Towards Energy Efficient Mobile Data Collection In Cluster-based IoT Networks

Sana Benhamaid, Hicham Lakhlef, Abdelmadjid Bouabdallah

2021 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops) | 978-1-6654-0424-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/PERCOMWORKSHOPS51409.2021.9431037

Towards Energy Efficient Mobile Data Collection In Cluster-based IoT Networks

Sana Benhamaid *, Hicham Lakhlef *, Abdelmadjid Bouabdallah *
* Sorbonne Universités, Université de Technologie de Compiègne, CNRS,
CS 60319, 60203 Compiègne, France
sana.benhamaid@hds.utc.fr, hlakhlef@utc.fr, madjid.bouabdallah@hds.utc.fr

***Abstract*—Mobile data collection is a very efficient solution to gather information from spatially distributed IoT nodes. In order to enhance the efficiency of mobile data collection, the trajectory planning of the mobile node has been widely studied. In the literature, most of the solutions proposed use static and non learning-based approaches. In our paper, we opted for a learning based approach in which we study a trajectory planning problem in mobile data collection for IoT where IoT nodes are organized in clusters. A relay node is chosen from each cluster in order to collect data from IoT nodes and transmit it to the mobile node in its range. In order to plan the trajectory of the mobile node, we train a deep Q-learning network (DQN) with combined experience replay. This solution will allow us to maximize the amount of data collected and reduce the energy consumption. It will also allow us to adapt the trajectory of the mobile node to the environment parameters without doing expensive recomputations and learning.**

***Index Terms*—Internet of Things, mobile data collection, trajectory planning, deep reinforcement learning**

I. Introduction

One of the major challenges of IoT networks is the lifetime of the battery limited IoT devices. IoT devices consume large amounts of energy in order to communicate data with their neighbours. These communications will quickly lead to the devices battery depletion. In the literature, various authors studied how to reduce the energy consumption of spatially distributed IoT nodes. Researchers have studied cluster-based energy efficient solutions which consist on regrouping IoT devices into clusters and elect a node depending on its energy capability. This node will serve as a relay node between the IoT devices of the cluster and the gateways or receivers. The relay node will be responsible for sending data and will allow IoT devices to save their energy by only sending their data to the relay node [1]. In order to enhance the energy efficiency of cluster-based solutions, authors proposed energy efficient routing protocols applied to these contexts. For example, in [2], the authors proposed an energy efficient cluster-based routing protocol for wireless sensor networks that uses different parameters in order to balance the load of energy and decrease the energy consumption of the network. However, using a routing protocol to reduce the energy consumption in an IoT network implies that these devices must be connected to each other which is not always realistically possible if the nodes are placed in isolated or harsh conditions. One of the solutions proposed to tackle the limitation of routing-based energy efficient solutions are the mobile sink-based solutions. In this case, a node will be responsible for collecting the data of all IoT devices in a cluster of devices and transmitting it to the mobile node when the mobile node is in its range. Mobile data collection is a very promising energy saving solution for IoT networks. However, the most challenging part in mobile sink-based solutions is to determine and plan the trajectory of the mobile sink in order to gather data from the nodes. Most existing approaches to mobile data collection are static and only find a solution for a scenario with fixed parameters. These solutions do not consider a change of context in the system such as a change in the amount of data generated by the IoT nodes in the clusters or the mobility of the IoT devices where an IoT device can move from a cluster to another.

Recently, researchers have been interested in using artificial intelligence techniques such as machine learning or deep learning, in order to propose intelligent mobile data collection solutions that will adapt the trajectory of the mobile node depending on the activity of the IoT nodes and the environment. In our solution, we will use a machine learning paradigm called reinforcement learning. Reinforcement learning is suitable for complex environment that need adaptability depending on the context. This technique will allow us to have a path planning solution that will adapt the trajectory of the mobile node in order to maximize the amount of collected data. Unlike other existing solutions, our solution will focus on reducing the energy consumption and plan the trajectory of the mobile node depending on the activity of the clusters (e.g. a cluster which has no data to send will be less likely to be visited by the mobile node).

II. Background and brief description of our solution

A. Background on machine learning

In learning based problems where we need our agent or system to learn a behavior and make informed decisions, machine learning solutions have proved to be the most efficient. Machine learning algorithms learn from data or experiences and apply what they have learned to make informed decisions. They usually involve human intervention that gives feedback to the algorithm. Machine learning algorithms are divided into three categories : *supervised, unsupervised and reinforcement learning.* For our solution,

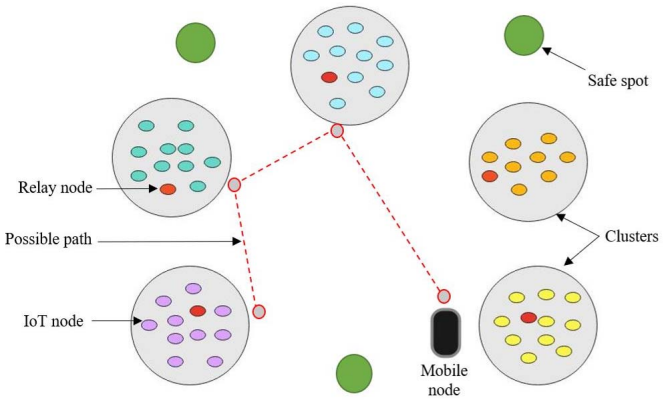

Fig. 1: Description of our system and a possible scenario and trajectory for the mobile node

we chose to use reinforcement learning. In this section, we will present the three machine learning paradigms and why reinforcement learning is the most appropriate approach for our problem rather than supervised or unsupervised learning. Reinforcement learning is a machine learning paradigm that is inspired from human learning. It is based on the concepts of agent, state, action and reward. The objective of the agent is to learn which actions to take in order to maximize the reward given a state of its environment and a set of possible actions.

Reinforcement learning is different from supervised learning which is the most studied machine learning paradigm. Supervised learning is learning from a set of labeled data provided by a knowledgeable external supervisor [3]. The goal of this type of learning is to build a system able to act correctly when a situation is not included in the training data set. Supervised learning alone is, consequently, not suitable for interactive problems since in our solution the mobile node needs to interact with its environment and learn from its own experience. Reinforcement learning is also different from unsupervised learning. Unsupervised learning is a learning paradigm that aims to find correlations in sets of unlabeled data. This type of learning is not based on correct examples. Finding structure in a reinforcement learning agent experience might be interesting. However, in our solution, the most important objective is to maximize the reward (e.g. maximize the collected data and minimize the energy consumption) which is not covered by unsupervised learning. Consequently, reinforcement learning is the most suitable technique to our problem.

B. Brief description of our solution

Our solution consists on planning the trajectory of a mobile node using a deep reinforcement learning algorithm. As shown in Fig. 1, the mobile node will successively visit clusters and collect data from an elected node in the cluster called relay node. The relay node is responsible for collecting the data from the IoT nodes in the cluster and transmitting it to the mobile node. The objective of our solution is to maximize the collected data while minimizing the energy consumption of the mobile node and IoT devices.

Our solution will also take into consideration the change of activity in the IoT network which means that the mobile node will be less likely to visit a relay node with no data to send. As shown in Fig.1, the mobile node chose to ignore a cluster that do not meet certain conditions. The mobile node has information on the states of all relay nodes (amount of data to collect). The learning algorithm is also computed on the mobile node. Other solutions proposed reinforcement learning solutions to plan the trajectory of mobile nodes in order to either optimize data collection or minimize the energy consumption and need to perform expensive computations in order to adapt to a change in the context. To the best of our knowledge, our solution is the only one solution that uses deep reinforcement learning in a cluster-based IoT network in order to maximize the data collection, reduce energy consumption of the mobile node and takes into consideration the variation in activity of the different clusters.

III. Related works

In order to plan the trajectory of mobile nodes in a data collection context, authors proposed non learning based solutions. For example, in [4], the authors proposed a framework that optimizes the deployment and mobility of multiple UAV in order to collect data in the uplink from ground IoT devices and minimize the energy consumption of the mobile nodes. In [5], the authors proposed an algorithm that optimizes the UAV stops for data collection from neighboring sensors and the itinerary followed by the UAV in order to ensure efficient collection of all data with minimum energy consumption. However, in IoT networks, devices do not constantly collect and transmit data, their activity depend on the environment around them or the period of time the node is visited in. In the previous solutions a change in the activity of the IoT network is not considered. Consequently, having a mobile sink periodically collecting data, following a static trajectory and visiting devices that have no data to send may cause energy waste and does not constitute an optimal solution to achieve energy efficiency.

In the literature, most existing data collection and trajectory planning solutions use non-learning based approaches. However, in the recent years, researchers have considered learning based solutions and especially reinforcement learning as a very promising trend in this field. In [6], the authors proposed a distributed reinforcement learning approach for path planning and collision avoidance of UAVs. The optimal path design is studied for IoT networks with devices having different communication radio. In [7], the authors proposed a deep reinforcement learning scheme that plans the trajectory of a mobile node which is used as a data collector and charger in wireless powered IoT networks. The solution minimizes the average data buffer length, minimizes the residual battery level of the system and avoid devices data overflow.
In [8], the authors proposed a multi-agent reinforcement

learning approach that allows the control the trajectory of a team of cooperative UAVs in order to maximize the collected data under flying time and collision constraints. In [9], the authors proposed a double deep Q network to plan the trajectory of an UAV on an IoT data harvesting mission. The solution proposed aims to maximize the collected data under flying time and navigation constraints and allows the UAV to adapt to variations in the number of IoT devices. On contrary to our solution this solution does not take into consideration the inconsistencies in the IoT devices activity (e.g a device can collect more data on a certain period of the day more than another).

IV. System Model and problem formulation

A. System Model

We consider an IoT network composed of a number of clusters randomly distributed in a square grid world of size $M \times M \in \mathbb{N}^2$ with a mobile node collecting data through a relay node from a number K of clusters of IoT devices. We suppose that the relay node is already chosen depending on its energy capability. The k-th relay node is permanently located at $[x_k(n), y_k(n)]$ in the grid world with $k \in [1, K]$ and $n = 1, 2, ..., N$, represents the number of time slots. We consider the velocity $v(n)$ of the mobile node constant and enough for the mobile node to move from one square of the grid to the one next to it. We consider the velocity as constant since we want our energy efficient solution to be adapted to any type of mobile node. At time step *n*, the mobile node has an energy capacity $w(n)$ and $w_c(n)$ the amount of energy needed to visit the next cluster at time step n. We also consider $w_f(n)$ the minimum level of energy needed for the mobile node to move to a safe stop spot z with z in$[1, Z]$. A safe spot is an area in the square grid where the mobile node will go to when its energy level is not sufficient to visit another cluster in order to recharge its energy.

B. Markov Decision Process

In order to find an optimal data collection policy that maximizes the data collection and minimizes the energy consumption of the mobile node. We will formulate our problem as a Markov decision process problem (MDP). In this section, we define the state space, action space and reward function. We solve this MDP problem using reinforcement learning. The MDP is defined by the tuple (S, A, R, P) with state-space S, action-space A and reward function R.

The state in time t, is given by $s_t = p_t, D_t, e_t, C$ where:

- $p_t \in \mathbb{R}^2$ is the mobile node position on the grid;
- $D_t \in \mathbb{R}^{2\times K}$ is the collected data and the remaining data for each cluster head;
- $e_t \in \mathbb{N}$ is the remaining energy of the mobile node in time t;
- $C \in \mathbb{R}^{2\times K}$ represents the coordinates of the K cluster heads.

The mobile node is limited to fly to one of the four adjacent grids from its current grid in each time slot. The action-space is defined as

$$A = \text{north, east, south, west}$$

The movement of the mobile node from a position p_t is expressed as

$$p_{t+1} = \begin{cases} p_t + (-X, 0) & \text{if } a_t = west \\ p_t + (X, 0) & \text{if } a_t = east \\ p_t + (0, X) & \text{if } a_t = north \\ p_t + (0, -X) & \text{otherwise} \end{cases}$$

where *X* is length of a square in the grid. The reward function is a function maps state-action pairs to a real-valued reward, i.e. $R : S \times A \to \mathbb{R}$. It consist of the following:

- r_{ed} is a reward given if the data collected is superior than a given threshold and the energy consumed is less than a given threshold;
- r_{sd} is a reward given if the data collected is superior to a given threshold but the energy consumed is superior than a given threshold;
- r_{ee} is a reward given if the data collected is less than a given threshold but the energy consumed is less than a given threshold;
- r_{ne} is a penalty given if the data collected is less than a given threshold but the energy consumed is more than a given threshold;
- r_{move} is a penalty given each time the mobile moves without collecting data;
- r_{finish} represents a penalty given in case the mobile node energy reaches zero without being in a safe spot.

V. Methodology

A. Q-Learning

Q-learning is a model-free reinforcement learning technique that helps finding an optimal policy and maximize the expected value of the total reward from the current state and all the consecutive states. For our solution, we chose to use Q-learning since we want the mobile node to freely explore the environment which will help it modify its trajectory in case a new cluster of IoT nodes appears rather than using a more conservative algorithm like SARSA. Formally, for an autonomous agent observing a state *s* of its environment at time step *n* $s_n \in S$ where S is the set of states. The agent executes an action $a_n \in A$, where A is the set of possible actions and interacts with its environment. This action changes the state of the environment to a new state s_{n+1} and the agent will receive a reward r_n accordingly. The environment is expected to be non-deterministic as taking the same action in the same state on different occasions may result in different states and different rewards. The agent's goal is to find a policy π that maps a state s_n to a probability of choosing action a_n and can be represent as following

$$\pi : S \to P(A)$$

The reward is the sum of discounted future rewards. To calculate the reward we use γ, where $\gamma \in [0, 1]$ is a discount factor that determines the effect of the future rewards to the current one.

$$R_n = \sum_{i=n}^{N} \gamma^{i-n} r(s_i, a_i)$$

The optimal policy π^* is defined as

$$\pi^* = argmax\mathbb{E}[R_n|\pi]$$

This optimal policy has an optimal state-action value function that satisfies Bellman optimality equation. It is described by the following

$$Q^{\pi^*}(s,a) = \mathbb{E}[R_{n+1} + \gamma maxQ_*(s_{n+1}, a_{n+1})]$$

Algorithm 1 DQN-based Energy Efficient Mobile Data Collection

initialize action-value with random weights θ and θ', replay memory size *H*, number of episodes *F*
initialize ϵ for exploration and ϵ_{expt}
for *episode* :$1, ..., F$ **do**
  Initialize the environment and receive initial state s_1;
  Set $n = 0$;
  while $w > w_c$ **do**
    if $\epsilon > \epsilon_{expt}$ **then**
      Choose a_n randomly from action space;
    else
      Select $a_n = argmaxQ(s_n, a_n, \theta)$;
    end
    Execute action a_n, compute r_n and observe the next state s_{n+1};
    Store experience (s_n, a_n, r_n, s_{n+1}) in the replay memory with a random placement policy;
    Sample a random mini-batch of G experiences (s_i, a_i, r_i, s_{i+1}) from replay memory;
    Calculate the target value;
    Update the weights $\theta' = \theta$ every U time step;
    Decrease w;
    Set $n = n + 1$;
  end
  Decrease ϵ;
end

B. Deep Q-learning

Q-learning is a very efficient algorithm when the environment is limited to small state spaces. However, computing and updating table values for each state-action pair is not efficient when it comes to more complex and sophisticated environments with large state and action such as our environment problem. In these cases, it is more interesting to find an approximation of the Q-value rather than directly computing it. In order to do that, we will use Deep Q-Learning which is technique that uses neural networks to approximate the optimal Q-function $Q_\pi(s, a)$.

Deep Q-learning often uses experience replay which is a technique that allows us to store the agent's experiences at each time step n in a data set called replay memory. At time step *n*, the agent's experience is defined as following

$$e_n = (s_n, a_n, r_{n+1}, s_{n+1})$$

The main reason for using experience replay is to break the correlation of consecutive samples. In our solution we uses two neural networks. The policy network θ which approximates the optimal policy by finding the optimal Q-function. It accepts the current state s_n and finds the evaluation of the value $Q(s_n, a_n, \theta)$. We also use a second network called target network θ' to improve the stability of learning. The target network weights are frozen with the original policy network and are updated periodically. It accepts the next state s_{n+1} and outputs the Q-value $Q(s_{n+1}, a_{n+1}, \theta')$. These values are optimized to minimize the loss function defined by

$$L(\theta) = \mathbb{E}[(T_n - Q(s_n, a_n))^2]$$

where T_n is the target value which is defined as following

$$T_n = r_n + \gamma^{n-1} maxQ(s_{n+1}, a_{n+1}, \theta')$$

where the Q-value for the next state s_{n+1} is passed to the target neural network θ' for more stability in learning. ϵ is the exploration rate which represents the probability that our mobile node will explore the environment and ϵ_{expt} is the threshold from which our mobile node will stop exploring the environment and will exploit the experience acquired through the policy network.

VI. Conclusion

In this paper, we have introduced a deep Q-learning method with experience replay for trajectory planning of a mobile node in a mobile data collection and cluster-based scenario. We are currently working on training a neural network that uses information about the environment to learn and find an energy efficient trajectory for the mobile node and adapt to significant changes in the context scenario (e.g. the amount of collectible data) without the need for expensive retraining or recollection of data.

References

[1] L. Song, K. K. Chai, Y. Chen, J. Schormans, J. Loo, and A. Vinel, "Qos-aware energy-efficient cooperative scheme for cluster-based iot systems," *IEEE Systems Journal*, vol. 11, no. 3, pp. 1447–1455, 2017.

[2] A. S. Toor and A. Jain, "Energy aware cluster based multi-hop energy efficient routing protocol using multiple mobile nodes (meacbm) in wireless sensor networks," *AEU-Inter. Journal of Electronics and Communications*, vol. 102, pp. 41–53, 2019.

[3] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[4] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Mobile unmanned aerial vehicles (uavs) for energy-efficient internet of things communications," *IEEE Trans. on Wireless Communications*, vol. 16, no. 11, pp. 7574–7589, 2017.

[5] M. B. Ghorbel, D. Rodríguez-Duarte, H. Ghazzai, M. J. Hossain, and H. Menouar, "Joint position and travel path optimization for energy efficient wireless data gathering using unmanned aerial vehicles," *IEEE Trans. on Vehicular Technology*, vol. 68, no. 3, pp. 2165–2175, 2019.

[6] Y.-H. Hsu and R.-H. Gau, "Reinforcement learning-based collision avoidance and optimal trajectory planning in uav communication networks," *IEEE Trans. on Mobile Computing*, 2020.

[7] J. Zhang, Y. Yu, Z. Wang, S. Ao, J. Tang, X. Zhang, and K.-K. Wong, "Trajectory planning of uav in wireless powered iot system based on deep reinforcement learning," in *2020 IEEE/CIC Inter. Conf. on Communications in China (ICCC)*. IEEE, 2020, pp. 645–650.

[8] H. Bayerlein, M. Theile, M. Caccamo, and D. Gesbert, "Multi-uav path planning for wireless data harvesting with deep reinforcement learning," *arXiv preprint arXiv:2010.12461*, 2020.

[9] ——, "Uav path planning for wireless data harvesting: A deep reinforcement learning approach," *arXiv preprint arXiv:2007.00544*, 2020.