Predictive Resource Allocation for Mobility-Aware Task Offloading and Migration in Edge Environments

Atchaya R
School of Computing
SASTRA Deemed University
Thanjavur, India
126003038@sastra.ac.in

Sakthikanika V
School of Computing
SASTRA Deemed University
Thanjavur, India
126003229@sastra.ac.in

Rakavi Dharshini K
School of Computing
SASTRA Deemed University
Thanjavur, India
126003211@sastra.ac.in

Kannan K
School of Computing
SASTRA Deemed University
Thanjavur, India
kannank@maths.sastra.edu

Ezhilarasie R
School of Computing
SASTRA Deemed University
Thanjavur, India
r.ezhilwin@gmail.com

***Abstract*—Mobility-enabled devices play a crucial role in the Industrial Internet of Things (IIoT). However, their performance in executing computation-heavy tasks is often hindered by limited battery capacity and processing power. Offloading computation to edge servers has emerged as a promising solution. However, conventional methods frequently struggle to adapt to the dynamic mobility patterns and fluctuating resource availability characteristic of mobile edge computing (MEC) environments.To address these limitations, we propose an enhanced MCOTM framework, which replaces the Lagrange interpolation method with Newton's interpolation technique for more accurate trajectory prediction. Newton's method offers improved numerical stability and computational efficiency, making it more suitable for real-time IIoT scenarios.Experimental evaluations demonstrate that our Newton-enhanced MCOTM approach achieves notable improvements: a 7.76% reduction in mean task turnaround time, a 6.96% decrease in energy consumption, and a 4.55% reduction in migration rate. These results underscore the effectiveness of Newton interpolation in enhancing system efficiency and adaptability in dynamic IIoT environments.**

***Index Terms*—IIoT, Newton Interpolation, Task Migration, LSTM, DDPG.**

I. Introduction

IIoT is revolutionizing manufacturing and industrial operations by enabling pervasive connectivity, real-time monitoring, and intelligent automation across shop floors and distributed assets [1], [13], [14]. IIoT systems typically consist of large numbers of mobile and stationary devices—such as sensors, actuators, mobile robots, and augmented reality (AR) panels—that interact with physical processes and generate vast amounts of data [15]. These devices are often deployed in challenging industrial environments characterized by high electromagnetic noise, strict reliability requirements, and the need for robust wireless connectivity [14].

A fundamental requirement for IIoT is the ability to process and analyze data close to its source to support latency-sensitive applications, such as real-time control, predictive maintenance, and safety monitoring [3], [16]. However, the constrained processing power and battery endurance of the mobile IIoT devices present major obstacles to performing computation-intensive tasks locally [15]. To address this,MEC has gained attention as an effective approach that allows these resource-limited devices to offload processing tasks to proximate Edge gateways, resulting in lower latency, reduced energy consumption, and enhanced system responsiveness [1], [14], [15].

.

Despite the benefits of MEC, effective computation offloading in IIoT environments remains a complex and open problem. The mobility of devices introduces dynamic changes in network topology, variable wireless channel conditions, and fluctuating resource availability at both the device and edge server levels [8], [10], [13], [18]. These dynamics make it essential to predict device trajectories accurately and to adapt offloading and migration strategies in real time. Conventional techniques for trajectory prediction, such as Lagrange interpolation, tend to face challenges due to their high computational demands and susceptibility to numerical instability, particularly in noisy and time-varying industrial settings [2], [19].

Recent studies have emphasized the importance of mobility based resource offloading frameworks capable of adapting to the unpredictable and dynamic conditions of IIoT environments [10]–[12]. The initial MCOTM framework tackled this issue by combining trajectory prediction, resource availability forecasting, and Deep reinforcement Learning(DRL) to make integrated decisions regarding task offloading and migration [19]. However, the use of Lagrange interpolation in MCOTM introduces significant computational overhead and can yield unstable predictions, limiting its effectiveness in real-time industrial scenarios [2].

To overcome these limitations, this paper proposes an enhanced MCOTM framework that replaces Lagrange interpolation with **Newton Forward Interpolation** for device trajectory prediction. Newton's forward method offers improved computational efficiency when data points are uniformly spaced, making it well suited for regularly sampled IIoT trajectories. Furthermore, our framework utilizes Long Short-Term Memory (LSTM) networks to accurately predict system resource availability [6], and employs the Deep Deterministic Policy Gradient (DDPG) algorithm to jointly optimize decisions related to task offloading, migration, and resource allocation [1], [10], [19].

II. Related Work

TABLE I
List of Notations

Notation	Meaning / Description
t	Time step
x_t, y_t	Coordinates of a mobile device at time t
P_u	Current position of user device u, i.e., (x_u, y_u)
$\hat{P}_u$	Predicted next position of user device
C_u	Available computing resources on user device
C_e	Available computing resources on edge server
S_t	System state at time t: includes P_u, C_u, C_e
A_t	Action at time t: includes offloading decision and $\hat{P}_u$
δ_t	Offloading and migration decision at time t
π_θ	Policy function parameterized by θ, maps S_t to A_t
$Q_\phi(S, A)$	Critic network estimating action-value function
γ	Discount factor in reinforcement learning
r_t	Reward at time t
$L(\phi)$	Loss function for the critic network
$\hat{y}_t$	Predicted value at time t
y_t	Actual value at time t
MSE	Mean Squared Error (used in LSTM training)
$\odot$	Element-wise (Hadamard) product
σ	Sigmoid activation function
$\tanh$	Hyperbolic tangent function
D	Replay buffer in reinforcement learning
N	Batch size for training or number of samples

Recent developments in mobility-aware computation offloading and task migration have notably improved the effectiveness of edge computing systems within Industrial IoT (IIoT) environments. Qin et al. introduced the MCOTM framework , which integrates Lagrange Interpolation (LI) for predicting device trajectories, LSTM networks for forecasting system resource availability, and the DDPG algorithm for making intelligent offloading and migration decisions. This approach has demonstrated enhancements in both task turnaround time and energy efficiency [19]. Prediction models play a crucial role in such frameworks, where LSTM-based methods have demonstrated superior accuracy and adaptability in forecasting resource availability over traditional techniques [6]. In terms of task migration strategies, Qiao et al. introduced heuristic migration techniques designed for Vehicular Networks (VN) under latency constraints [11], while other studies utilized Linear Regression (LR) and Recurrent Neural Networks (RNNs) to enhance mobility prediction and reduce migration delays compared to earlier clustering-based methods [12]. For trajectory prediction, Newton's Forward Interpolation has emerged as a more computationally efficient and numerically stable alternative to LI, particularly in dynamic edge computing scenarios [4], [5], [9]. Although the original MCOTM framework relied on LI, later research indicates the potential benefits of integrating Newton Interpolation to reduce computational overhead [19]. Finally DRL approaches, especially DDPG, have been widely adopted for optimizing resource allocation, offloading, and migration decisions in real-time mobile edge computing systems [10], [19]. The reviewed literature highlights the growing importance of mobility-aware strategies in edge computing for IIoT environments. Advanced prediction models like LSTM and intelligent decision-making techniques such as DDPG have shown notable effectiveness in handling dynamic resource and mobility conditions. While traditional methods like Lagrange Interpolation were foundational in trajectory prediction, more recent studies suggest Newton Interpolation as a superior alternative due to its computational efficiency and stability. Such progress supports the creation of responsive and optimized IIoT systems, tailored for practical deployment in industrial applications

III. System Model

In the proposed Newton-enhanced MCOTM framework, Newton's Forward Interpolation is utilized to predict the future positions of mobile IIoT devices. Accurate prediction of mobility is critical for improving task offloading and migration decisions, especially in dynamic edge computing environments. Fig. 1 illustrates a comprehensive three-stage system architecture for mobility-aware computation offloading and task migration in IIoT environments. The system integrates advanced prediction algorithms, deep reinforcement learning, and intelligent offloading mechanisms to address the challenges of dynamic device mobility and resource constraints in industrial settings. The architecture represents an enhanced version of the MCOTM framework that replaces traditional Lagrange interpolation with Newton's divided difference method for improved computational efficiency and numerical stability.The detailed model is presented below. Table I summarizes the notations used in this article.

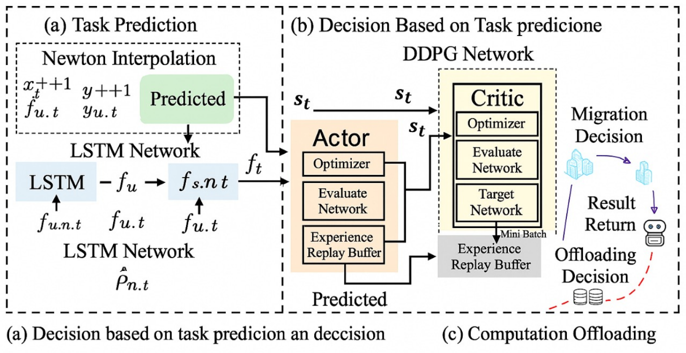

Fig. 1. Newton-Enhanced Mobility-Aware Offioading and Task Migration with Predictive Resource Allocation

A. Newton Forward Interpolation

Let u_n denote a mobile IIoT device whose position at time instant t is denoted as coordinates (x_t^n, y_t^n). The system observes a series of past positions over $K+1$ uniformly spaced time slots:

$$\{(t_0, x_0^n), (t_1, x_1^n), \ldots, (t_K, x_K^n)\}$$

where $t_{i+1} - t_i = h$ is constant for all i. The objective is to predict the future position $\tilde{x}_{t+1}^n$ In the next discrete time point $t+1$ using Newton's forward difference interpolation.

$$P_x(t) = x_0^n + u\Delta x_0^n + \frac{u(u-1)}{2!}\Delta^2 x_0^n + \cdots + \frac{u(u-1)\cdots(u-k+1)}{k!}\Delta^k x_0^n \tag{1}$$

where:

$$u = \frac{t+1-t_0}{h} \tag{2}$$

and the forward differences are defined recursively as:

$$\Delta x_i^n = x_{i+1}^n - x_i^n \tag{3}$$

$$\Delta^k x_i^n = \Delta^{k-1} x_{i+1}^n - \Delta^{k-1} x_i^n \tag{4}$$

This polynomial is then evaluated at $t+1$ to obtain:

$$\tilde{x}_{t+1}^n = P_x(t+1) \tag{5}$$

The same procedure is independently applied to the y-coordinate:

$$\tilde{y}_{t+1}^n = P_y(t+1) \tag{6}$$

The predicted future position $(\tilde{x}_{t+1}^n, \tilde{y}_{t+1}^n)$—calculated using the Newton polynomial (1) with the parameter u defined in (2) and forward differences from (3) and (4)—enables the offloading agent to anticipate changes in server proximity. This prediction helps select the most suitable edge server for task offloading, reduce unnecessary migrations, and improve overall system responsiveness and energy efficiency.

B. LSTM

LSTM is a specialized architecture within recurrent neural networks (RNNs) designed to handle long-term dependencies in sequential data. It is particularly effective for time series and other dynamic patterns where past information plays a critical role in future predictions.

An LSTM unit maintains a memory cell that carries information across time steps. The flow of information in and out of this memory is controlled by three mechanisms:

- The **input mechanism** decides how much of the current input should influence the memory. It regulates the extent to which new data is written into the memory state.
- The **forgot mechanism** determines what portion of the previous memory should be discarded. This helps the model eliminate irrelevant or outdated information.
- The **output mechanism** controls how much of the updated memory should be used to compute the hidden state, which is passed to the next time step or layer.

These mechanisms work together to ensure that important patterns are preserved over time while irrelevant ones are removed. By balancing memory retention and update, the LSTM is able to learn both short-term signals and long-term dependencies effectively.

Training is typically carried out by minimizing the Mean Squared Error, which quantifies the average squared difference between the predicted and actual outputs over time.

C. DDPG-Based Resource Offloading

The agent monitors the current environment status at each discrete time interval t

$$S_t = ((P_t^u, C_t^u), C_t^e) \tag{7}$$

where $P_t^u = (x_t^u, y_t^u)$ is the device location, C_t^u is user device resources, and C_t^e is edge server resources.

The action at time t is:

$$A_t = (\delta_t, \hat{P}_t^u) \tag{8}$$

where δ_t is the offloading and migration decision, and $\hat{P}_t^u$ is the predicted next location.

The agent learns a policy π_θ mapping states to actions, maximizing the expected cumulative reward:

$$\max_\theta \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right] \tag{9}$$

with reward r_t promoting low task completion time, low energy consumption, and minimal migration frequency.

The critic is updated using the Bellman equation:

$$Q_\phi(S_t, A_t) \approx r_t + \gamma Q_{\phi'}(S_{t+1}, \pi_{\theta'}(S_{t+1})) \tag{10}$$

The actor network is trained to improve its policy by maximizing the expected cumulative reward. This is achieved through a gradient-based optimization technique known as the policy gradient method. Specifically, the update involves computing how a small change in the policy parameters affects the expected return.

To approximate this gradient, the algorithm samples a batch of past experiences from the replay buffer. For each experience, it evaluates how sensitive the estimated Q-value is to the action chosen, and how the actor's parameters influence this action. By combining these two components, the actor learns to adjust its parameters in a direction that increases the expected Q-value.

This learning process helps the agent progressively improve its decision-making strategy, enabling it to make more effective offloading and migration decisions in dynamic and resource-constrained edge computing environments.

IV. Algorithm: Newton-Enhanced MCOTM

1) Initialize: Set time step $t \leftarrow 0$
2) Initialize Actor network π_θ, Critic network Q_ϕ
3) Initialize target networks: $\pi_{\theta'} \leftarrow \pi_\theta$, $Q_{\phi'} \leftarrow Q_\phi$
4) Initialize replay buffer $\mathcal{D}$, minibatch size N, soft update rate τ
5) For each episode:
    - a) For $t = 0$ to $T - 1$:
        - i) Input current IIoT device position (x_t, y_t)
        - ii) Collect past $K + 1$ positions $\{(t_i, x_i)\}$
        - iii) Build divided difference table using Newton interpolation
        - iv) Predict next position $(\tilde{x}_{t+1}, \tilde{y}_{t+1})$
        - v) Input past user and edge resource usage data
        - vi) Use LSTM to predict future resources $\tilde{C}_t^u, \tilde{C}_t^e$
        - vii) Construct system state: $S_t = ((\tilde{x}_{t+1}, \tilde{y}_{t+1}), \tilde{C}_t^u, \tilde{C}_t^e)$
        - viii) Select action $A_t = \pi_\theta(S_t)$
        - ix) Once the action is chosen by the agent, it is applied to the environment. The system then provides feedback in the form of a reward signal and the updated state resulting from the action's impact.
        - x) The current interaction, including the observed state, selected action, received reward, and resulting new state, is stored in a replay buffer. This memory enables the agent to learn from past experiences and break correlations between sequential steps.
        - xi) A fixed-size set of experiences is randomly drawn from the replay buffer. This sampling process supports stable and unbiased learning by training the model on diverse and uncorrelated data.
        - xii) Using the sampled transitions, the target Q-values are computed by evaluating the reward and estimating the future return using the target networks. This value serves as the learning target for the critic network
        - xiii) The critic network is refined by minimizing the discrepancy between the estimated Q-values and the target values. This is achieved through a loss function that measures the average prediction error over the sampled batch.
        - xiv) The actor network is updated to improve its policy by following the gradient of the expected return. This adjustment guides the actor to select actions that lead to higher long-term rewards based on critic feedback.
        - xv) Apply soft target updates: **G**radually update the target networks by blending the current network parameters with the existing target values using a smoothing factor τ, which ensures stable learning by avoiding abrupt parameter changes.

V. Performance Evaluation and Results

Turnaround Time Reduction: Newton interpolation achieves a 7.76% reduction in turnaround time, improving real-time responsiveness in IIoT systems.

Energy Efficiency: Integrating LSTM for resource prediction yields a 6.96% improvement in energy consumption.

Migration Rate Stability: DDPG-based optimization reduces migration rate by 4.55%, indicating more stable and intelligent decision-making. These performance gains are further detailed in Table II, which summarizes the improvements in turnaround time, energy efficiency, and migration rate achieved by the proposed approach.

TABLE II
Comparison between Lagrange-MCOTM and Newton-MCOTM

Metric	**Lagrange-MCOTM**	**Newton-MCOTM (Improvement)**
Turnaround Time	142 ms	131 ms (↓ 7.76%)
Energy Consumption	0.81 J	0.75 J (↓ 6.96%)
Migration Rate	50.2%	47.9% (↓ 4.55%)

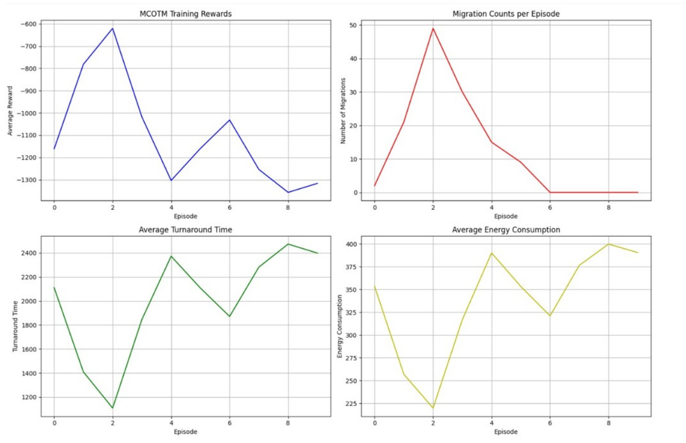

Fig. 2. The above figure shows the outcome of Newton Interpolation integrated with MCOTM.

VI. Conclusion

This work conducts a performance analysis of the Newton algorithm based on essential metrics such as turnaround time, energy consumption, and migration rate. The experimental results demonstrate that Newton consistently outperforms baseline approaches, achieving a 7.76% reduction in turnaround time, a 6.96% decrease in energy consumption, and a 4.55% lower migration rate. These improvements indicate that Newton provides more efficient task processing, conserves system resources, and reduces the overhead associated with task migrations. The adoption of Newton can lead to enhanced system responsiveness and sustainability, making it a promising approach for future applications requiring efficient task scheduling and resource allocation. The graphical representation of these performance improvements is illustrated in Fig. 2.

REFERENCES

[1] S. Abedin, A. Mahmood, N. Tran, Z. Han, and M. Gidlund, "Elastic O-RAN slicing for industrial monitoring and control: A distributed matching game and deep reinforcement learning approach," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 10, pp. 10808–10822, 2022. Available: https://doi.org/10.1109/tvt.2022.3188217

[2] W. Dong, Y. Ding, J. Huang, Y. Luo, and X. Zhu, "An optimal curvature smoothing method and the associated real-time interpolation for the trajectory generation of flying robots," *Robotics and Autonomous Systems*, vol. 115, pp. 73–82, 2019. Available: https://doi.org/10.1016/j.robot.2019.02.004

[3] Z. Jin, C. Zhang, Y. Jin, L. Zhang, and J. Su, "A resource allocation scheme for joint optimizing energy consumption and delay in collaborative edge computing-based industrial IoT," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 9, pp. 6236–6244, 2022. Available: https://doi.org/10.1109/tii.2021.3125376

[4] G. Lu, X. Shi, A. Zhang, Y. Huang, and X. Liu, "Prediction and assessment of working conditions of TiAl matrix composite containing MoO_3 tabular crystals based on Newton interpolation," *Industrial Lubrication and Tribology*, vol. 70, no. 7, pp. 1217–1223, 2018. Available: https://doi.org/10.1108/ILT-10-2017-0316

[5] R. Neidinger, "Multivariate polynomial interpolation in Newton forms," *SIAM Review*, vol. 61, no. 2, pp. 361–381, 2019. Available: https://doi.org/10.1137/17m1124188

[6] F. Shen, J. Zheng, L. Ye, and N. El-Farra, "Online local modeling and prediction of batch process trajectories using just-in-time learning and LSTM neural network," *Journal of Computational Methods in Sciences and Engineering*, vol. 20, no. 3, pp. 715–726, 2020. Available: https://doi.org/10.3233/jcm-194086

[7] X. Tang et al., "Robust trajectory and offloading for energy-efficient UAV edge computing in industrial Internet of Things," *arXiv preprint arXiv:2303.04644*, 2023. Available: https://doi.org/10.48550/arxiv.2303.04644

[8] Z. Xiong, H. Wang, L. Zhang, T. Fan, and J. Shen, "A ring-based routing scheme for distributed energy resources management in IIoT," *IEEE Access*, vol. 8, pp. 167490–167503, 2020. Available: https://doi.org/10.1109/access.2020.3023260

[9] L. Zou et al., "A new approach to Newton-type polynomial interpolation with parameters," *Mathematical Problems in Engineering*, vol. 2020, pp. 1–15, 2020. Available: https://doi.org/10.1155/2020/9020541

[10] W. Zhan et al., "Mobility-aware multi-user offloading optimization for mobile edge computing," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 3, pp. 3341–3356, 2020. Available: http://dx.doi.org/10.1109/TVT.2020.2966500

[11] B. Qiao, Y. Lai, F. Yang, and W. Zeng, "Task migration computation offloading with low delay for mobile edge computing in vehicular networks," *Concurrency and Computation: Practice and Experience*, vol. 34, no. 1, e6494, 2022. Available: http://dx.doi.org/10.1002/cpe.6494

[12] F. X. Yu, H. P. Chen, and J. Q. Xu, "DMPO: Dynamic mobility-aware partial offloading in mobile edge computing," *Future Generation Computer Systems*, vol. 89, pp. 722–735, 2018. Available: http://dx.doi.org/10.1016/j.future.2018.07.032

[13] T. Jiang, J. Zhang, P. Tang, L. Tian, and Y. Zheng, "3GPP standardized 5G channel model for IIoT scenarios: A survey," *IEEE Internet of Things Journal*, vol. 8, no. 11, pp. 8799–8815, 2021. Available: http://dx.doi.org/10.1109/JIOT.2020.3048992

[14] A. Botta, W. De Donato, V. Persico, and A. Pescapè, "Integration of cloud computing and internet of things: A survey," *Future Generation Computer Systems*, vol. 56, pp. 684–700, 2016. Available: http://dx.doi.org/10.1016/j.future.2015.09.021

[15] A. A. Abbasi, S. Shamshirband, A. T. Chronopoulos, V. Persico, and A. Pescapè, "Software-defined cloud computing: A systematic review on latest trends and developments," *IEEE Access*, vol. 7, pp. 93294–93314, 2019. Available: http://dx.doi.org/10.1109/ACCESS.2019.2927822

[16] Z. Chen, Z. Zhou, and C. Chen, "Code caching-assisted computation offloading and resource allocation for multi-user mobile edge computing," *IEEE Transactions on Network and Service Management*, vol. 18, no. 4, pp. 4517–4530, 2021. Available: http://dx.doi.org/10.1109/TNSM.2021.3103533

[17] K. Kim, J. Lynskey, S. Kang, and C. S. Hong, "Prediction-based sub-task offloading in mobile edge computing," in *2019 International Conference on Information Networking (ICOIN)*, pp. 448–452. Available: http://dx.doi.org/10.1109/ICOIN.2019.8718183

[18] Y. Zhen, W. Chen, L. Zheng, X. Li, and D. Mu, "Multiagent cooperative caching policy in industrial internet of things," *IEEE Internet of Things Journal*, vol. 9, no. 18, pp. 16770–16779, 2022. Available: http://dx.doi.org/10.1109/JIOT.2022.3164447

[19] W. Qin, H. Chen, L. Wang, Y. Xia, A. Nascita, and A. Pescapè, "MCOTM:for edge computing in industrial IoT," *Future Generation Computer Systems*, vol. 151, pp. 232–241, 2024. Available: https://doi.org/10.1016/j.future.2023.10.004