Deep Reinforcement Learning for IoT Network Dynamic Clustering in Edge Computing

Qingzhi Liu[*], Long Cheng[†], Tanir Ozcelebi[*], John Murphy[†], Johan Lukkien[*]
[*]MCS, Eindhoven University of Technology, The Netherlands
[†]CS, University College Dublin, Ireland
Email: q.liu.1@tue.nl, long.cheng@ucd.ie, t.ozcelebi@tue.nl, j.murphy@ucd.ie, j.j.lukkien@tue.nl

***Abstract*—How to process the big data generated in large IoT networks is still challenging current techniques. To date, a lot of network clustering approaches have been proposed to improve the performance of data aggregation in IoT. However, most of them focus on partitioning networks with static topologies, and thus they are not optimal on handling the case with moving objects in the networks. Moreover, as the best of knowledge, none of them has ever considered the performance of following computing in edge servers. To improve these problems, in this work, we propose a highly efficient IoT network dynamic clustering solution in edge computing using deep reinforcement learning (DRL). Our approach can both fulfill the data communication requirements from IoT networks and load-balancing requirements from edge servers, and thus provide a great opportunity for future high performance IoT data analytics. We implement our approach by Deep Q-learning Network (DQN) model, and our preliminary experimental results show that the DQN solution can achieve higher score in cluster partitioning compared with the current static benchmark solution.**

I. Introduction

Edge computing is an efficient solution to process large amount of data aggregated from Internet of Things (IoT). The traditional aggregated data sets in IoT are of small size, such as sensing data. Therefore, a non-optimized data aggregation solution will not obviously affect the performance of the edge computing. Because the edge servers can re-allocate the data to multiple servers through wired network, which can easily achieve Giga-byte communication speed. However, with the fast development of IoT in recent years, the property of data flow in IoT changes. Firstly, data in IoT becomes increasingly larger, such as video, images, etc. Secondly, data comes from not only homogeneous static devices, but also heterogeneous dynamic devices. To process these data in the edge in an efficient way, such as applying the advanced parallel computing techniques, we need an effective data aggregation in IoT, since a non-optimized data aggregation solution could result in performance bottlenecks for parallel computing. For the interesting of this work, we focus on the load balancing issues in such scenarios.

To date, many solutions for data aggregation in IoT networks and load balancing for parallel computing have been proposed respectively. However, almost all the techniques in the two fields are total independent from each other, and the research on optimizing the performance on both aspects is limited for two main challenges.

CCGrid'19, online at https://doi.org/10.1109/CCGRID.2019.00077

- Different Requirements: The requirements on IoT network and parallel computing are different. On one hand, most solutions for data aggregation in IoT focus on increasing the communication performance, such as data aggregation speed. On the other hand, most solutions for parallel and distributed computing focus on balancing the data size aggregated in servers to improve their load balancing. For example, from the IoT viewpoint, the deployed servers should be close to most of the IoT devices, so that the communication latency and energy consumption can be minimized. However, from the computing viewpoint, balanced data size on the servers could greatly speed up the data processing [1].
- Dynamic Networks: The data aggregation solution should not only consider the topology of a network, but also the dynamic objects in the network. As a dynamic object moving inside the detection range of IoT devices, such as proximity sensing, an IoT device would produce data triggered by the object. In this condition, the data aggregation scheduling should be adaptive to the dynamic networks. However, most existing solutions just focus with a static data flow pattern.

In this paper, we propose a deep reinforcement learning (DRL) based solution for IoT network clustering. In each cluster, the data produced from cluster member devices is aggregated to the edge server located in the cluster. In general, our solution has two main properties.

- We assume each IoT device produces regular data of a fixed size periodically. Our solution partitions the IoT network into clusters to maximize the performance that is related with data aggregation in both edge computing and IoT networks.
- We assume objects move in the area of IoT network, which trigger the neighbor IoT devices to produce data with larger size than regular data. Our solution adapts the pattern of network clustering to the changing of aggregated data.

To achieve the goal, we propose a Deep Q-Learning Network (DQN) model. The preliminary experiments prove the feasibility of our solution. The experimental results show that the DQN model improves the system performance significantly in the IoT network with dynamic objects compared with the solution using static partitioned clusters.

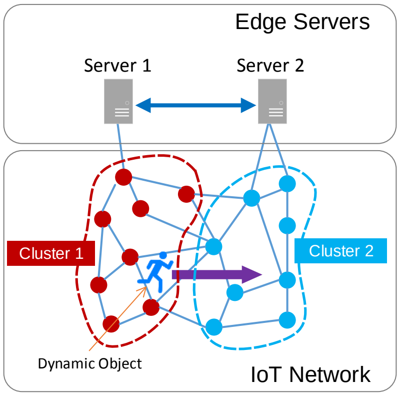


Fig. 1. The system model of IoT network and parallel servers. The data aggregation is based on the partitioned clusters.

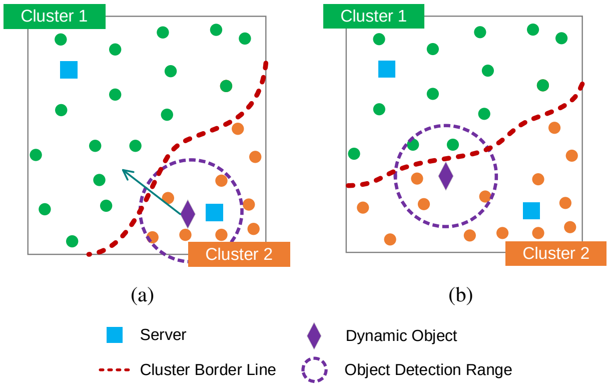


Fig. 2. The dynamic object triggers the neighbor IoT devices to produces data of larger size. As the object moves from the position as in (a) to (b), the cluster partition changes to maintain the balance of aggregated data in the servers.

The paper is organized as follows. Related work is discussed in Section II. The system model is present in Section III. The DQN based solution is explained in Section IV. The preliminary experimental results are presented in Section V. Finally, we discuss the future work in Section VI and conclude the paper in Section VII.

II. Related Work

Cluster based data aggregation is widely used in wireless networks [2] [3] [4]. Typically, the network nodes are partitioned to several groups, and the member nodes in a cluster send the data to the cluster header. The network clustering is used to optimize the performance of the network, such as energy consumption, communication latency, etc. Most of the network clustering solutions are designed based on the global information of network. To cope with the situation without global information, some researches focus on self-adaptive self-organization systems [5]. For example, [6] makes the network converge to the optimized network clustering by the local decision of each node using evolutionary theory.

To cope with more complicated resource management problems in networks, some researches exploit Reinforcement Learning (RL). For example, [7] presents a solution that translates the tasks with multiple resource demands into a RL problem. Its experimental results show that the RL model has comparable performance to the state-of-the-art heuristic solutions. [8] presents a resource allocation mechanism based on DRL in vehicle-to-vehicle communications. The solution makes each agent optimize sub-band and transmission power for satisfying the latency constraints using distributed local information.

III. System Model

The system model includes a IoT network and edge computing servers as shown in Figure 1. The IoT network consists of homogeneous nodes. Each node produces data periodically, and transfers messages to the servers by multi-hop communication. Several edge servers are deployed to make parallel computing for the aggregated data from the IoT network. For collecting data from IoT devices to multiple servers, the network is partitioned into clusters, and each edge server resides in a cluster. Each server is responsible for collecting the data from the IoT devices of a cluster.

We suppose each IoT device sends regular report data of a fixed size periodically to the server of its cluster. An object moves in the area of IoT network, and triggers the neighbor IoT devices to produce sensing data of larger size. For example, the object could be a person moving around. The sensor attached to each IoT device detects the proximity of person. Once a person appears in the detection range of the device, the IoT device captures the image of the person and sends image data to the server of its cluster.

The network clustering depends on the position of the moving object in the IoT network and the data aggregation requirements. Figure 2 presents an example. To guarantee the aggregated data size in both servers are equal, the network clustering pattern must change to adapt to the movement of object in the area. In this paper, we utilize DRL approach to find the optimized network clustering in the IoT network with dynamic objects.

IV. Deep Reinforcement Learning Solution

We use Deep Q-Learning Network (DQN) as the DRL model for dynamic network clustering. We assume that the agent performing DQN model resides in a server, and all the required information about IoT network are collected to the DQN model.

A. Actions

Suppose a set of servers $P = \{p_1, ...p_i, ...p_n\}$ reside in the network with $i \in [1, n]$. A cluster-core o_i is set at the position of server p_i. To partition the IoT network into clusters, we first select the IoT network node that is closest to the cluster-core o_i as the cluster header h_i. After that, we partition the network into clusters $C = \{c_1, ...c_i, ...c_n\}$ with $i \in [1, n]$ based on the cluster headers. To simplify the implementation, we partition

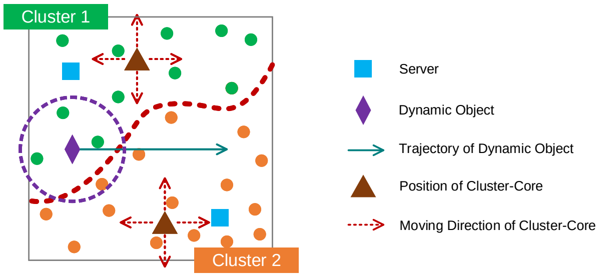


Fig. 3. The IoT network is partitioned into clusters based on the movement position of cluster-cores.

the network to Voronoi clusters. After the initialization, each server resides in a cluster.

To control the partition of clusters, we move the cluster-cores of clusters. The action state space A is the movements of all the cluster-cores. In our implementation, each cluster-core has five movement actions: Up, Down, Left, Right, Stay. At each time step, the DQN model selects and performs a movement action on a cluster-core. After the movement, the network re-selects the IoT network node that is closest to the cluster-core o_i as the cluster header h_i, and re-partitions the network into clusters based on the new cluster headers. An example of moving cluster-cores and partitioning clusters is shown in Figure 3.

B. Reward

The action reward is the satisfactory to the requirements on both IoT network and edge computing aspects. We define the two requirements that the network clustering should fulfill as follows.

- Edge Servers: Name the size of data aggregated to each server p_i as d_i, and $D = \{d_1, ...d_i, ...d_n\}$ with $i \in [1, n]$. To maximize the parallel computing speed by edge servers, the aggregated data size should be balanced in the servers. We use the average absolute deviation value of D that is normalized in $[0, 1]$ to quantify this load balancing requirement. We have $W = \frac{1}{n}\sum_{i=1}^{n}[1 - \frac{|d_i - \text{mean}(D)|}{\text{mean(D)}}]$.
- IoT Network: Name the total number of communication hops to transfer all the data in cluster c_i to server p_i as b_i. $B = \{b_1, ...b_i, ...b_n\}$ with $i \in [1, n]$. As we assume the IoT network uses multi-hop communication, more communication hops means higher communication time for aggregating the same amount of data. To keep the speed for aggregating data of each cluster on the same level, the number of communication hops for transferring all the data in each cluster should be balanced. We use the average absolute deviation value of B that is normalized in $[0, 1]$ to quantify the requirement of IoT network. We have $H = \frac{1}{n}\sum_{i=1}^{n}[1 - \frac{|b_i - \text{mean}(B)|}{\text{mean(B)}}]$.

We combine the two indexes together as the reward of DQN model. The reward function at time t is expressed as $r_t = W \times H$. The objective of DQN model is to find a policy to maximize the expected cumulative discounted rewards R_t as follows, where $\beta \in [0, 1]$ is the discount factor for reward r_t and k is the number of time steps from t.

$$R_t = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \beta^k r_{t+k}\right] \tag{1}$$

TABLE I
THE PARAMETERS OF THE WIRELESS SENSOR NETWORK IN THE REAL DEPLOYMENT.

Parameters	**2 Clusters**	**3 Clusters**	**4 Clusters**
Learning Rate	0.001	0.001	0.001
Discount	0.9	0.9	0.9
Min Espilon	0.01	0.01	0.01
DNN Layers	3	3	3
Neurons in Layer 1	200	400	600
Neurons in Layer 2	100	200	300
Neurons in Layer 3	50	100	150

C. Q-Value

Name s_t as the input state data of the DQN model at time t. The input state space S includes: (i) the neighbor connectivity matrix of the network; (ii) the cluster ID of each node; (iii) the size of data that is produced in each node. A policy π is a mapping from the state space S to the action space A. DQN aims to find the optimal policy to maximize R_t. The Q-value $Q(s_t, a_t)$ is defined as the reward R_t when taking action a_t in state s_t using policy π. We use Q-value to measure the quality of a certain action in a given state. Our DQN model uses a fully connected DNN with weight set θ to approximate the Q-function. After taking the action, the agent receives the reward r_t and the IoT network moves to a new state s_{t+1}. We select the action to be taken in the state s_t is the one that maximizes the Q-value.

$$a_t = \arg\max_{a \in A} Q(s_t, a) \tag{2}$$

The improved policy with Q-values is based on the following update equation, where $Q^{'}$ is the improved policy and $\alpha \in [0, 1]$ is the learning rate.

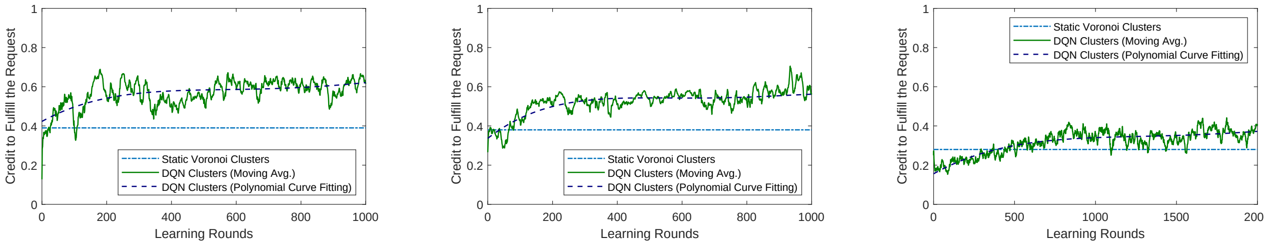
$$\begin{aligned} Q^{'}(s_t, a_t) =& Q(s_t, a_t) + \alpha[r_t + \\ & \beta \max_{a \in A} Q(s_{t+1}, a) - Q(s_t, a_t)] \end{aligned} \tag{3}$$

The DQN model updates its DNN at each iteration to minimize the loss function L.

$$L = [r_t + \beta \max_{a \in A} Q(s_{t+1}, a) - Q(s_t, a_t)]^2 \tag{4}$$

V. Preliminary Experimental Results

We evaluate the performance of the DQN model in a simulation IoT network. The network is deployed in an square area of 150m×150m. The nodes are randomly scattered in the area. As a preliminary experiment, we test the model in 40 nodes with 2 clusters, 60 nodes with 3 clusters, and 80 nodes with 4 clusters respectively. The data transmission range of nodes is 40m, 35m, 30m for 2, 3 and 4 clusters respectively.



(a) 2 Clusters.

(b) 3 Clusters.

(c) 4 Clusters.

Fig. 4. The credits of data aggregation using the DQN model in multiple clusters.

The detection range of objective is 2 times larger than the transmission range. We set one dynamic object in the IoT area. The speed of object is 1m/s, and it is initialized at the position of a random cluster header. We randomly select another cluster header as the moving direction of the object. The object moves to the direction until it is outside the deployment area. After that, the object moves to the opposite direction. We assume each node produces one message of 1Kb every 10 seconds. If the object is in the detection range of a node, the node produces a message of 10Kb. The communication speed between IoT nodes is 1Kb/s. To simplify the simulation, we ignore the interference in wireless communication. The DQN model is processed in every 10 seconds. Each round of training lasts for 500 seconds. The parameter values of DQN model are shown in Table I.

We use a static cluster partition algorithm without DQN model for comparison. In the comparison solution, each server is set as a cluster header, and the network is partitioned to Voronoi clusters. We compare the average reward score between our DQN solution and the Voronoi clustering solution.

The experimental results of 2, 3 and 4 clusters are shown in Figure 4(a), 4(b) and 4(c) respectively. The results of DQN model are shown by the moving average with window size 11, and the result trend is illustrated by a 4 round polynomial curve fitting. In all the three experiments, our DQN approach has significant improvement compared with the benchmark results. After convergence, the improvements are around 56%, 47%, and 32% in 2, 3 and 4 clusters respectively. The improvement decreases as the number of clusters increases. The main reason is that we only select one cluster-core to move in a time step. If all cluster-cores perform actions in a time step, the chance to find the cluster partition with higher reward will increase.

VI. Future Work

There are multiple points in the DRL model that we will improve in the future work.

- We use only one object with direct moving pattern in the implementation. This moving pattern cannot represent the real dynamic objects in IoT systems. We will evaluate whether the reinforcement learning model could cope with multiple objects moving in a more real pattern, such as random walk.
- To simplify the DQN model, we set one cluster-core in each cluster. In our next step work, we will set multiple cluster cores in a cluster. This will provide smaller granularity for controlling the cluster shape.
- This paper assumes that the IoT network is static, and only the object moves around. It is unknown whether the reinforcement learning solution can cope with a IoT network in which every node moves randomly.

VII. Conclusions

This paper presents a DRL solution for dynamic network clustering in IoT networks with edge servers. The aim is to optimize the data aggregation clustering to fulfill the requirements from both the IoT network and edge computing aspects. Our preliminary experiments show that the proposed DQN model can achieve better results compared to the static benchmark solution. In such scenarios, our work has showed that it is feasible to apply DRL techniques to IoT networks and edge computing. Moreover, we also believe that our design and implementation have offered an alternative to the current network clustering solutions.

References

[1] L. Cheng, S. Kotoulas, T. E. Ward, and G. Theodoropoulos, "Improving the robustness and performance of parallel joins over distributed systems," *Journal of Parallel and Distributed Computing*, vol. 109, pp. 310–323, 2017.

[2] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *System sciences, 2000. Proceedings of the 33rd annual Hawaii international conference on*. IEEE, 2000, pp. 10–pp.

[3] G. Chen, C. Li, M. Ye, and J. Wu, "An unequal cluster-based routing protocol in wireless sensor networks," *Wireless Networks*, vol. 15, no. 2, pp. 193–207, 2009.

[4] J. Yu, Y. Qi, G. Wang, and X. Gu, "A cluster-based routing protocol for wireless sensor networks with nonuniform node distribution," *AEU-International Journal of Electronics and Communications*, vol. 66, no. 1, pp. 54–61, 2012.

[5] Q. Liu, A. Pruteanu, and S. Dulman, "Gradient-based distance estimation for spatial computers," *The Computer Journal*, vol. 56, no. 12, pp. 1469–1499, 2013.

[6] Q. Liu, S. Dulman, and M. Warnier, "Area: an automatic runtime evolutionary adaptation mechanism for creating self-adaptation algorithms in wireless networks," *2013) colocated with AAMAS (W09)*, p. 23, 2013.

[7] H. Mao, M. Alizadeh, I. Menache, and S. Kandula, "Resource management with deep reinforcement learning," in *Proceedings of the 15th ACM Workshop on Hot Topics in Networks*. ACM, 2016, pp. 50–56.

[8] H. Ye and G. Y. Li, "Deep reinforcement learning for resource allocation in v2v communications," in *2018 IEEE International Conference on Communications (ICC)*. IEEE, 2018, pp. 1–6.