本周主要阅读了 Liang Huang 的 Deep Reinforcement Learning for Online Computation Offlfloading in Wireless Powered Mobile-Edge Computing Networks ，这篇文章主要考虑一种采用二进制卸载策略的无线供电 MEC 网络，使得无线设备的每个计算任务要么在本地执行，要么完全卸载到 MEC 服务器。文章主要通过获得一个在线算法，最优地适应任务卸载决策和无线资源分配到时变的无线信道条件。文章中提出了一个基于深度强化学习的在线卸载(DROO)框架，该框架实现了一个深度神经网络，从经验中学习二进制卸载决策。它消除了求解组合优化问题的需要，从而大大降低了计算复杂度，为了进一步降低复杂度，提出了一种自适应的算法，自动调整 DROO 算法的参数。数值结果表明，与现有的优化方法相比，该算法可以获得接近最优的性能，并将计算时间显著减少一个数量级以上。例如，在 30 个用户的网络中，DROO 的 CPU 执行延迟不到 0.1 秒，即使在快速衰落的环境中，也可以实现实时优化卸载。

接下来考虑由一个边缘服务器和 N 个无线设备组成的 MEC 网络。
符号表示：

Notation	Description
N	The number of WDs
T	The length of a time frame
i	Index of the i-th WD
h_i	The wireless channel gain between the i-th WD and the AP
a	The fraction of time that the AP broadcasts RF energy for the WDs to harvest
E_i	The amount of energy harvested by the i-th WD
P	The AP transmit power when broadcasts RF energy
μ	The energy harvesting efficiency
w_i	The weight assigned to the i-th WD
x_i	An offloading indicator for the i-th WD
f_i	The processor's computing speed of the i-th WD
ϕ	The number of cycles needed to process one bit of task data
t_i	The computation time of the i-th WD
k_i	The computation energy efficiency coefficient
τ_i	The fraction of time allocated to the i-th WD for task offloading
B	The communication bandwidth
N_0	The receiver noise power
$\mathbf{h}$	The vector representation of wireless channel gains $\{h_i \| i \in \mathcal{N}\}$
$\mathbf{x}$	The vector representation of offloading indicators $\{x_i \| i \in \mathcal{N}\}$
$\boldsymbol{\tau}$	The vector representation of $\{\tau_i \| i \in \mathcal{N}\}$
$Q(\cdot)$	The weighted sum computation rate function
π	Offloading policy function
θ	The parameters of the DNN
$\hat{\mathbf{x}}_t$	Relaxed computation offloading action
K	The number of quantized binary offloading actions
g_K	The quantization function
$L(\cdot)$	The training loss function of the DNN
δ	The training interval of the DNN
Δ	The updating interval for K

文中先将计算资源分配问题表示为

$$\min_{\mathcal{F}} \max_{n\in\mathcal{N}_k} T_n^{Comm} + \frac{\gamma_n}{f_{nk}}$$

然后通过 KKT 条件进行求解，进行了拉格朗日求导获得了问题的最优解为

$$f_{nk} = \frac{\gamma_n}{Z - T_n^{Comm}}$$

。然后进行数据的训练，设计一个策略，可以有效地从每个系统状态中产生一个卸载的操作，来最小化期望。其中为了最小化总延迟可以得到优化问题 P1 表示为：

（P1）： $$min(s_t, a_t)$$

约束条件： $$f_{nk} > f_n, \forall n \in \mathcal{N}$$

$$a_t \in \{0,1\},$$

一旦卸载方案给出，就可以求解资源计算分配问题

$$\min_{\mathcal{F}} \max_{n\in\mathcal{N}_k} T_n^{Comm} + \frac{\gamma_n}{f_{nk}}$$

就可以得到问题 P2，使得奖励最大化：

（P2）： $$maxQ(s_t, a_t)$$

受限于：

$$\sum_{n\in\mathcal{N}_k} f_{nk} \le f_k$$

$$f_{nk} > 0, \forall n \in \mathcal{N}_k$$

。

算法流程图：

Algorithm 1: An online DROO algorithm to solve the offloading decision problem.

input : Wireless channel gain $\mathbf{h}_t$ at each time frame t, the number of quantized actions K

output: Offloading action $\mathbf{x}_t^*$, and the corresponding optimal resource allocation for each time frame t;

Initialize the DNN with random parameters θ_1 and empty memory;

Set iteration number M and the training interval δ;

for $t = 1, 2, \ldots, M$ **do**

  Generate a relaxed offloading action $\hat{\mathbf{x}}_t = f_{\theta_t}(\mathbf{h}_t)$;

  Quantize $\hat{\mathbf{x}}_t$ into K binary actions $\{\mathbf{x}_k\} = g_K(\hat{\mathbf{x}}_t)$;

  Compute $Q^*(\mathbf{h}_t, \mathbf{x}_k)$ for all $\{\mathbf{x}_k\}$ by solving (P2);

  Select the best action $\mathbf{x}_t^* = \arg\max_{\{\mathbf{x}_k\}} Q^*(\mathbf{h}_t, \mathbf{x}_k)$;

  Update the memory by adding $(\mathbf{h}_t, \mathbf{x}_t^*)$;

  if $t \bmod \delta = 0$ **then**

    Uniformly sample a batch of data set $\{(\mathbf{h}_\tau, \mathbf{x}_\tau^*) \mid \tau \in \mathcal{T}_t\}$ from the memory;

    Train the DNN with $\{(\mathbf{h}_\tau, \mathbf{x}_\tau^*) \mid \tau \in \mathcal{T}_t\}$ and update θ_t using the Adam algorithm;

  end

end

分析：获得的最佳卸载策略将会被用于更新 DNN。在 t 个时间帧时，一个新的训练数据样本(st,at)被添加至内存，当内存满的时候将先前样本用于训练 DNN 并用新的样本替换它们。采用 Adam 算法来对参数进行更新。因此随着时间的推移，卸载决策将会越来越准确，并不断地改进它所产生的卸载决策。