D_i. The larger Q_{D_i} is, the lower load of D_i is, and the stronger task-collaboration

capability is. CS_{D_i} is the Credit score of D_i, D_i needs to be able to complete more tasks steadily and reliably in the near future, in order to have higher Credit score. $E^r_{D_i}$ is the current remaining energy of D_i.

All the ESDs would send information to the closest EGs, according to the schedule probability $p_{D_i G_k}$. The task scheduler in EG would arrange each ESD in descending order, according to the value of task-collaboration capability f_{D_i} of ESD. $p_{D_j G_k}$ and f_{D_i} would be explained in detail in Subsection 5.2. Then the task scheduling queue D' $\{D_1', D_2',..., D_i',..., D_n'\}$ is formed, in order to match tasks and ESDs.

There are m tasks to be migrated from all ESDs. $T=\{T_1, T_2, T_3,..., T_j,..., T_m\}$ is used to distinguish these tasks, where $0 < j \le m, m \in N^+$. $\{T_j\}$={ $URI^0_{T_j}$, URI_{T_j}, M_{T_j}, B_{T_j}, V_{T_j}, $t^d_{T_j}$, $t^w_{T_j}$ } represents specific parameter configuration information of T_j in EG, where $URI^0_{T_j}$ is the initial URI address of T_j, URI_{T_j} stores the current URI address of T_j. B_{T_j} is the communication link transmission bandwidth (in *Bit*/*s*) required for migrating task. $t^d_{T_j}$ is the latest deadline processing time for T_j. $t^w_{T_j}$ is the waiting time in the task scheduling queue after the task information is transmitted to EG. M_{T_j} is the data size (in *Bits*) of T_j. V_{T_j} is the computing value of the task T_j. Therefore, the task priority Pr_{T_j} can be expressed as $\mathrm{Pr}_{T_j} = V_{T_j} / M_{T_j}$. The requested task transmission bandwidth should also not exceed the maximum bandwidth of the communication link B^0, $\sum_{j=1}^{m} B_{T_j} \le B^0$. The task information $\{T_j\}$ is stored in the task scheduler of the EG G_k. The task scheduler arranges the tasks sent by devices in descending order of priority, in order to form a pending task scheduling queue $T'\{T_1', T_2', ..., T_j',..., T_k'\}$. T_k' represents the lowest priority task. After the migration task is completed, the computing result of the task can be transmitted back to the device to which the task originally belongs, according to initial URI address $URI^0_{T_j'}$ of task T_k'.

The objective of the task migration model is to minimize the overall power consumption of the Marine IoT system while satisfying various task load delay requirements.

III. TASK COLLABORATIVE MIGRATION METHOD

In our proposed Marine IoT system, the majority of real-time tasks for the ESDs are executed locally. In the event of an abnormal occurrence, such as load saturation or insufficient service capabilities on a device to meet task requirements, the task would be locally migrated to alternative ESDs with the assistance of the EG. The EG assumes the responsibility of managing task schedules and distribution, but it does not handle the computation and processing of tasks. Instead, other resource-rich ESDs collaborate to execute the task, and only the results are transmitted back to the original ESD. This approach aims to minimize the volume of transmitted data.

A. Edge sensing device capability function for task collaborative migration

The ESD should meet following constraints, to complete the task collaborative migration.

1) The bandwidth B_{D_j} occupied by migrating the task T_j from device D_j should not exceed shared transmission bandwidth $B^0_{D_i}$ provided by D_i. The computational cost P_{D_j} per unit time for migrating T_j cannot exceed the computing capacity $P^0_{D_i}$ of D_i.

2) The sum of task migration time and the edge sensing device's task processing time should not exceed the task's final deadline time.

$$t^c_{D_i} + t^m_{D_i} \le t^d_{T_j} \tag{1}$$

where D_i means the edge sensing device for task cooperation, $t^c_{D_i}$ represents D_i's task cooperation and computing time. $t^m_{D_1}$ represents D_i's task migration time. $t^d_{T_j}$ is the deadline processing time for task cooperation which is defined when D_i initiated migration of task T_j.

3) After the edge sensing device completes the migration task, the remaining energy should not be lower than specified energy threshold:

$$E^r_{D_i} \text{-} E^m_{D_i} \text{-} E^c_{D_i} \ge E_0 \tag{2}$$

where $E^m_{D_i}$ is the transmission energy consumption of D_i for task migration, $E^c_{D_i}$ is the energy consumption of D_i for task-collaboration computing, $E^r_{D_i}$ is the remaining energy of D_i before task T_j migration starts, and E_0 is the prescribed energy threshold.

As the ESDs participating in task collaboration are heterogeneous and independent, EG will establish a fair and reliable evaluation mechanism for task-collaboration capability of edge sensing devices. The initial task-collaboration capability function $f^0_{D_i}$ of D_i in the task schedule is as shown in equation (4). The larger the value of $f^0_{D_i}$ is, the greater the task-collaboration capability of D_i could be. Then the priority order of edge sensing devices in the task queue is formed, according to function $f^0_{D_i}$.

$$f^0_{D_i} = A_{D_i} B^0_{D_i} P^0_{D_i} CS_{D_i} Q_{D_i} t^e_{D_i} \tag{3}$$

where CS_{D_i} is credit score of D_i. Device credit evaluation can measure the credibility of the device. It is accumulated by the device's recent good task completion behavior. The device's credibility is evaluated using the interactive data set sampled by the time window. D_i should be able to complete large number of tasks in a stable and reliable manner in the near future, in order to obtain a high credit score. The credit evaluation function is cited from [11]:

$$CS_{D_i} = CS^0_{D_i} + CS^g_{D_i} - CS^r_{D_i} \tag{4}$$

where $CS^0_{D_i}$ is D_i's basic Credit Score, $CS^g_{D_i}$ is D_i's gain Credit Score for maintaining good behavior, $CS^r_{D_i}$ is D_i's debit Credit Score for task-collaboration failure. The equation is as follow [11].

$$\begin{cases} CS_{D_i}^{0} = V_{D_i} \\ CS_{D_i}^{g} = a\,ln(\sum_{j=1}^{n} \frac{V_{D_i j}}{V_{D_i}}) + n \\ CS_{D_i}^{r} = (\sum_{l=1}^{x} \frac{V_{D_i l}}{V_{D_i}}) \times \varepsilon^{x} \end{cases} \tag{5}$$

where V_{D_i} represents the max task value that D_i has completed recently, $V_{D_i j}$ is the j-th task's value that D_i has recently successfully completed, a is gain controlling factor, n is the number of completed tasks in the time window. $V_{D_i l}$ is the l-th task's value that D_i recently has not completed, ε is credit deduction controlling factor, x is the number of recent failures.

B. Task collaborative migration process

The migration process of task T_j of D_j is as below:

1) Device selection of EG. Each device would initially choose an EG and add to its task scheduler. Then its task migration would be managed by this EG. Based on the idea of proximity principle, the probability that device D_j would choose G_k for managing task migration is:

$$p_{D_jG_k} = \frac{\sum_{k=1}^{m} \sqrt{t_{D_jG_k}^{m}}}{\sqrt{t_{D_jG_k}^{m}}} \tag{6}$$

where $t_{D_jG_k}^{m}$ is time delay of task migration between G_k and D_j. G_k with the highest probability will be selected as the gateway to manage task-migration of D_j.

2) Periodic status reporting. D_j periodically reports its status information to G_k, including its resource information $\{D_j\}$. G_k periodically updates the information. If G_k cannot receive information from D_j within the prescribed period of time, D_i is considered to be failed, its uncompleted tasks will be migrated to other replaced device by G_k, and the computing result would not return to D_j again. Else if G_k finds that the capability of D_j cannot meet the task requirements, the task would be migrated to other collaborative device by G_k, and the computing result will be returned to D_j.

3) Task Information transmission. When D_j plans to migrate task T_j, the task information $\{T_j\}$ is placed in the migration request packet, and sent to G_k. If current $A_{G_k} = 0$, D_j chooses another G_{k+1} to send migration request when $p_{D_jG_{k+1}}$ value is the highest and $A_{G_{k+1}} = 1$.

4) Task Queuing. G_k sends GET order to URI address of D_j, to obtain task information $\{T_j\}$ from D_j. Then G_k places $\{T_j\}$ into its task queue T '$\{T_1$', T_2', ..., T_j',…, T_k'$\}$ of task scheduler by task priority Pr_{T_j}. After that, G_k starts to manage task-migration from T' in order.

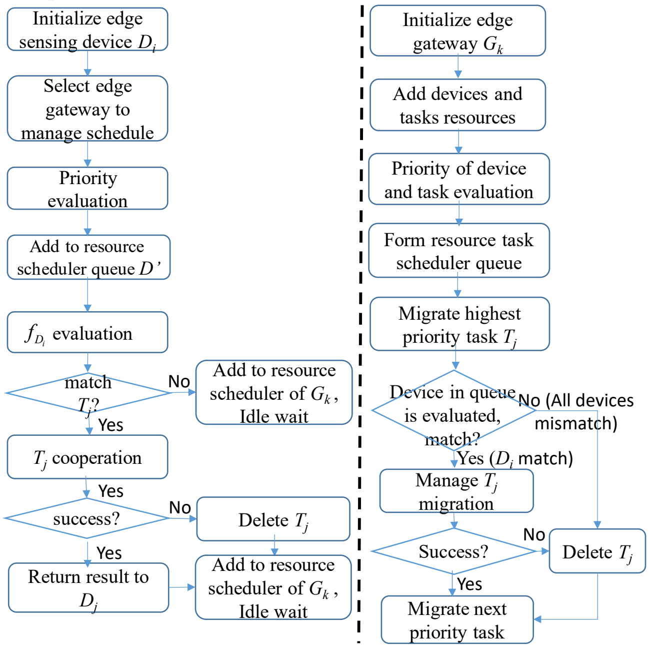
5) Device Selection for Collaboration. G_k decides and selects the collaborative object D_i that is most suitable for T_j's migration request from the task queue D' $\{D_1$', D_2', ..., D_i',…, D_n'$\}$ of the task scheduler, according to $\{T_j\}$ information. D_i's collaborative capability function to complete task T_j belonging to D_j is $f_{D_i} = \frac{f_{D_i}^{0}}{\sqrt{t_{D_i}^{m}}} \times (1 - \frac{E_{D_i}^{m} + E_{D_i}^{c}}{E_{D_i}^{r} - E_0})$. $t_{D_i}^{m}$ is task migration time delay from D_i to D_j. $E_{D_i}^{m}$ is D_i's transmission energy consumption for task migration. $E_{D_i}^{c}$ is D_i's energy consumption for task-collaboration computing. G_k sequentially calculates f_{D_i} of each candidate edge sensing device to complete the task T_j cooperation according to the priority order of the queue D'. The edge sensing device D_x that has obtained the max value $\max f_{D_x}$ would be selected as the optimal task-collaboration device to complete T_j cooperation.

6) Task migration execution. G_k sends PUT order to URI address of D_x, to send task information $\{T_j\}$ to D_x, and notify D_x to prepare for task migration.

7) Task update and migration. G_k sends PUT order to T_j, to update current URI address URI_{T_j} of T_j, including D_x's URI information that T_j should be migrated to. Then the migration of T_j from D_j to D_x would be started. After that, G_k starts to perform operations on next priority task T_{j+1} in T', and goto step 5).

8) Task Completion and Return. After D_x has completed computation of task T_j, if D_x is collaborative device, it returns the computing result to device D_i that T_j originally belongs to, according to initial URI address $URI_{T_j}^{0}$ of T_j. Else if D_x is replaced device, it would not return the computing result. Then the task migration process is completed.

If T_j is unable to meet the deadline processing time requirements after being matched with all ESDs in D', the task's computation will be abandoned. T_j will be removed from the task scheduler queue. G_k will then begin performing operations on the next priority task T_{j+1} in T', and proceed to step 5). This process will be repeated until the task queue T' is empty and all current tasks' computations in G_k are completed.



(a) Flowchart for ESD (b) Flowchart for EG

Fig. 3 Diagram of task cooperative migration process for ESD and EG

The calculation of each task is separate and not dependent on the others. Given the limitation of resources, each independent computation event must minimize the delay and energy consumption of the collaborative capability function for device D_x. This presents a multi-constrained optimization problem, as shown in equation (7). The flowchart for EG and task migration are presented in Fig. 3.

$$\max f_{D_x} = \frac{A_{D_x} B_{D_x}^0 P_{D_x}^0 CS_{D_x} Q_{D_x} t_{D_x}^e}{\sqrt{t_{D_x}^m}} \left(1 - \frac{E_{D_x}^m + E_{D_x}^c}{E_{D_x}^r - E_0}\right) \tag{7}$$

$$\begin{cases} B_{T_j} \le B_{D_x}^0 \quad P_{T_j} \le P_{D_x}^0 \\ t_{D_x}^c + t_{D_x}^m \le t_{D_x}^0 \\ E_{D_x}^r - E_{D_x}^m - E_{D_x}^c \ge E_0 \end{cases}$$

IV. RESULTS AND DISCUSSION

A. *Model and Assumption*

The performance of the proposed ECR based task collaborative migration method is investigated by computer simulations with MATLAB. The Joint Cooperation Protocol (JCP) and Communication Cooperation Protocol (CCP) proposed by [13] also adopts the task collaborative migrating mechanism, which is suitable for comparing the ECR method. The Prediction & offloading protocol (P&O) proposed by [12] adopts task offloading strategy, which is helpful to compare the task collaborative migrating and task offloading strategies. To analyze the performance characteristics of the collaborative migration method, the ECR would be compared with JCP [13], CCP [13] and P&O protocol [12] in terms of energy consumption, standard deviation of energy consumption, task-migration time delay, and success rate of task migration.

In order to evaluate performance of these four protocols, they are simulated under the same conditions: There are 20 EGs and 500 ESDs placed randomly across a 10 *Km*×10 *Km* underwater network area. Initial energy of ESDs is 4000 *mAH*. The wireless and wired data transmission bandwidths are 100 *Kbit/min and* 50 *Kbit/s*. Half of ESDs adopt wireless transmission. Each ESD is randomly arranged around EGs. During the simulation, ESDs freely interact with the EGs, following the IEEE 802.11 protocol cluster standard. The comparison experiment is performed on 5 sets of tasks with 100 number of data tasks to be migrated on the edge sensing devices. The size of each computing task to be migrated is randomly distributed in $M_{T_j} \in [100Kbits, 2.5Mbits]$. The edge sensing devices generating computing tasks would randomly send migration service request with a request frequency f = 5 times per *min*. Each time the request includes 5 consecutive tasks, the sampling period of credit score, load, task acceptance rate and cooperation success rate are 60 *s*. The simulation runs 10 times, each lasting 2 hours. The average simulation results across these runs are then calculated and presented in Figures **4-6**. MATLAB 2014b serves as the simulation platform, generating random task data within the defined range and simulating the corresponding task parameters for computation.

Different collaborative migration protocols are employed during the simulation process based on the task's data scale and priority. These protocols aim to select devices that match the corresponding cooperative abilities for each task.

B. *Performance evaluation*

In Fig. 4, it is evident that the energy consumption values for the ECR, JCP, CCP, and P&O protocols increase over time, as task collaborative migration continuously consumes system energy. The proposed method demonstrates superior performance, with lower energy consumption and communication overhead compared to JCP, CCP, and P&O, which follow with energy consumption rates of 17%, 24%, and 27% respectively. This is attributed to the dynamic scheduling of computing tasks based on resource type and edge sensing device computing ability, which minimizes energy consumption during data transmission and maximizes application performance.

In the Marine IoT environment with limited transmission bandwidth, the P&O protocol experiences access congestion as a large number of tasks gather at the gateway, leading to higher energy consumption due to overloaded data links. On the other hand, in the ECR method, efficient migration of tasks between ESDs results in lower energy consumption compared to P&O.

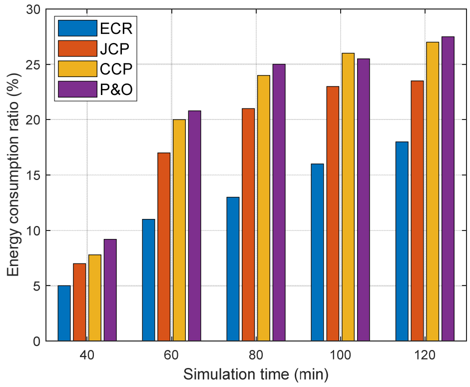


Fig. 4 Average energy consumption rate *vs* time in minutes

Fig. 5 illustrates the average energy consumption rate as a function of the size of migrated tasks in Mbit. The four protocols show similar energy consumption at small task values ($M_{T_j} \in (0, 0.5]$), but as the task size increases, the energy consumption rate for the ECR method is significantly lower compared to the JCP, CCP, and P&O protocols, with differences of 4.4%, 6.2%, 7.2%, and 7.1% respectively.

This indicates the effectiveness of the ECR method in dynamically matching tasks to ESD based on their cooperation ability and task size. The ECR method optimizes energy consumption by selecting ESDs with higher credit scores and lower energy consumption for task migration.

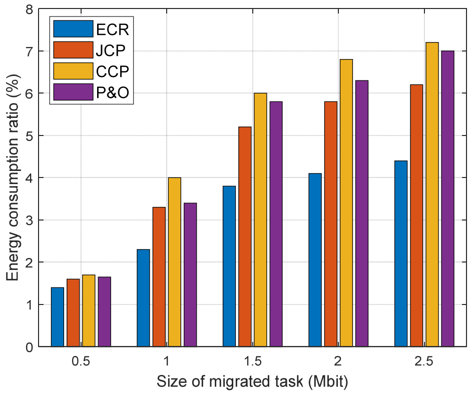


Fig. 5 Average energy consumption rate *vs* size of migrated task in bits

In Fig. 6, the time delay curves for the four protocols show an increase over time, with the ECR method performing 10-17% better than the other protocols. This demonstrates the efficient use of transmission bandwidth and reduced overall delay in the ECR method. The reason is that the ECR can dynamically match tasks to edge sensing devices with better QoS parameters (such as more remaining energy, better device credibility for recent task completion behavior, and larger shared transmission bandwidth). The global optimal task-scheduling scheme could also be quickly obtained, then the requirements of minimizing migration delay of each task can be meet.

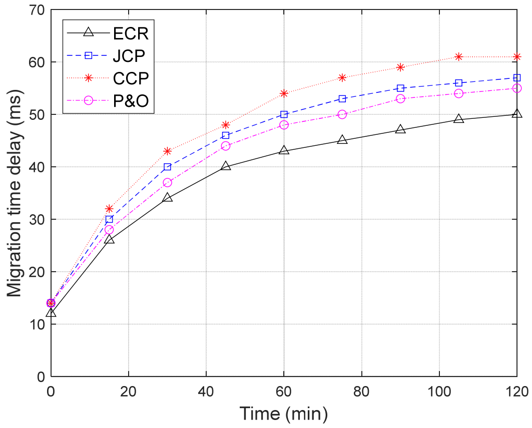


Fig. 6 Average migration time delay *vs* time

V. CONCLUSION

The paper addressed the various challenges present in the recent Marine IoT system, and proposed a multi-layer model based on ECR to efficiently integrate and process large volume of marine observing data. Additionally, the paper introduced a task collaborative migration method based on ECR to schedule and migrate tasks within the Marine IoT system. This method utilized ECR to select the optimal ESD for collaborative migration, taking into consideration energy consumption, task migration delay, and transmission bandwidth limitations. Simulation results demonstrated that the proposed ECR-based task collaborative migration method outperformed other collaborative task migration strategies (JCP and CCP) and edge computing offloading strategy (P&O). Overall, the paper provides valuable insights and solutions for addressing the problems of Marine IoT systems.

REFERENCES

[1] Y. Song *et al*., "Internet of Maritime Things Platform for Remote Marine Water Quality Monitoring," in *IEEE Internet of Things Journal*, vol. 9, no. 16, pp. 14355-14365, 2022.

[2] J. Luo, Y. Chen, M. Wu and Y. Yang, "A Survey of Routing Protocols for Underwater Wireless Sensor Networks," in *IEEE Communications Surveys & Tutorials*, vol. 23, no. 1, pp. 137-160, Firstquarter 2021.

[3] M. M. Hassan, J. Abawajy, M. Chen, *et al*., "Special section on Cloud-of-Things and edge computing: Recent advances and future trends," in *Journal of Parallel and Distributed Computing*, vol. 133, pp. 170-173, 2019.

[4] H. H. Esmat, B. Lorenzo and W. Shi, "Toward Resilient Network Slicing for Satellite–Terrestrial Edge Computing IoT," in *IEEE Internet of Things Journal*, vol. 10, no. 16, pp. 14621-14645, 15 Aug.15, 2023.

[5] J.C. Yang, J.B. Wen, B. Jiang, *et al*., "Marine depth mapping algorithm based on the edge computing in Internet of things, " in *Journal of Parallel and Distributed Computing*, vol. 114, pp. 95-103, 2018.

[6] Y. Jiang, X. Xu, H. Gao, *et al*., "LBlockchainE: A Lightweight Blockchain for Edge IoT-Enabled Maritime Transportation Systems," in *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 2, pp. 2307-2321, Feb. 2023.

[7] B. Costa, P.F. Pires, F.C. Delicato, P. Merson, "Evaluating REST architectures Approach, tooling and guidelines," in *Journal of Systems and Software*, vol. 112, pp. 156-180, 2016.

[8] Y. Cai, D.Y. Qi, "Physical Control Framework and Protocols Design for Cyber-Physical Control System," in *International Journal of Distributed Sensor Networks*, vol. 13, no. 7, pp. 1-12, 2017.

[9] Z. Xu, L. Chao and X. Peng, "T-REST: An Open-Enabled Architectural Style for the Internet of Things," in *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4019-4034, June 2019.

[10] Z.W. Xu, C. Zeng, "Zone-Oriented Architecture: An architectural style for smart web of everything," in *Journal of Computer Research and Development*, vol. 56, no. 1, pp. 90-102, 2019. (in Chinese with abstract in English).

[11] G.X. Yue, Y.S. Dai, X.H. Yang, "Multi-constrained trusted cooperative task migration strategy for edge computing," in *Telecommunications Science*, vol. 35, no. 11, pp. 36-50, 2019.

[12] Y.M. Miao, G.X. Wua, "Intelligent task prediction and computation offloading based on mobile-edge cloud computing, " in *Future Generation Computer Systems*, vol. 102, pp. 925-931, 2020.

[13] X. Cao, F. Wang, J. Xu, *et al*., "Joint Computation and Communication Cooperation for Energy-Efficient Mobile Edge Computing," in *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4188-4200, June 2019.