Computation for Reinforcement Learning at the Mobile Edge Network

Zixu Zhao*

School of Electrical and Information engineering, Northeast Agriculture University, Harbin, Heilongjiang province, China, 150006

ABSTRACT: Wireless Mobile Edge Computing (MEC) is one of several promising models emerging in recent years. A wireless powered MEC network is researched in this paper. The proposed online offload framework based on deep reinforcement learning (DROO) draws lessons from the previous offload experience. As such, it fulfills the need to solve complex problems like MIP. The computational complexity will not surge with the increase of network size. DROO decomposes the original optimization problem into a secondary problem from design to unloading decision and a resource allocation sub-problem. It is suitable for the space without interruption, and there is no need to discretize the channel gain, so as to avoid the disaster caused by the problem of dimension. By studying the simulation results, the DROO algorithm is found that it can achieve almost perfect performance by designing and calculating the method at the current stage, but it also needs to reduce CPU execution latency by at least one order of magnitude, so as to ensure that the real-time system optimization is fading. It is really feasible to use MEC networks when wirelessly powered in the environment.

1. INTRODUCTION

In general, modern devices are limited in two areas: power supply and computer source. By studying the latest progress of wireless power transmission technology, it can be known that the battery of wireless devices can be continuously charged in air [1]. At the same time, the latest development of Mobile Edge Computing (MEC) technology can effectively improve the computing power of equipment [2][3]. Using MEC and WDS, computing intensive tasks can be transferred to nearby edge servers, thus reducing computing delay and energy consumption [4][5].

Wireless-powered MEC benefits when combined in the above two ways, so it is expected to solve the two basic performance limitations of Internet of things devices [6][7]. In the above text, a wireless power supply MEC system is mainly considered. Locations related to access points (APs) for transmitting RF (radio frequency) strength to the WDs and taking over computing load from the WDs. WDS follows the operation of a binary unloading strategy [8]. When performing a general algorithm such as local computing, offloading can be performed first, and then remote computing can be performed through the MEC server. It can be considered that the settings of the system correspond to many typical related networks of the Internet of Things. Each wireless sensor used for energy collection completes relevant sensing tasks with the help of MEC server.

A lot of work will be combined to model the typical problem of solving problems and rationally allocating resources through calculation in the MEC network. In [7], the author introduced a method that can perform coordinate descent (CD), which can be used to find along a variable dimension. In [9], the main contribution of the authors is to develop an analogous and step by step search way mainly used in multi-connector MEC networks. Another used heuristic-type approach is to approximate binary constraints with slack, such as in a way that turns integer type variables into continuous from 0 to 1, or by methods such as quadratic constraints [10][11]. Even so, there is no way to guarantee the quality of a certain solution to the complexity reduction heuristic in a particular aspect.

The second aspect is mainly based on search methods and relaxation methods, which require a lot of iterations to reach a satisfactory optimal value. This optimal quality is for local data, but this is not suitable for fast fading channels. But it has a great impact on the work, such as the advantages and action spaces when dealing with reinforcement learning problems with relatively large state spaces when it works and learns [12-14].

In the text, the author considers a kind of wireless network such as MEC with a binary offload strategy, making each computing mission of the wireless device (WDs) either performed locally or unloaded to a MEC server. The goal is to obtain an online type of algorithms to make task allocation and radio energy allocation most suitable for time-varying wireless channel conditions. In the channel coordination, there will be problems such as more complex combination optimization, which is also difficult to achieve by traditional numerical optimization methods. To a certain extent, to completely solve this problem, the author mainly proposes a heterogeneous

*Corresponding author. Email: zzx2128351803@163.com online offloading framework based on deep reinforcement learning. The main purpose of this part is to use heterogeneous deep neural networks as a scalable solution. The main approach is to learn decisions about binary unloading from experience, so that the need for solving combinatorial optimization problems can be eliminated, which can reduce the computational complexity to some extent. This effect is more obvious in large networks.

Through the research of this paper, it is found that if some sub problems of resource allocation can be solved, the resource allocation problem can be solved to a great extent in the subsequent process, so that the quality of integer decision variables can be determined in the evaluation process. In this case, the proposed DROO framework is very suitable, which can simplify various steps as much as possible. The research of this paper enriches the relevant contents of wireless power transmission technology, and makes relevant thinking on solving the basic performance limitations of Internet of things devices.

2. SYSTEM MODEL

Each WD follows a binary unloading strategy. The main goal is to jointly and inversely optimize the decision of task allocation for each WD, WPT, and the allocation of transmission time between each task and time allocation among many WDs according to the time-varying wireless channel. Based on this situation, an online learning framework is proposed here, that is, DROO, which can be weighted to maximize all WD calculation rates, and will also get the number of bits processed per unit time. The author considers a wireless powered MEC system composed of AP and WD, denoted by the set $N=\{1,2,\dots\}$, where we design antennas for each device.

In practice, this may correspond to a low-power Internet of things system. Mainly because the AP has a relatively stable power supply, it can broadcast radio frequency energy to the WD. Each WD will have a rechargeable battery, so that it can store various energies collected to power the device. Assume that AP has more computing power than WDs, so WDs can transfer its computing tasks to AP. It can be assumed that the communication process of WPT and computational transmission is performed in the same frequency band. Therefore, a circuit can be implemented, such as time division multiplexing, which is TDD, on each device, so as to reduce the mutual interference between WPT and communication in the process of transmitting signals.

For the division of system time, it is mainly divided into a continuous time frame of the same time T, and the time frame set at design time is less than the time of channel coherence. At each marking time, the energy obtained by WD from the speed of communication between APs is related to the gain of the wireless channel. Using it to represent the gain of the wireless channel between the AP and WD in the marked time frame. At the beginning of a time period, WPT uses a certain time, $a\in[0,1]$, in which the AP's associated energy is available for the WD to acquire,where $u\in[0,1]$ is the energy harvesting efficiency and P is the transmission power of AP[1]. Using the collected energy, each WD needs to complete a priority task before the end.

Dividing WI weights equally into WD. The larger the weight w_i, the greater the calculation rate assigned to the i th WD.

2.1. The computing mode of the local machine

WD in local computing mode can obtain energy and calculate its tasks at the same time [6]. Set f_i as the calculation speed of the processor, $0\le t_i\le \mathrm{T}$ as the calculation time. φ>0 indicates that the time required to process each task when processing task data. At the same time, the calculated WD energy consumption is restricted by $k_i f_i t_i \le E_i$, where k_i is the calculated energy level factor. It can be seen to deal with the maximum number of data in T under force constraints, WD needs to consume the collected energy and calculate it over the whole time period, that is, $t_i^*=T$ so $f_i^*=\left(\frac{E_i}{k_i T}\right)^{\frac{1}{3}}$. Therefore, the local computing rate (in bits per second) is

$$r_{L,i}^*(a)=\frac{f_i^* t_i^*}{\phi T}=\eta_1\left(\frac{h_i}{k_i}\right)^{\frac{1}{3}}a^{\frac{1}{3}} \qquad (1)$$

2.2. Boundary calculation model

Because of the limitations of TDD, if the WD is in offload mode, then the task can only be offloaded to the AP after the energy harvesting is over. It denotes $\tau_i T$ as the offloading time of the i-th WD, $\tau_i\in[0,1]$. The authors assume that the calculation speed and transmission power of AP are more than three orders of magnitude larger than that of WD with size and energy constraints. In addition, the computational feedback downloaded to WD is much shorter than that of the edge server. Therefore, the author safely ignores the time spent by AP on task calculation and download, so that there is only WPT and task offload per time frame, that is,

$$\sum_{i=1}^{N}\tau_i + a \le 1 \qquad (2)$$

3. EXPRESSION OF THE PROBLEM

Among some common system parameter, it is assumed only wireless channels are included gain. $h=\{h_i \mid i\in N\}$ is during a certain period of time. When the time changes, it also changes with it while others are fixed parameters. Therefore, the weighted sum calculation rate of wireless power supply MEC network in the marked time frame is:

$$Q(h,x,\tau,a)=\sum_{i=1}^{N} w_i((1-x_i)r^{*}_{L,i}(a)+x_i r^{*}_{O,i}(a,\tau_i)) \quad (3)$$

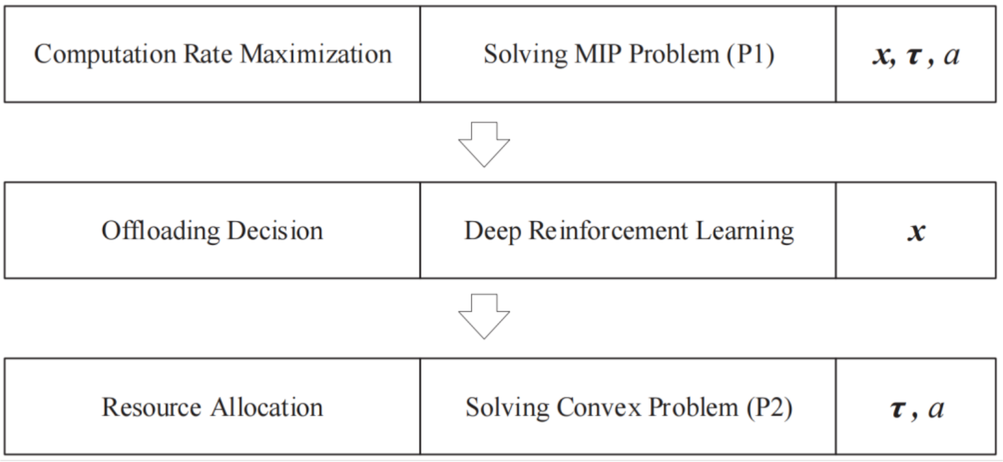
Computation Rate Maximization
Solving MIP Problem (P1)
x, τ, a
Offloading Decision
Deep Reinforcement Learning
x
Resource Allocation
Solving Convex Problem (P2)
τ, a

Figure 1 Solutions for bipolar structures

If the implementation of the channel can be done every time frame, the author aimed to maximize the weighted sum computation rate:

$$\text{(P1):}\quad Q^{*}(h)=\underset{x,\tau,a}{\max imize}\, Q(h,x,\tau,a) \quad (4)$$

$$\text{subject to } \sum_{i=1}^{N}\tau_i + a \le 1,$$
$$a \le 0, \tau_i \ge 0, \forall_i \in N,$$
$$x_i \in \{0,1\}$$

For problem 1, it is a very intractable problem when doing multiple integer programming. Although this problem is difficult to deal with, once given, this kind of problem can be reduced to a certain kind of convex problem.

$$\text{(P2)}:\quad Q^{*}(h,x)=\underset{\tau,a}{\max imize}\ \mathrm{Q(h,x,}\tau\mathrm{,a)} \quad (5)$$

$$\textit{subject to } \sum_{i=1}^{N}\tau_i + a \le 1,$$
$$a \ge 0, \tau_i \ge 0, \forall_i \in N$$

Therefore, the first problem can be decomposed that the unloading decision and resource allocation in P2.

Settlement Decisions: Possible settlement decisions need to be examined to obtain the best or most satisfactory settlement decision x. For example, metaheuristic search algorithms [7] have been proposed to improve settlement decisions. However, as the search space grows exponentially, the algorithm takes longer to compile.

In the process of resource allocation, convex problems about optimal time allocation can be effectively solved. For example, when allocating time to O(N) complexity, we can constrain the dual variables that are related to each other and use a Dimensional dichotomy to search for related content.

The main question of (P1) lies about the unloading decision. The normal way of the optimization algorithm needs to adjust the unloading decision to the optimal, which is not feasible for time system majorization in fast decreasing channels. To solve this problem, a new online unloading algorithm based on deep reinforcement learning (DROO) is proposed.

It is also important to note before the end of this section that the DNN method, a method of applying deep reinforcement learning to dynamic wireless applications, is based on supervised learning [15][16]. Generally speaking, in the process of deep reinforcement learning, it is not necessary to mark samples as DNN. Another outstanding advantage is that it is faster to analyze changes in users' channel distribution. For example, when a WDS has an obvious position change or sudden shutdown, DNN needs to be fully retrained. The unloading decision can be updated through more in-depth learning strategies according to the changes of channel distribution without manual intervention.

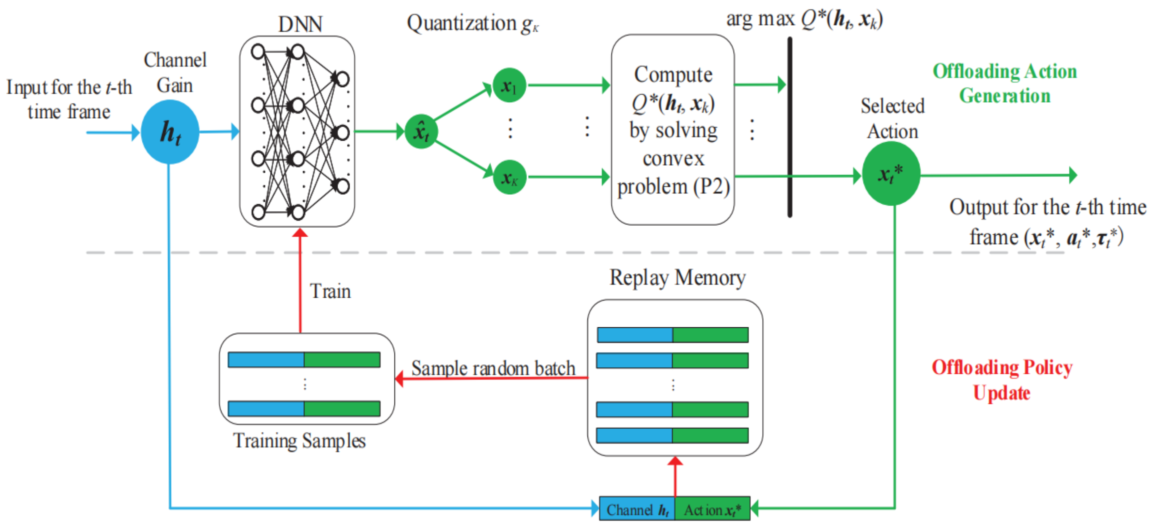

Figure 2 The schematics of the proposed DROO algorithm

4. DROO CALCULATION METHOD

The author hopes to design a strategy function that can be uninstalled. If the implementation of the channel is displayed at the beginning of each time frame, then this strategy function can generate the optimal unloading action at a very fast speed. This method can be considered as:

$$\pi\text{:}\ \ h \mapsto x^* \qquad (6)$$

It can be explored what kind of function this strategy has and learn from past experience.

The details are shown in figure 2. It mainly includes two alternating processes, namely, action generation when uninstalling and the related new ways, generating offload action when using DNN. The most prominent point is the embedded data θ, e.g., connecting hidden neurons. In the t-th time frame, DNN transmits ht as input, and outputs the relatively simple unloading action $\hat{X}_t$ (each part operates between 0 and 1 to make it a continuous type) take the existing uninstall policy as a reference π_{θ_t}, parameterized by θ_t. The re-relaxing action reduces to K binary aspect unloading actions. One of the most effective actions is to observe (P2) the rate at which computation can be performed. The corresponding $\{x_t^*, a_t^*, \tau_t^*\}$ is to treat the output as a solution to ht, which guarantees that it meets the requirements shown in (4b)-(4d). The network is using the uninstall action x_t^*, receives a reward $Q^*(h_t, x_t^*)$, and Put the reappearing state-action pair into memory for use.

A batch of sample DNNs can then be drawn from memory at a time frame t for policy updates, and then updated with the relevant data from θ_t to θ_{t+1}. The $\pi_{\theta_{t+1}}$ (uninstallation measures) is used to generate an offloading decision x_{t+1}^* according to the new channel h_{t+1} observed. After that, as new channel implementations are observed, this iteration will be repeated. The policy π_{θ_t} of the DNN is gradually improved.

Using the uninstall solution obtained to update the uninstall policy of DNN. At the t-th time frame, adding a new training data sample (h_t, x_t^*) to the memory. When the memory is full, using the new data sample.

If the experience playback technique is used, then the stored data samples for subsequent processing of the DNN can be studied. In the t-th time frame, the author randomly selects a batch of training data samples $\{(h_\tau, x_\tau^*) \mid \tau \in \Gamma_t\}$ from the memory, characterized by a set of time indices Γ_t. Adam algorithm is applied to update the parameters θ_t of the DNN to reduce the average cross entropy loss, as following:

$$L(\theta_t) = -\frac{1}{|\Gamma_t|}\sum_{\tau\in\Gamma_t}((x_\tau^*)^T \log f_{\theta_t}(h_\tau) + (1-x_\tau^*)^T \log(1-f_{\theta_t}(h_\tau)))$$

where $|\Gamma_t|$ denotes the size of Γ_t, the superscripts represent the specific operators required to perform the transpose, and the log function is a form of describing the element-wise logarithm of a vector. To simplify the process, the update sequence of the Adam algorithm will not be repeated. In practical use, the author trains the DNN every delta time frame after obtaining many new data samples. This technique has several major advantages. Firstly, for the entire data sample, the complexity is greatly reduced when doing batch updates. Secondly, using more of the past data can reduce the variance value during the update period. Thirdly, random sampling speeds up the convergence by reducing the correlation of the training data. The DNN learns from actions with the best state, and the longer it takes, the better the offloading decision emerges. Due to limited memory space, DNNs can only learn from the latest data samples. This mechanism of reinforcement learning in a stationary system optimizes the offloading decision and stops it when it becomes convergent.

5. CONCLUSION

This paper mainly studies an calculation method of online unloading, DROO, which can calculate the ratio when weighted in a wirelessly powered MEC network processing when performing binary calculations. This method is learned from previous relevant unloading experience, and then improved through reinforcement learning to obtain unloading actions based on DNN generation.

Generally speaking, when the algorithm is used, it needs to ensure that it can converge quickly, so the author also invented a method for order-preserving quantization and parameter setting according to the actual situation. This can make the results more accurate. Generally speaking, compared with traditional methods, the DROO algorithm designed in this paper does not have related problems such as a related scheme for hard mixed integers. The author hopes that the proposed framework can also solve some other common problems, such as the MIP problem, which mainly refers to the need for some continuous resources to make decisions when coupled integers in wireless communication and networks need to be determined. For example, in the allocation process, the most common problems are the mode selection of D2D during communication, the processing problems of the system when the base station is associated, and the problems encountered by the wireless sensor network in the process of continuous routing and cache placement. By reading the article, it is found that if some resource allocation's sub-problems can be solved to a large extent in the subsequent process, so that the integer decision variables' quality can be determined during the evaluation. Then, in this case, the DROO framework proposed is very suitable, which can simplify various steps as much as possible.

REFERENCES

1. Bi, S. , Ho, C. , & Zhang, R. . Wireless powered communication: opportunities and challenges. Communications Magazine, IEEE, 53(4) (2015) pp. 117-125.
2. Chiang, M. , & Tao, Z. . Fog and iot: an overview of research opportunities. IEEE Internet of Things Journal, 3(6) (2017) pp. 854-864.
3. Mao, Y. , Zhang, J. , & Letaief, K. B. . Dynamic computation offloading for mobile-edge computing with energy harvesting devices. IEEE J. Sel.Areas Commun., 34(12) (2016) pp.3590-3605,
4. You, C. , Huang, K. , Chae, H. , & Kim, B. H. . "Energy-effificient resource allocation for mobile-edge computation offloading," IEEE Trans. Wireless Commun., 16(3) (2017) pp. 1397–1411.
5. Chen, X. , Jiao, L. , Li, W. , & Fu, X. . Efficient multi-user computation offloading for mobile-edge cloud computing. IEEE/ACM Transactions on Networking, 24(5) (2016) pp.2795-2808.
6. Wang, F. , Xu, J. , Wang, X. , & Cui, S. ."Joint offloading and computing optimization in wireless powered mobile-edge computing systems," IEEE Trans. Wireless 17(3) (2018) pp.1784– 1797.
7. S.Bi, Y.J.A. Zhang, "Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading," IEEE Trans. Wireless Commun.,17(6), 2018, pp.4177–4190.
8. Y. Mao, C.You, J. Zhang, K. Huang, K. B. Letaief, "A survey on mobile edge computing: The communication perspective," IEEE Commun. Surveys Tuts., 19(4) (2017) pp.2322–2358.
9. T. X. Tran and D. Pompili, "Joint Task Offloading and Resource Allocation for Multi-Server Mobile-Edge Computing Networks," IEEE Transactions on Vehicular Technology, 68(1) (2019) pp. 856-868.
10. Guo, S. , Xiao, B. , Yang, Y. , & Yang, Y. Energy-efficient dynamic offloading and resource scheduling in mobile cloud computing. IEEE INFOCOM 2016 - IEEE Conference on Computer Communications. IEEE. pp. 1–9. 2016.
11. Dinh, T. Q. , Tang, J. , La, Q. D. , & Quek, T. . Offloading in mobile edge computing: task allocation and computational frequency scaling. IEEE Transactions on Communications, 65(8) (2017) pp. 3571-3584.
12. Volodymyr, M. , Koray, K. , David, S. , Rusu, A. A. , Joel, V. , & Bellemare, M. G. , et al. Human-level control through deep reinforcement learning. Nature, 518(7540), (2015). p. 529.
13. Dulac-Arnold G , Evans R , Hasselt H V , et al. Deep Reinforcement Learning in Large Discrete Action Spaces. Computer Science, 2015.
14. Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," nature, 521(7553) (2015) p. 436
15. H.Sun, X.Chen, Q.Shi, M.Hong, X.Fu, N.D. Sidiropoulos, "Learning to optimize: Training deep neural networks for wireless resource management," in Proc. IEEE SPAWC, Jul. 2017, pp. 1–6.
16. Sun, Haoran, Chen, Xiangyi, Shi, & Qingjiang, et al. Learning to optimize: training deep neural networks for interference management. IEEE Transactions on Signal Processing: A publication of the IEEE Signal Processing Society, 66(20), (2018) pp. 5438-5453.
17. T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, et al, "Continuous control with deep reinforcement learning," in Proc. ICLR, 2016.
    https://doi.org/10.48550/arXiv.1509.02971