Sumedh Dongare, Andrea Ortiz and Anja Klein, ”Deep Reinforcement Learning for Task Allocation in Energy Harvesting Mobile Crowdsensing,” in *Proc. of the IEEE Global Communications Conference - (IEEE GLOBECOM 2022)*, December 2022.

Deep Reinforcement Learning for Task Allocation in Energy Harvesting Mobile Crowdsensing

Sumedh Dongare, Andrea Ortiz, Anja Klein
Communications Engineering Lab, Technical University of Darmstadt, Germany.
{s.dongare, a.ortiz, a.klein}@nt.tu-darmstadt.de

***Abstract*—Mobile crowd-sensing (MCS) is an upcoming sensing architecture which provides better coverage, accuracy, and requires lower costs than traditional wireless sensor networks. It utilizes a collection of sensors, or crowd, to perform various sensing tasks. As the sensors are battery operated and require a mechanism to recharge them, we consider energy harvesting (EH) sensors to form a sustainable sensing architecture. The execution of the sensing tasks is controlled by the mobile crowd-sensing platform (MCSP) which makes task allocation decisions, i.e., it decides whether or not to perform a task depending on the available resources, and if the task is to be performed, assigns it to suitable sensors. To make optimal allocation decisions, the MCSP requires perfect non-causal knowledge regarding the channel coefficients of the wireless links to the sensors, the amounts of energy the sensors harvest and the sensing tasks to be performed. However, in practical scenarios this non-causal knowledge is not available at the MCSP. To overcome this problem, we propose a novel Deep-Q-Network solution to find the task allocation strategy that maximizes the number of completed tasks using only realistic causal knowledge of the battery statuses of the available sensors. Through numerical evaluations we show that our proposed approach performs only 7.8% lower than the optimal solution. Moreover, it outperforms the myopically optimal and the random task allocation schemes.**

I. Introduction

Mobile crowd-sensing (MCS) is a sensing architecture which utilizes a collection of sensors or 'crowd' to perform sensing tasks [1]. With an increasing number of smart devices equipped with a variety of sensors and due to recent advancements in the Internet of Things (IoT), MCS has become a popular topic of research [2]. There are many advantages MCS has over the traditional wireless sensor networks (WSNs) such as lower infrastructure costs, higher coverage and wider range of applications due to sensor mobility [3]. As a consequence, MCS is rapidly becoming a viable alternative to WSNs in sensing applications. In fact, there are many existing applications ranging from traffic monitoring (FourSquare,Waze), environmental monitoring [4], spectrum sensing [5], to mHealth [6] which efficiently employ MCS.

A typical MCS scenario consists of data requesters, a mobile crowd-sensing platform (MCSP), and the sensors. The data requesters require some sensing data which they communicate to the MCSP as a request. The MCSP then reformulates this request as a sensing task and allocates it to the appropriate sensors. The sensors perform the task and transmit the sensed data back to the MCSP. The MCSP then provides the sensed data back to the data requesters. In an MCS scenario, the task allocation strategy used to choose the sensors which perform the sensing tasks has a strong impact on the completion of the tasks, the energy efficiency of the system, and the quality of the sensed data. For example, tasks allocated to unsuitable sensors can take too long to be completed or the quality of the results might not be adequate. For this reason, one of the main challenges to maximize the number of completed tasks or coverage in MCS, is finding a suitable task allocation strategy.

The sensors are usually battery operated and therefore require a mechanism to recharge their batteries to form a sustainable sensing architecture. One option to achieve this is by using energy harvesting (EH) [7]. EH is a technology where the nodes can collect energy from the environment using different renewable energy sources [8]. EH has been previously used in the context of WSN [9], [10]. However, its application in MCS scenarios is not straightforward because the solution to the task allocation problem depends on the amounts of energy harvested by the sensors, which are usually not known in advance at the MCSP. The availability of the sensors to perform a sensing task is constrained by the energy available in their batteries. Therefore, there is a trade-off between selecting a sensor to perform a task at a given time and spend its energy, or saving the sensor's energy to perform more complex and energy demanding tasks in the future.

To make an optimal task allocation decision, the MCSP requires perfect non-causal knowledge of the time varying parameters of the MCS system, e.g., the channel coefficients of the wireless links to the sensors, the tasks to be performed, the sensor's locations or their trajectories. So far, the MCS research has focused on non-EH systems. The state-of-the-art approaches assume that the MCSP always has a sensing task to perform. Additionally, non-causal knowledge of the sensor's locations, their trajectories, and the channel coefficients is assumed at the MCSP. [11]–[14]. Specifically, in [11], the authors investigate a greedy approach for task assignment to minimize the energy consumption under a coverage constraint under the assumption of known sensor mobility information at the MCSP. In [12], the authors propose a genetic algorithm for the task allocation problem to maximize the task completion assuming the sensor locations are known. The authors then

This work has been funded by the German Research Foundation (DFG) as a part of the projects C1 and T2 within the Collaborative Research Center (CRC) 1053 - MAKI (Nr. 210487104) and has been supported by the BMBF project Open6GHub (Nr. 16KISK014) and the Loewe Center EmergenCity.

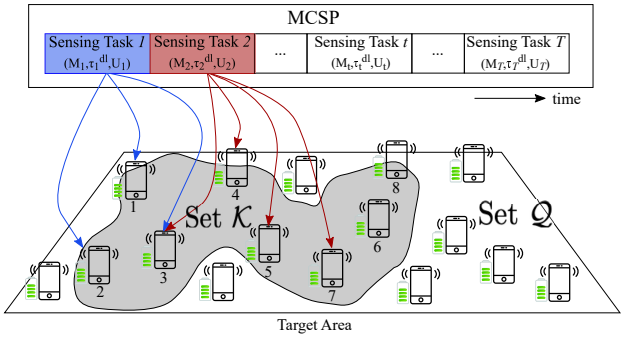

Fig. 1. System model with $K = 8$ reserved sensors

optimize the travelling path of the sensors with constraints based on known user preferences, such as travel distance to the task. In [13], the authors assume knowledge about the channel and prior user trajectories. They propose a machine learning-based approach for task allocation by formulating a path planning problem to maximize the MCSP's profit and coverage under deadline constraints. In [14], the authors formulate an optimization problem to achieve a given quality of information level from the MCS under the constraint of minimum energy consumption. The authors assume knowledge about the channel coefficients and the user trajectories at the MCSP. In practical scenarios, this knowledge may not always be available at the MCSP.

We formulate the task allocation problem in an MCS scenario to maximize the number of completed tasks in a finite time horizon. To form a sustainable sensing architecture, we assume EH sensors. Moreover, the sensing tasks are deadline constrained. We propose a novel Deep-Q-Network Reinforcement Learning (RL) approach to find the task allocation strategy that maximizes the number of completed tasks, without the strict requirement of non-causal knowledge about the channel coefficients, the amounts of harvested energy, and the tasks to be performed. In particular, our proposed approach learns to make allocation decisions considering the trade-off between selecting a sensor at a given time and spending its energy, or not selecting it and thus save its energy to perform more complex and energy demanding tasks in the future.

In the rest of the paper, Section II presents the system model. Section III describes the task assignment problem in EH MCS to maximize the number of completed tasks. Our proposed algorithm is explained in Section IV and the simulation results to compare the performance of the proposed algorithm with reference schemes are presented in Section V. Finally, Section VI concludes the paper.

II. System Model

We consider an MCS scenario which includes an MCSP with sensing tasks and multiple EH sensors distributed randomly in an area, as depicted in Fig. 1. As in [15], we consider a time slotted structure where a finite time horizon T_{H} is divided equal time steps indexed with $t \in \{1, 2, \ldots, T\}$. The duration of each time step is τ^{int}. The duration of one time horizon is then, $T_{\mathrm{H}} = T\tau^{\mathrm{int}}$. All the available sensors in the area are included in the set $\mathcal{Q}$. At the beginning of the time horizon, the MCSP broadcasts a participation request to all sensors in $\mathcal{Q}$ with indices $q = \{1, 2, \ldots, Q\}$. The participation request informs the sensors about the duration of the time horizon T_{H} for which the sensors will be reserved to perform sensing tasks. We assume a subset $\mathcal{K} \subseteq \mathcal{Q}$ of sensors responds positively to the request. These $k = \{1, 2, \ldots, K\} \in \mathcal{K}$ sensors are reserved for the time horizon T_{H}.

At the beginning of each time step t, a task arrives at the MCSP with probability $\lambda \in (0, 1]$. As we consider at maximum one task per time step, the variable t is used as the task index as well as the time step index. Each task t is characterized using a tuple $\langle M_t, \tau_t^{\mathrm{dl}}, U_t \rangle$ where M_t denotes the sensing task size measured in bits, τ_t^{dl} is a task specific deadline and U_t describes the number of sensors required to execute the task t. We assume the maximum task size is $M_{\max}$ and $\tau_t^{\mathrm{dl}} \leq \tau^{\mathrm{int}}$ for all t. The tuple $\langle M_t, \tau_t^{\mathrm{dl}}, U_t \rangle$ is provided by the data requesters as a requirement to be fulfilled in order to complete the task t. U_t can be based on the requester's budget and the number of samples it requires for that sensing task such that $U_t \leq K$. We define a set of sensors which are assigned the task t as $\mathcal{K}_t$. Therefore, $|\mathcal{K}_t| = U_t$ must hold.

Immediately after receiving a task, the MCSP makes a task allocation decision in two consecutive steps: a task acceptance and a task assignment decision. The task acceptance decision is represented by a binary variable x_t. In some time steps it is not possible for the sensors to complete the task because of the task requirements might be too strict. In this situation, the MCSP should decide to drop the task, i.e., $x_t = 0$, in order to save the sensor's energy. If the MCSP decides to perform the task t, i.e., $x_t = 1$, then the MCSP decides to which sensors to assign this task. The task assignment decision for each sensor k in time step t is denoted by $y_{k,t}$. If a sensor k is chosen for a task t, then $y_{k,t} = 1$, otherwise $y_{k,t} = 0$. We assume that allocation of the sensing task to the sensors requires negligible time. The task acceptance and the task assignment decisions for all t are stored in the vector $\mathbf{x} = (x_1, x_2, \ldots, x_T)^{\mathrm{T}}$ and matrix $\mathbf{Y} = (y_1, y_2, \ldots, y_T)$ where $y_t = (y_{1,t}, y_{2,t}, \ldots, y_{K,t})^{\mathrm{T}}$, respectively. We assume that the task is independently executed by each sensor k once the task is allocated to it, that means, each sensor will generate its sensing data independently.

For the execution of task t, the sensor k spends time $\tau_{k,t}^{\mathrm{s}}$, and energy $E_{k,t}^{\mathrm{s}} = p_t^{\mathrm{s}} \tau_{k,t}^{\mathrm{s}}$ for sensing, where p_t^{s} is the power required for sensing. After the sensing, the sensor transmits the sensed data back to the MCSP. For this purpose, each sensor k has its own wireless channel with channel coefficient $h_{k,t}$ in time step t and bandwidth W [15]. We assume that channels of different sensors are orthogonal to each other. Transmitting the sensed data back to the MCSP requires time,

$$\tau_{k,t}^{\mathrm{tx}} = \frac{M_t}{W \log_2 \left(1 + \frac{p_{\max}^{\mathrm{tx}} |h_{k,t}|^2}{\sigma^2}\right)}, \tag{1}$$

where $p_{\max}^{\mathrm{tx}}$ is the transmit power and σ^2 is the noise power. For the transmission, the sensor spends energy $E_{k,t}^{\mathrm{tx}} =$

TABLE I
NOTATIONS

Description	Notation	Description	Notation
Total number of sensors	Q	Sensing task t size	M_t
Length of the time horizon	T_{H}	Deadline of task t	τ_t^{dl}
Total reserved sensors for T_{H}	K	Number of required sensors for task t	U_t
Set of sensors assigned to task t	$\mathcal{K}_t$	Sensing, transmission and execution time of sensor k for task t	$\tau_{k,t}^{\mathrm{s}}$,$\tau_{k,t}^{\mathrm{tx}}$ and $\tau_{k,t}^{\mathrm{exec}}$
Task index, time step index	t		
Index of reserved sensor	k	Sensing, transmission and execution energy of sensor k for task t	$E_{k,t}^{\mathrm{s}}$, $E_{k,t}^{\mathrm{tx}}$ and $E_{k,t}^{\mathrm{exec}}$
Battery status of sensor k in time step t	$b_{k,t}$		
Battery capacity	$B_{\max}$	Channel bandwidth for sensor k	W
Channel coefficient of sensor k in time step t	$h_{k,t}$	Task acceptance vector	$\mathbf{x}$
Task arrival probability	λ	Task assignment matrix	$\mathbf{Y}$

$p_{\max}^{\mathrm{tx}}\tau_{k,t}^{\mathrm{tx}}$. Therefore, to perform a sensing task, the allocated sensor k requires time $\tau_{k,t}^{\mathrm{exec}} = \tau_{k,t}^{\mathrm{s}} + \tau_{k,t}^{\mathrm{tx}}$ and energy $E_{k,t}^{\mathrm{exec}} = E_{k,t}^{\mathrm{s}} + E_{k,t}^{\mathrm{tx}}$. Note that $E_{k,t}^{\mathrm{exec}} \neq 0$ only if task t is allocated to sensor k, otherwise $E_{k,t}^{\mathrm{exec}} = 0$. We assume that the task execution is complete only if the task deadline requirement is fulfilled. After the execution, the MCSP collects the sensed data and provides it back to the data requester. The task is completed if each sensor $k \in \mathcal{K}_t$ follows the deadline constraint such that $\tau_{k,t}^{\mathrm{exec}} \leq \tau^{\mathrm{int}}$. Else, the task is incomplete. If after the deadline task t is incomplete, all the sensors stop the execution and wait to be assigned to a new task.

The sensors harvest energy from the environment. The maximum amount of energy that can be harvested in one time step t is $E_{\max}^{\mathrm{harv}}$. Note that the sensors harvest energy also in all time steps t where there is no task to be performed. We assume that the energy $E_{k,t}^{\mathrm{harv}}$ harvested by sensor k is stored in its battery with capacity $B_{\max}$ without any losses. However, the harvested energy $E_{k,t}^{\mathrm{harv}}$ is available in the battery in time step $t+1$. The sensors can only use the energy which is stored inside the battery at the beginning of time step t. A new battery status $b_{k,t}$ is calculated at the end of t from the previous battery status $b_{k,t-1}$ as,

$$b_{k,t} = b_{k,t-1} - E_{k,t}^{\mathrm{exec}} + E_{k,t}^{\mathrm{harv}}. \tag{2}$$

All the reserved sensors K transmit their updated battery status $b_{k,t}$ to the MCSP via a dedicated control channel.

Note that our system model is not restricted to one particular device as a sensor. The sensors can be any sensing device such as smartphones, wearable, smart vehicles, or IoT devices, which are battery operated and have EH capabilities. A summary of the notation is presented in Table II.

III. Problem Formulation

A. Optimization problem

In this section, we formulate the task allocation problem in the considered MCS scenario to maximize number of completed tasks in a finite time horizon T_{H}. As described in section II, the task allocation decision made by the MCSP consists of two steps, the task acceptance and the task assignment for all tasks in time horizon T_{H}. These two decisions are stored in vector $\mathbf{x}$ and matrix $\mathbf{Y}$ respectively. Since tasks can have different requirements in terms of M_t,τ_t^{dl}, and U_t, they require different amounts of resources for their completion. To handle these different requirements, we introduce the normalized quantities $\left\langle M_t', \tau_t^{\mathrm{dl}'}, U_t' \right\rangle$ to form a single weight,

$$V_t = \xi M_t' + \omega \tau_t^{\mathrm{dl}'} + \psi U_t', \quad \forall t, \tag{3}$$

where the variables ξ, ω, $\psi \in [0,1]$ are importance factors for M_t', $\tau_t^{\mathrm{dl}'}$, and U_t' respectively. Their values can be modified to favour task execution w.r.t. the respective requirement. Specifically, M_t' is $\frac{M_t}{M_{\max}}$ where $M_{\max}$ is the size of the largest task, $\tau_t^{\mathrm{dl}'}$ is $1 - (\frac{\tau_t^{\mathrm{dl}}}{\tau^{\mathrm{int}}})$, and U_t' is $\frac{U_t}{K}$. Therefore, $M_t', \tau_t^{\mathrm{dl}'}, U_t' \in [0,1]$. If task size M_t is large, more resources are required for successful completion and hence, V_t increases. Since fulfilling shorter task deadlines is harder than longer ones, τ_t^{dl} affects the weight V_t inversely. Finally, U_t directly impacts V_t as a higher value of U_t implies more resources.

Since each task t has deadline τ_t^{dl}, the allocated sensors for task t must fulfil the deadline constraint given by,

$$\tau_{k,t}^{\mathrm{exec}} y_{k,t} \leq \tau_t^{\mathrm{dl}}, \quad \forall k \in \mathcal{K}_t, \forall t. \tag{4}$$

Furthermore, the task assignment decision $y_{k,t}$ is constrained by the requirement U_t as,

$$\sum_{k=1}^{K} y_{k,t} = U_t x_t, \quad \forall k \in \mathcal{K}_t, \forall t. \tag{5}$$

Notice that the task assignment decision $y_{k,t}$ depends on the task acceptance decision x_t made by the MCSP. Moreover, the contributions from all sensors are equally important. If a task t is allocated to U_t sensors, and if one or more of them cannot successfully perform the task under given requirements, then the task is incomplete.

The sensors cannot spend more energy than their current battery status $b_{k,t}$. To ensure this, an energy causality constraint is imposed as,

$$\sum_{j=1}^{J} E_{k,j}^{\mathrm{exec}} y_{k,t} \leq \sum_{j=0}^{J-1} E_{k,j}^{\mathrm{harv}}, \quad \forall k \in \mathcal{K}_t, \forall t, \forall J = 1, ..., T. \tag{6}$$

The energy causality constraint in (6) also ensures that the harvested energy cannot be used instantaneously and needs

to be stored in the battery first. Since sensors cannot store energy more energy than the battery capacity $B_{\max}$, an energy overflow constraint is imposed on the sensors as

$$\sum_{j=0}^{J-1} E_{k,j}^{\text{harv}} - \sum_{j=1}^{J} E_{k,j}^{\text{exec}} y_{k,t} \leq B_{\max}, \quad \forall k \in \mathcal{K}_t, \forall t, \forall J = 1, ..., T. \tag{7}$$

The optimization problem to maximize the average weighted sum of completed tasks is formulated as follows,

$$\begin{aligned} \underset{\{x_t, y_{k,t}\}}{\arg\max} \quad & \sum_{t=1}^{T} \sum_{k=1}^{K} x_t V_t \\ \text{subject to} \quad & (4), (5), (6), (7). \end{aligned} \tag{8}$$

The formulated task allocation problem is NP-hard. Also, the constraints are interdependent. To find an optimal task allocation strategy to maximize the average weighted sum of completed tasks, the MCSP requires perfect non-causal knowledge about the channel coefficients, the amount of harvested energy and the sensing tasks to be performed. Such non-causal knowledge is, in practical scenarios, usually not available at the MCSP. To overcome this challenge, we propose a Deep-Q-Network (DQN)-based approach which does not have this strict requirement. Our approach only assumes that the current battery statuses $b_{k,t}$ of the sensors and the requirements of the current task t are known.

B. Reformulation as Markov Decision Process

The MCSP has to make task allocation decisions in each time step. Such decision making problems can be modelled using a Markov Decision Process (MDP). Therefore, in this section, we reformulate the optimization problem as an MDP.

An MDP is characterized by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R} \rangle$ such that the set $\mathcal{S}$ contains the states which the decision making agent, the MCSP in our model, can experience. The state $S_t \in \mathcal{S}$ is the knowledge the MCSP has about the MCS scenario in time step t. The set $\mathcal{A}$ contains the possible actions the MCSP can take, i.e., the possible task allocation decisions. The set $\mathcal{P}$ contains probabilities $P(S_{t+1}|S_t, A_t)$ that the MCSP will observe state $S_{t+1} \in \mathcal{S}$ when it takes action $A_t \in \mathcal{A}$ in the state $S_t \in \mathcal{S}$. The reward set $\mathcal{R}$ contains the rewards $R_t \in \mathcal{R}$ the MCSP receives after taking action $A_t \in \mathcal{A}$ in state $S_t \in \mathcal{S}$. In our scenario, the state S_t is the collection of battery statuses $b_{k,t}$ of all reserved sensors K, the task size M_t and the number of required sensors U_t. It is defined as $S_t = \{b_{1,t}, b_{2,t}, \ldots, b_{K,t}, M_t, U_t\}$. Since the battery levels can take any value in the continuous range $[0, B_{\max}]$, there can be infinitely many possible states in the set $\mathcal{S}$. The action set $\mathcal{A}$ consists of all the task allocation decisions the MCSP can take. Since U_t sensors are chosen for the task execution t, with $1 \leq U_t \leq K$, the set $\mathcal{A}$ consist of $2^{|\mathcal{K}|}$ combinations, including an action to not allocate any sensor for the current task, or in other words, to not perform the task to save energy. As perfect knowledge about the environment is not available, the transition probability matrix $\mathcal{P}$ is assumed to be unknown. Finally, after selecting an action A_t, the MCSP receives a rewards $R_t \in \mathcal{R}$ according to (8).

Algorithm 1 Deep Q-network agent training

Require: Battery status b for all reserved sensors K, Task tuple $\langle M, \tau^{\text{dl}}, U \rangle$.

1: **for** each episode $i = 1, 2, \ldots, I$ **do**
2: Initialize DQN Agent critic $Q(S, A; \theta)$ with random parameter values θ and a target DQN critic with parameter values θ^t.
3: **for** each time step $t = 1, 2, \ldots, T$ **do**
4: For current state S_t, select a random action with probability ϵ, or select an action which has highest $Q(S_{t+1}|S_t, A_t)$ ▷ ϵ-greedy policy
5: Observe R_t and evaluate S_{t+1}. ▷ Constraints (4-7)
6: Store $\langle S_t, A_t, R_t, S_{t+1} \rangle$ in replay memory
7: Sample random L experience samples to form a mini-batch
8: Evaluate the loss function
9: Update θ, update ϵ using ϵ-decay rate
10: Update $\theta^t = \theta$ periodically for target network
11: **end for**
12: **end for**
13: **return** Trained DQN Agent with parameters $Q(S, A; \theta^t)$

An action selection policy π determines the relation between the state $S_t \in \mathcal{S}$ and an action $A_t \in \mathcal{A}$ taken by the agent as $A_t = \pi(S_t)$. Furthermore to evaluate how good or bad some policy π is, an action-value function $Q^\pi(S_t, A_t)$ is calculated. This function calculates the expected reward starting from state $S_t \in \mathcal{S}$, selecting action $A_t \in \mathcal{A}$ and following the same policy π until the end of the time horizon. There also exists an optimal policy π^* which guarantees an optimal action-value function $Q^*(S_t, A_t)$. Conversely, when an action $A_t \in \mathcal{A}$ maximizes the function $Q^*(S_t, A_t)$, it is an optimal action.

IV. Reinforcement Learning Solution

In our proposed approach, the MCSP implements a DQN to solve the task allocation problem. The use of a DQN is motivated by the infinitely large state space and the combinatorial but finite action space. The DQN agent, i.e., the MCSP, learns the task allocation strategy by interacting with the environment. From this interaction, the MCSP builds an estimate of the action value function $Q(S_t, A_t)$ and selects actions which yield the highest $Q(S_t, A_t)$. The general aim of the DQN agent is to maximize the expected long term return,

$$G = \lim_{T \longrightarrow \infty} \mathbb{E}\left[\sum_{t=1}^{T} \gamma^t R_t\right]. \tag{9}$$

Here, $\gamma \in (0, 1]$ is the discount factor that describes the exponentially decreasing weight given to future rewards. $\gamma \to 1$ indicates that the DQN agent prefers to achieve higher rewards in the future compared to lower values of γ that indicate a preference for immediate rewards.

The proposed algorithm is based on [16]. In a nutshell, our DQN works as follows: The DQN agent is trained over multiple instances of the finite time horizon. Each of these instances is called an episode. During the training phase, our proposed algorithm initializes a deep neural network for the estimation of a parameterized action-value function $\hat{Q}(S_t, A_t; \theta)$. This parameterized value function is used to select an action A_t in each time step. Through the interaction with the environment, the DQN tunes the parameters θ to

TABLE II
SIMULATION PARAMETERS

Parameter	Value
Total time steps T in time horizon T_{H}	3000 time steps
Duration of one time step $t = \tau^{\mathrm{int}}$	200 ms
Number of reserved sensors K	8
Sensor distances to MCSP $[d_{\min}, d_{\max}]$	$[200, 1000]$ m
Battery capacity $B_{\max}$	32 mW s
Maximum harvested energy $E^{\mathrm{harv}}_{\max}$ per t	10% of $B_{\max}$
Total Bandwidth W per sensor k	1 MHz
Noise power σ^2	10^{-16} W
Transmit power $p^{\mathrm{tx}}_{\max}$ of sensor k	100 mW
Channel gain $\lvert h_{k,t}\rvert^2$	$\sim d^{-3}$ (Urban scenario)
Sensing task throughput M	$[10 - 250]$ kbit
Deadline τ^{dl}	Uniform in $[\frac{\tau^{\mathrm{int}}}{2}, \tau^{\mathrm{int}}]$s
Required sensors U_t per task t	Uniform in $[1, K]$

improve the accuracy of $\hat{Q}(S_t, A_t; \theta)$ and converge it to the true value function $Q(S, A)$.

A summary of the proposed algorithm is presented in Alg. 1. In each episode i, a finite time horizon T_{H} is considered in the corresponding time steps of τ^{int}. In each time step t, the MCSP may explore a new action or exploit the so far best possible action A_t which maximizes $\hat{Q}(S_t, A_t; \theta)$ for the current state S_t. This policy is known as ϵ-greedy action selection. Each action A_t selected by the MCSP in state S_t returns a reward R_t based on the objective function in (8). In simple terms, the MCSP gets a reward $R_t > 0$ if the current task is completed, otherwise zero. The reward R_t linearly depends on the values of the task tuple $\left\langle M'_t, \tau^{\mathrm{dl}'}_t, U'_t \right\rangle$. The proposed algorithm uses a replay memory which stores the experiences of the MCSP in the form of $\langle S_t, A_t, R_t, S_{t+1} \rangle$. This memory helps the MCSP to tune the parameters θ to improve the estimate $\hat{Q}(S_t, A_t; \theta)$. The estimate $\hat{Q}(S_t, A_t; \theta)$ for the current state S_t and all actions $A_t \in \mathcal{A}$ is calculated using a subset of samples from the replay memory, known as a mini-batch of size C. A loss function, i.e., Huber loss function, is calculated over this subset of experiences w.r.t. θ. The parameters θ are then tuned such that the loss function is minimized. At the end of the time step t, the MCSP updates the neural network parameters. Since the MCSP improves $\hat{Q}(S_t, A_t; \theta)$ based on previous estimates, stability needs to be maintained. To achieve this, two identical sets of neural networks are used. One network with parameters θ is tuned in every time step, whereas another neural network, also known as a target network, with parameters θ^t is updated from the tuned network θ periodically until convergence is achieved.

V. NUMERICAL EVALUATION

In this section, we present simulation results to evaluate the performance of our proposed DQN-based approach by comparing it with reference schemes. These results are generated by taking an average over $I = 1000$ independent realizations. In each realization, we consider $T = 3000$ time steps. Each task size M_t in time step t is an element of a discrete set of task sizes $\mathcal{M} = \{M_{t,1}, \ldots, M_{t,5}\}$ such that $M_{t,1} < \ldots < M_{t,5}$. Moreover, all of these task sizes are equiprobable. The task arrival probability λ is set to $\frac{|\mathcal{M}|}{|\mathcal{M}|+1}$ to include the fact that in some time steps, no task arrives at the MCSP. The sensors are located in an area such that the maximum distance between a sensor and the MCSP is 1 km. The channel between sensor k and the MCSP is modelled as Rayleigh fading with path loss exponent of three. The sensors can move freely in the area at an average speed of 5 km/h. We set the parameters χ, ω, and ψ from (3) to one to encourage fairness in task completion. Table II provides a summary of simulation parameters.

To train the DQN agent, a neural network with two hidden layers is used for the $Q(S, A)$ estimation. Additionally, a learning rate $\alpha = 3 \times 10^{-3}$ is used. In order to learn from a sufficiently large data sample set, a replay memory of size 5×10^4 is used out of which a mini batch of $C = 256$ samples of experiences is considered for the $Q(S, A)$ estimation.
We consider following reference schemes for comparison:
Optimal task allocation: In this approach, we assume that the MCSP has perfect non-causal knowledge about the channel coefficients, the amounts of harvested energy and the task to be performed for the complete time horizon T_{H}.
Myopically optimal task allocation: This approach assumes the MCSP has perfect causal knowledge of the channel coefficients and the amounts of harvested energy. The myopically optimal allocation is found by iteratively selecting the sensors with the best channel coefficients $h_{k,t}$ and adequate battery statuses $b_{k,t}$ for the current task t to ensure task completion.
Random task allocation: This strategy provides the lower bound for the performance since it participates in every task and randomly allocates the sensors for execution.

The average weighted sum of completed tasks in a finite time horizon T_{H} for all approaches is shown in Fig. 2. The optimal task allocation strategy performs the highest average weighted sum of completed tasks, i.e. 1997 tasks, and sets the upper bound of the performance. In comparison, our proposed algorithm performs near-optimally by completing 1843 tasks which are approx. 92.2% of the optimal performance without the strict requirement of non-causal knowledge about the channel coefficients, the amounts of harvested energy and the tasks to be performed. Our approach performs at least 12% better than the myopically optimal task allocation scheme which requires perfect causal knowledge about the channel coefficients and the amounts of harvested energy. The reason for this is that our proposed approach learns to make task allocation decisions considering their future consequences. For example, instead of always allocating the task to the sensors with best channel coefficients, like in the case of the myopically optimal strategy, our proposed algorithm allocates the task to sensors which can fulfil the requirements despite of not having the best channel conditions. By doing so, we ensure that future, and potentially more demanding tasks, can also be performed. As a consequence, our proposed approach performs consistently well especially in case of limited resources. For the random task allocation strategy there is no guarantee that assigned sensors will fulfil the task requirements. Therefore, the number of completed tasks is low. Our proposed approach outperforms the random allocation by performing at least 80% more tasks.

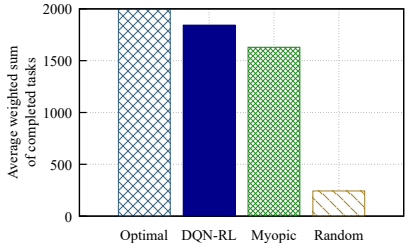

Fig. 2. Average weighted sum of completed tasks for all approaches

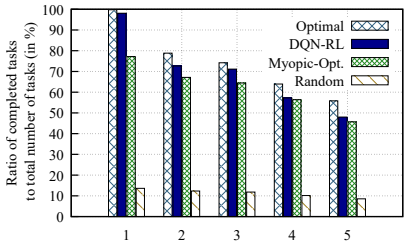

Fig. 3. Average weighted sum of completed tasks (%) vs. Task size

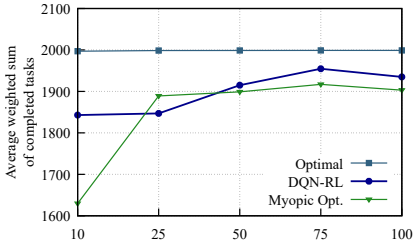

Fig. 4. Average weighted sum of completed tasks vs. $E_{\max}^{\mathrm{harv}}$ in % of $B_{\max}$

To analyze the performance of all the considered approaches w.r.t. the task size we compare the task completion rate in percentage, as shown in Fig. 3. This analysis highlights the importance of the weight V introduced in (3). As the task size increases, the sensors require more time $\tau_{k,t}^{\mathrm{tx}}$ to transmit the result back to MCSP which makes the fulfillment of the deadline constraint (4) harder. Therefore a downward trend is seen in the task completion as the task size increases. Even in these conditions, our proposed approach outperforms the myopically optimal task allocation scheme at least by 23% and 3% in the completion of the $M_{t,1}$ and $M_{t,5}$, respectively.

To study the effect of the amount of harvested energy per time step t on the performance, we repeat our experiment for different values of maximum harvested energy $E_{\max}^{\mathrm{harv}}$. The results are shown in Fig. 4. As $E_{\max}^{\mathrm{harv}}$ increases, the probability that the sensors will have enough energy to perform the tasks increases. Due to this, all the approaches perform better with an increase in $E_{\max}^{\mathrm{harv}}$. In Fig. 4, we omit the random allocation because it performs consistently worse than the other schemes. The performance of the myopically optimal task allocation strategy improves drastically by 16.1% when $E_{\max}^{\mathrm{harv}} = 0.25B_{\max}$. This is because the MCSP no longer needs to reserve the resources for the future. As $E_{\max}^{\mathrm{harv}}$ increases further, the performance of our proposed approach is only 2.2% below the performance of the optimal approach. The performance of our proposed algorithm outperforms the myopically optimal strategy throughout and tends to converge to the optimal performance as $E_{\max}^{\mathrm{harv}}$ increases.

VI. Conclusion

In this work, we study the task allocation problem in an energy harvesting mobile crowd-sensing scenario to maximize the average weighted sum of completed tasks in a finite time horizon. The mobile crowd-sensing platform (MCSP) makes task allocation decisions, i.e., it decides whether or not to perform a task depending on the available resources, and if the task is to be performed, assigns it to suitable sensors. The optimal task allocation strategy in this scenario requires perfect non-causal knowledge of the channel coefficients, the amounts of harvested energy and the sensing tasks to be performed. In realistic scenarios, this knowledge is unavailable at the MCSP. We proposed a DQN approach to solve the allocation problem without this strict requirement. According to the simulation results, our proposed approach performs only 7.8% lower than the optimal task allocation strategy. Moreover, our proposed approach outperforms the myopically optimal task allocation strategy by 12%; and the random allocation strategy by 80%.

References

[1] R. K. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: current state and future challenges," *IEEE Commun. Mag.*, vol. 49, no. 11, pp. 32–39, 2011.

[2] J. An, X. Gui, J. Yang, S. Yu, and X. He, "Mobile crowd sensing for internet of things: A credible crowdsourcing model in mobile-sense service," in *IEEE Int. Conf. on Multimedia Big Data*, 2015, pp. 92–99.

[3] H. Ma, D. Zhao, and P. Yuan, "Opportunities in mobile crowd sensing," *IEEE Commun. Mag.*, vol. 52, no. 8, pp. 29–35, 2014.

[4] H. To, L. Fan, L. Tran, and C. Shahabi, "Real-time task assignment in hyperlocal spatial crowdsourcing under budget constraints," in *IEEE Int. Conf. on Pervasive Comput. and Commun.*, 2016, pp. 1–8.

[5] X. Li and Q. Zhu, "Social incentive mechanism based multi-user sensing time optimization in co-operative spectrum sensing with mobile crowd sensing," *Sensors*, vol. 18, no. 1, 2018.

[6] R. Pryss, J. Schobel, and M. Reichert, "Requirements for a flexible and generic API enabling mobile crowdsensing mhealth applications," in *Int. Workshop on Requirements Engineering for Self-Adaptive, Collaborative, and Cyber Physical Systems (RESACS)*, 2018, pp. 24–31.

[7] Z. Yu, H. Ma, B. Guo, and Z. Yang, "Crowdsensing 2.0," *Commun. ACM*, vol. 64, no. 11, p. 76–80, Oct 2021. [Online]. Available: https://doi.org/10.1145/3481621

[8] X. Cui, J. Zhang, H. Zhou, and C. Deng, "Powerpool: Multi-source ambient energy harvesting," in *6th Int. Conf. on Big Data Comput. and Commun. (BIGCOM)*, 2020, pp. 86–90.

[9] M. M. Sandhu, K. Geissdoerfer, S. Khalifa, R. Jurdak, M. Portmann, and B. Kusy, "Towards energy positive sensing using kinetic energy harvesters," in *IEEE Int. Conf. on Pervasive Comput. and Commun. (PerCom)*, 2020, pp. 1–10.

[10] M. M. Sandhu, S. Khalifa, R. Jurdak, and M. Portmann, "Task scheduling for energy-harvesting-based iot: A survey and critical analysis," *IEEE IoT Journal*, vol. 8, no. 18, pp. 13 825–13 848, 2021.

[11] D. Zhang, H. Xiong, L. Wang, and G. Chen, "Crowdrecruiter: Selecting participants for piggyback crowdsensing under probabilistic coverage constraint," in *Proc. ACM Int. Joint Conf. on Pervasive and Ubiquitous Comput.*, New York, 2014.

[12] X. Tao and W. Song, "Efficient task allocation for mobile crowd sensing based on evolutionary computing," in *2018 IEEE Int. Conf. on Internet of Things (iThings) and IEEE Green Comput. and Commun. (GreenCom) and IEEE Cyber, Physical and Social Comput. (CPSCom) and IEEE Smart Data (SmartData)*, 2018, pp. 374–380.

[13] ——, "Task allocation for mobile crowdsensing with deep reinforcement learning," in *2020 IEEE Wireless Commun. and Networking Conf. (WCNC)*, 2020, pp. 1–7.

[14] C. H. Liu, B. Zhang, X. Su, J. Ma, W. Wang, and K. K. Leung, "Energy-aware participant selection for smartphone-enabled mobile crowd sensing," *IEEE Systems Journal*, vol. 11, no. 3, pp. 1435–1446, 2017.

[15] A. Ortiz, T. Weber, and A. Klein, "A two-layer reinforcement learning solution for energy harvesting data dissemination scenarios," in *2018 IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP)*, 2018, pp. 6648–6652.

[16] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," 2013. [Online]. Available: https://arxiv.org/abs/1312.5602