Online Optimization of Wireless Sensors Selection over an Unknown Stochastic Environment

Muhammad Anjum Qureshi, Wardah Sarmad, Hira Noor, Ali Hassan Mirza
Department of Electrical and Electronics Engineering
Bilkent University, Ankara, Turkey
{qureshi, sarmad, noor, mirza}@ee.bilkent.edu.tr

***Abstract*—Wireless communication is considered to be more challenging than the typical wired communication due to unpredictable channel conditions. In this paper, we target coverage area problem, where a group of sensors is selected from a set of sensors placed in a particular area to maximize the coverage provided to that area. The constraints to this optimization are the battery power of the sensor and number of sensors that are active at a given time. We consider a variant of the coverage related to a particular sensor, where coverage is considered to be an unknown stochastic variable, and hence, we need to learn the best subset of sensors in real time. We propose an online combinatorial optimization algorithm based on multi-armed bandits framework that learns the expected best subset of sensors, and the regret of the proposed online algorithm is sub-linear in time. The achieved performance proves the robustness and effectiveness of the proposed online algorithm in wireless sensor selection over an unknown stochastic environment.**

***Keywords*—bandits, wireless sensors, super sensor, interference.**

I. Introduction

The low-cost and low-power wireless sensors are manufactured these days due to rapid advances in integrated circuits. A wireless sensor is composed of sensing unit, storage and data processing units along with the battery [1]. These sensors are equipped with different sensing abilities based on the application. In addition to sensing, these sensors are also capable of processing the data. Coverage requirement of an area is defined as either the full coverage of the area or a percentage threshold that is required to be attained. In order to cover a large area, usually a larger sensor network is formed using multiple sensors placed in different locations. Considering the power consumption, the task is to prolong the network lifetime via concurrently using the sensor and resting the sensor. A wireless sensor in its working state provides all the functions and consumes power at the same time, while in the sleeping state it only receives the control signal and consumes less power. The advantage of the larger sensor network is that it doesn't require all sensors to be in working state to provide the required coverage, and hence, few sensors are in working state and remaining sensors are in sleeping state [2], [3], [4].

In this paper, we target to enhance the network lifetime via considering the limited lifetime of the sensor batteries and number of active wireless sensors to provide the target coverage. We assume that the coverage of the wireless sensors is drawn from an unknown probability distribution and the current instantaneous coverage realization is available to the control unit via mobile sensing devices available in the area. This information which is the reward of the selected sensor is provided to the proposed algorithm which updates its parameters to learn the minimal number of sensors that can achieve the required coverage. The algorithm also takes into account the remaining power of the sensor while selecting the best subset of the sensors.

Multi-armed bandit (MAB) is a fundamental problem in machine learning, where there are n arms with unknown rewards [5], [6]. The objective is to maximize the total accumulated reward up to time t, achieved via Exploration-Exploitation trade-off. Exploration-Exploitation trade-off is a balance between exploring the unexplored arms and exploiting the arm providing a best-expected reward. The main objective is to minimize the *Regret*, which is the difference between accumulated expected reward of optimal arm and arms selected by the algorithm. It is shown in the literature that $O(logT)$ regret is achieved asymptotically using upper confidence bounds (UCBs).

As an extension to this, *combinatorial multi-armed bandits* target real-world applications where the combination of arms is optimized with usually linear rewards [7], [8], [9], [10]. The combination of selected arms is referred to as *super arm*. The target of combinatorial algorithm is to select the best super arm based on the previous observations to maximize the overall expected reward. Recent work in combinatorial bandits extends the generic framework to a major class of non-linear rewards [11]. When rewards of the individual arms are revealed along with the reward of super arm, this feedback is known as *semi-bandit feedback* [10]. The reward of the super arm is referred to as *global reward*, which can be a non-linear function of these individual rewards and needs to satisfy two mild assumptions of monotonicity and bounded smoothness.

Main contributions of this paper are:

- We propose a variant of wireless sensors coverage problem by introducing randomness in the sensing coverage of the sensor. We propose an online combinatorial optimization algorithm that learns the best combination of wireless sensors in the stochastic environment.
- We analyze two different scenarios: firstly, we assume that wireless sensors do not interfere with each other,

978-1-5386-1501-0/18/$31.00

and secondly, we analyze the case when sensors interfere the sensing of the other sensors.

II. Problem Formulation

We denote the set of sensors $S = (\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_N) \in \mathcal{S}$ as a *super sensor*, where $\boldsymbol{s}_l, l \in \{1, \ldots, N\}$ denote a particular sensor in a super sensor. Let $N = |S|$ is number of selected sensors and L is total number of sensors in $\mathcal{S}$. In each round t, a super sensor is selected and coverage reward is revealed, referred to as *global reward*. We say it a semi-bandit feedback, since rewards of individual sensors as well as the reward of super sensor are revealed. The coverage of a particular sensor $\boldsymbol{s}_l$ is associated with a set of random variables $X_{\boldsymbol{s}_l,t}$ for $t \geq 1$, and the set of these random variables is denoted as X_t. These random variables represent the random coverage of corresponding sensor. Let $T_{\boldsymbol{s}_l,t}$ represents the number of times sensor $\boldsymbol{s}_l$ is selected at round t.

Let $R_t(S)$ represents the reward of the super sensor. We consider rewards with the bounded support of $[0, 1]$, for a particular sensor it is the ratio of the coverage provided by the sensor to full coverage of the area, and for super sensor it is the ratio of the summation of coverage of underlying sensors to the number of underlying sensors. We denote the expected individual rewards vector as $\boldsymbol{\mu} = \{\mu_1, \ldots, \mu_N\}$ and the expected reward of the super sensor as $r_{\boldsymbol{\mu}}(S) = \mathbb{E}[R_t(S)]$, which is the linear combination of the expected rewards of the underlying sensors. The objective is to satisfy the coverage constraint with minimum possible number of sensors. Let S_t is the super sensor selected by the algorithm at round t and $opt_{\boldsymbol{\mu}} = \max_{S \in \mathcal{S}} r_{\boldsymbol{\mu}}(S), S^* = \arg\max_{S \in \mathcal{S}} r_{\boldsymbol{\mu}}(S)$. As optimal super sensor, S^* may be computationally hard to find, we allow oracle to approximate the optimal super arm as $\alpha \cdot opt_{\boldsymbol{\mu}}$, and to have success probability β.

To apply the general combinatorial bandits, two mild assumptions of monotonicity and bounded smoothness [9] are required to be taken on $r_{\boldsymbol{\mu}}(S)$.

Assumption 1. *The expected reward of a particular super sensor S is monotonically non-decreasing with respect to the expected reward vector i.e., if $\forall \boldsymbol{s} \in S, \mu_{\boldsymbol{s}} \leq \mu'_{\boldsymbol{s}}$, then $r_{\boldsymbol{\mu}}(S) \leq r_{\boldsymbol{\mu}'}(S)$.*

Assumption 2. *There exists a strictly increasing and invertible function $f(\cdot)$, called bounded smoothness functions, such that for two expected reward vectors $\boldsymbol{\mu}$ and $\boldsymbol{\mu}'$, we have $|r_{\boldsymbol{\mu}}(S) - r_{\boldsymbol{\mu}'}(S)| \leq f(\Lambda)$ if $\max_{\boldsymbol{a} \in S} |\mu_{\boldsymbol{a}} - \mu'_{\boldsymbol{a}}| \leq \Lambda$.*

The regret of learning is written as

$$Reg(T) = T \cdot \alpha \cdot \beta \cdot opt_{\boldsymbol{\mu}} - \mathbb{E}\Big[\sum_{t=1}^{T} r_{\boldsymbol{\mu}}(S_t)\Big]. \tag{1}$$

III. The proposed System

In this section, we propose an online *coverage optimization scheme* (COS) for wireless sensors. We consider two different cases: first we study the non-interference case, where sensors are placed far from each other such that no interference occurs in sensing, and targets small sensor networks. In the second scenario, we consider larger sensor networks where sensors are placed close to each other such that interference occurs in the sensing ranges of sensors. We consider the interference among the chosen sensors as a disadvantage while optimizing the sensor subset, although redundancy is achieved via selecting the nearby overlapping sensing region sensors, but this is not the primary objective. We focus to select the smallest best possible subset to satisfy the coverage constraint. Let k^* denote the minimum number of sensors to provide the required coverage.

Algorithm 1 COS (Coverage Optimization Scheme)

Input: Action Space $\mathcal{S}$, Required Coverage C, p_{TH}
`Initialize`: $T_{\boldsymbol{s}} = 0, \hat{\mu}_{\boldsymbol{s}} = 0, p_{\boldsymbol{s}} = 1, \forall \boldsymbol{s} \in \mathcal{S}$, $L = |\mathcal{S}|$
For each sensor $\boldsymbol{s}$, perform an arbitrary super sensor $S \in \mathcal{S} \mid \boldsymbol{s} \in S$. Update $T_{\boldsymbol{s}}$ and $\hat{\mu}_{\boldsymbol{s}}$
$t \leftarrow L$
while $t \geq L$
  $t \leftarrow t + 1$
  For each action $\boldsymbol{s}$, $\bar{\mu}_{\boldsymbol{s}} = \hat{\mu}_{\boldsymbol{s}} + \sqrt{1.5 \frac{\ln t}{T_{\boldsymbol{s}}}}$
  $S = \textit{Oracle}(\bar{\mu}_1, \ldots, \bar{\mu}_L, \hat{\mu}_1, \ldots, \hat{\mu}_L)$
  Perform S and update $T_{\boldsymbol{s}}$, $\hat{\mu}_{\boldsymbol{s}}$ and $p_{\boldsymbol{s}}$
  if $p_{\boldsymbol{s}} < p_{TH}$ $\forall \boldsymbol{s} \in \mathcal{S}$
  //Sensor/Battery Replacement Request
  end if
  if new(sensor/battery) //service provided for $\boldsymbol{s}_l$
  $T_{\boldsymbol{s}_l} = 0, \hat{\mu}_{\boldsymbol{s}_l} = 0, p_{\boldsymbol{s}_l} = 1$
  end if
end while

The pseudo-code of the proposed method is shown in Algorithm 1, and the offline optimization is handled by the offline oracle. In the initialization step, a random super sensor is selected for each of the available sensors. Then by using *upper confidence bound* (UCB) method, upper bounds over the estimated rewards are calculated. These upper bounds are provided to an offline oracle, which performs offline optimization over given rewards. We propose a centralized solution, where location of all sensors and measured coverage is provided to the learning algorithm. The learning algorithm then generates control signals to the selected sensors in the subset and generates sleeping state commands to the remaining sensors. Since dense measurement points can provide an exact estimate compared to fewer measurement points network, the accuracy of the learning algorithm is directly dependent on the placement of measurement points.

The total lifetime of the sensor network is also an important part of the proposed system, and is provided with a threshold value p_{TH} for the remaining scaled battery power $p_{\boldsymbol{s}} \in [0, 1]$ for a particular sensor $\boldsymbol{s}$. If p_{TH} is reached, the proposed algorithm generates a service request to either replace the battery of the sensor or replace the sensor itself. The sensors over the provided threshold are not selected by the offline oracle until they are serviced. Since new placed sensor or repaired sensors may have different characteristics, we reinitialize the counters

Algorithm 2 Oracle for non-interference case

Input: Sensor Reward UCBs $(\bar{\mu}_1, \bar{\mu}_2, \ldots, \bar{\mu}_L)$, C, p_{TH}, $p_{\boldsymbol{s}}, \forall \boldsymbol{s} \in \mathcal{S}$, and Estimated Rewards $(\hat{\mu}_1, \ldots, \hat{\mu}_L)$
$\texttt{Sort}: [values, ind] = sort(\bar{\mu}_1, \bar{\mu}_2, \ldots, \bar{\mu}_L)$
S' is the array of estimated rewards with indexes ind i.e., $S' = \{\hat{\mu}_{ind(1)}, \ldots, \hat{\mu}_{ind(L)}\}$
$C' = 0, n = 1, n' = 1, S = \{\}$
while $C' < C$
  if $p_{S'(n')} \geq p_{TH}$
  $S(n) \leftarrow S'(n')$
  $C' \leftarrow C' + \hat{\mu}_{S(n)}$
  $n \leftarrow n + 1$
  end if
  $n' \leftarrow n' + 1$
end while
Return: S

Algorithm 3 Oracle for interference case

Input: Sensor Reward UCBs $(\bar{\mu}_1, \bar{\mu}_2, \ldots, \bar{\mu}_L)$, C, p_{TH}, $p_{\boldsymbol{s}}, \forall \boldsymbol{s} \in \mathcal{S}$, Estimated Rewards $(\hat{\mu}_1, \ldots, \hat{\mu}_L)$ and $\Sigma_{\mathcal{S},\mathcal{S}}$
$\texttt{Sort}: [values, ind] = sort(\bar{\mu}_1, \bar{\mu}_2, \ldots, \bar{\mu}_L)$
S' is the array of estimated rewards with indexes ind i.e., $S' = \{\hat{\mu}_{ind(1)}, \ldots, \hat{\mu}_{ind(L)}\}$
$C' = 0, n = 1, n' = 1, S = \{\}$
while $C' < C$
  if $p_{S'(n')} \geq p_{TH}$
    if $isempty(S)$
    $S(n) \leftarrow S'(n')$
    $C' \leftarrow C' + \hat{\mu}_{S(n)}$
    $n \leftarrow n + 1$
    elseif $\big((\hat{\mu}_{S'(n')} - \Sigma_{S'(n'),s} \geq \hat{\mu}_k - \Sigma_{k,s})$ or $(\Sigma_{S'(n'),s} = 0)\big), \forall k \in \{S' \setminus (S + S'(n'))\}, \forall s \in S$
    $S(n) \leftarrow S'(n')$
    $C' \leftarrow C' + \hat{\mu}_{S(n)}$
    $n \leftarrow n + 1$
    end if
  end if
  $n' \leftarrow n' + 1$
end while
Return: S

and expected reward for the replaced sensor.

A. Case I: Sensors with no interference

We considered a relatively simple scenario, where the placed sensors are considered to be far from each other such that there is no interference among the sensors. In this scenario, the offline oracle sorts the UCBs and finds the minimum number of sensors whose sum of the expected rewards are equal or more than the required coverage by adding the sorted sensors one by one in the super sensor. As shown in the Algorithm 2, offline oracle first sorts the UCB rewards of the sensors, and after sorting process, the sensor with best estimated reward is included in the super sensor as its first element. Then, the reward of the super sensor which is a summation of expected rewards of its elements is compared with the target coverage. If target coverage is not achieved, the sensor with highest estimated reward among the remaining sensors is selected and included in the super sensor as its second element. If target coverage is achieved, the subset is provided to the main algorithm, otherwise the process of selection continues.

It is worth mentioning that the sorting is performed over UCBs, which in addition to expected rewards considers the number of times sensor is selected to cater the exploration-exploitation trade-off. After sorting the UCBs, only expected rewards of the sensors are used to decide the selection of sensor in the super sensor. Furthermore, before adding the sensor as an element of the super sensor, the proposed algorithm checks the remaining power of the sensor, if it is below the given threshold p_{TH}, the candidate sensor is not included in the super sensor.

B. Case II: Sensors with interference

In this scenario, we consider a large sensor network where there are plenty of sensors and they interfere with each other. The offline oracle just by sorting cannot find the best subset, as there are dependencies among the sensors. The additional input is a correlation matrix $\Sigma_{\mathcal{S},\mathcal{S}}$, which provides the expected interference ratio among placed sensors. The diagonal entries of the matrix are set to zero, and these represent the self-interference. The offline oracle initially includes the estimated best reward sensor into the subset sensor as its first element. Before adding the second element, the dependencies among the candidate sensor and elements of the super sensor are compared with the remaining sensors i.e., $\Sigma_{S'(n'),S}$ provides the dependencies of the candidate sensor $S'(n')$ with elements of the super sensor S. If difference of the expected reward and dependencies is still greater than the difference of remaining sensors in the set $\{S' \setminus (S + S'(n'))\}$ (members of S' that are not in the super sensor S and not the element $S'(n')$), the candidate sensor is placed in the super sensor.

On the other hand, if reward excluding the dependencies is lower than any of the remaining sensors, the next sensor in the list is considered as the candidate sensor. The process is terminated if the coverage threshold is reached or there are no more sensors left to be explored. The pseudo-code for this scenario is provided in Algorithm 3. It is worth mentioning that no assumption is made regarding the sensing range of the sensors. The sensors are not required to have same sensing range and are allowed to have different sensing ranges, as the sensing is provided by the measurement points to the algorithm.

IV. Illustrative Results

In this section, performance of the proposed algorithm is evaluated and results are provided for both scenarios discussed in the previous section. The obtained results provide the fact that the expected regret of the proposed algorithm is sublinear in time. We set $T = 10^4$ for both experiments.

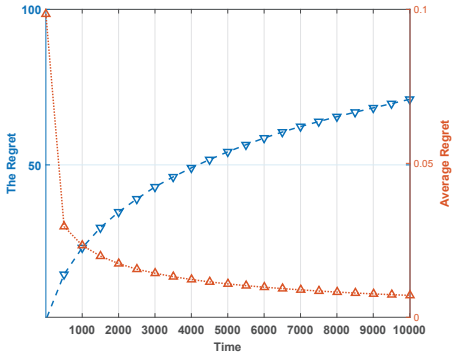

Fig. 1: The Regret and time-average Regret for a network of 6 non-interfering sensors aimed to achieve the net coverage area of 0.70

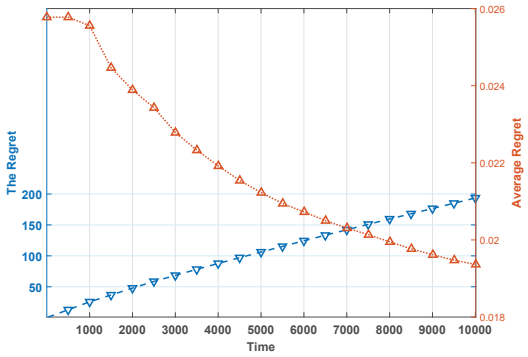

Fig. 2: The Regret and time-average Regret for a network of 6 interfering sensors aimed to achieve the net coverage area of 0.70

A. Experiment 1: No interference (Fig. 1)

In the first experiment, we consider a network of 6 sensors. The sensor coverage is represented by Gaussian random variable with mean values as $\{0.10, 0.30, 0.20, 0.10, 0.05, 0.25\}$ and variance of 0.50. We set the cumulative target coverage area to be 0.70. If we know the true values of the expected means, we can easily find that best possible super sensor which contains three sensors with mean values $\{0.30, 0.25, 0.20\}$ and the coverage area is obtained as 0.75 with $k^* = 3$ and the oracle expected reward is 0.25. But as we observe data in the stochastic environment and need to estimate the expected values, we perform COS on the given dataset and obtained the regret shown in Fig. 1. The proposed algorithm concurrently explores and exploits the sensors to obtain the best super sensor. We perform 100 random experiments and results are obtained by averaging over these runs. The obtained regret is sub-linear in time T. We assume $\alpha = \beta = 1$ for this experiment.

B. Experiment 2: Selection with interference (Fig. 2)

In this experiment, we again considered a network of 6 sensors. The sensor coverage is represented by Gaussian random variable with mean values as $\{0.20, 0.30, 0.25, 0.10, 0.05, 0.25\}$ and variance of 0.5. We set the cumulative target coverage area to be 0.7. If we know the true values and there is no interference among the sensors, we can calculate the best possible super sensor which contains three sensors with mean values $\{0.30, 0.25, 0.25\}$ and the coverage area is obtained as 0.80. For this experiment, we additionally provide the information that sensor 2 and 3 are very close to each other and the expected interference for both sensors is 90% of their sensing region. Hence, the super sensor in this modified scenario is $\{0.20, 0.30, 0.25\}$ (Sensor 1,2 and 6) and the coverage area is obtained as 0.75 with $k^* = 3$ and the oracle expected reward is 0.25. We perform COS on the given dataset and obtained the regret shown in Fig. 2. Similar to experiment 1, we perform 100 random experiments and results are obtained by averaging over these runs. Furthermore, for this experiment we allow offline oracle to have success probability and estimation error i.e., $\alpha = \beta = 0.90$.

V. Conclusion

In this paper, we propose an online combinatorial optimization algorithm for wireless sensors to provide desired coverage in a particular area. We consider the case when the sensing is random in nature, and hence, we propose combinatorial multi-armed bandits framework to provide online learning solution. The proposed algorithms are sub-linear in time T.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications magazine*, vol. 40, no. 8, pp. 102–114, 2002.

[2] H. Chen and H. Wu, "Selecting working sensors in wireless sensor networks," in *Combinatorial Optimization in Communication Networks*, pp. 189–206, Springer, 2006.

[3] H. Zhang and J. C. Hou, "Maintaining sensing coverage and connectivity in large sensor networks," *Ad Hoc & Sensor Wireless Networks*, vol. 1, no. 1-2, pp. 89–124, 2005.

[4] H. Chen, H. Wu, and N.-F. Tzeng, "Grid-based approach for working node selection in wireless sensor networks," in *Communications, 2004 IEEE International Conference on*, vol. 6, pp. 3673–3678, IEEE, 2004.

[5] T. L. Lai and H. Robbins, "Asymptotically efficient adaptive allocation rules," *Advances in applied mathematics*, vol. 6, no. 1, pp. 4–22, 1985.

[6] R. Agrawal, "Sample mean based index policies by o (log n) regret for the multi-armed bandit problem," *Advances in Applied Probability*, vol. 27, no. 4, pp. 1054–1078, 1995.

[7] Y. Gai, B. Krishnamachari, and R. Jain, "Learning multiuser channel allocations in cognitive radio networks: A combinatorial multi-armed bandit formulation," in *New Frontiers in Dynamic Spectrum, 2010 IEEE Symposium on*, pp. 1–9, IEEE, 2010.

[8] Y. Gai, B. Krishnamachari, and R. Jain, "Combinatorial network optimization with unknown variables: Multi-armed bandits with linear rewards and individual observations," *IEEE/ACM Transactions on Networking (TON)*, vol. 20, no. 5, pp. 1466–1478, 2012.

[9] W. Chen, Y. Wang, and Y. Yuan, "Combinatorial multi-armed bandit: General framework and applications," in *International Conference on Machine Learning*, pp. 151–159, 2013.

[10] B. Kveton, Z. Wen, A. Ashkan, and C. Szepesvari, "Tight regret bounds for stochastic combinatorial semi-bandits," in *Artificial Intelligence and Statistics*, pp. 535–543, 2015.

[11] W. Chen, W. Hu, F. Li, J. Li, Y. Liu, and P. Lu, "Combinatorial multi-armed bandit with general reward functions," in *Advances in Neural Information Processing Systems*, pp. 1659–1667, 2016.