Remote Tracking of Dynamic Sources under Sublinear Communication Costs

Jihyeon Yun
Computer Science and Engineering
Korea University, Korea
Email: jihyeonyun@korea.ac.kr

Atilla Eryilmaz
Electrical and Computer Engineering
The Ohio State University, USA
Email: eryilmaz.2@osu.edu

Changhee Joo
Computer Science and Engineering
Korea University, Korea
Email: changhee@korea.ac.kr

***Abstract*—We study the remote monitoring of multiple sensors with evolving states following a Wiener Process under communication cost. We assume that the communication cost is sublinear such that the cost decreases with the number of simultaneous state updates. Such sublinear structures emerge in various settings, such as frame aggregation, and give rise to interesting unexplored tradeoffs between: updating a smaller subset of the processes earlier at a higher cost-per-process; and updating a larger subset of them later at a lower cost-per-process. We attack this problem by first providing two competitive benchmark strategies of All-at-once and Multi-threshold policies. Then, we propose a novel strategy of MAX-k policy that not only includes the two benchmark threshold-based policies as special cases, but also improves over them by better exploiting the aforementioned tradeoff. Further, we develop the GPSO optimization technique to develop an online learning algorithm that adaptively optimizes the parameters of MAX-k policy. We demonstrate that the proposed scheme outperforms the well-known online learning algorithm based on UCB index.**

I. Introduction

In increasingly many domains of growing importance, we encounter the need for keeping track of many independently evolving processes at a remote location. A common theme in such environment is the need for efficiently transferring multiple different evolving states from a transmitter to a receiver over a communication channel so that their states can be closely tracked at the receiver, while at the same time maintaining low communication costs. In this paper, we attack this generic problem in the key case of N independent Wiener Processes describing the source dynamics.

This problem is closely related to the Age of Information (AoI) optimization, which schedules transmissions to optimize information freshness between transmitters and receivers. The information freshness can be measured by AoI metric which indicates the elapsed time from the latest transmission. For one transmitter-receiver pair, a scheduling policy that manages arrived samples and decides which packet is transmitted to the receiver has been studied to minimize AoI in [1]. For multiple transmitter-receiver pairs, overall information age is minimized considering set of links that share a common channel in [2]. In [3], optimal scheduling policy to minimize AoI in multi-source system with random delay has been investigated under a constraint that only one source can be updated at a time.

In related works in this domain, a remote estimation system that considers average Mean Square Error (MSE) over stochastic process is analyzed with different constraints in [4]–[11]. A joint optimization problem of scheduling and remote estimation with a communication cost is formulated in [4] over finite time horizon. An energy harvesting sensor is considered in [5] and the number of transmissions is constrained in [6]. It is further extended to the case of noise channel in [7]. Over an infinite time horizon, the cost minimization in remote estimation system is considered under a packet drop channel in [8], and under a communication cost in [9]. Specifically, in [10], one-dimensional Wiener Process is considered to minimize MSE under a sampling frequency constraint over an infinite-time horizon. Further, cost minimization over multi-dimensional Wiener Processes is considered in [11] with constraint on frequency of communication.

In our work, we focus on *the remote tracking of multi-dimensional Wiener Processes when the communication cost function takes a sublinear form.* Such a sublinear structure necessitates the balancing of interesting new tradeoffs between: updating the current subset of the processes now but at a higher cost per process; and updating a larger subset of the processes later but at a lower cost per process. We attack this problem by first discussing competing benchmark strategies (All-at-once and Multi-threshold). Then we propose a novel strategy, called MAX-k policy that aims to optimize the aforementioned tradeoff, and discuss its gains over the benchmarks. The basic MAX-k policy design requires the availability of the time-averaged mean-squared-error and the communication cost functions, which are often unknown (and intractable to obtain analytically). This motivates us next to study a variation to MAX-k that releases this assumption so that the policy can solve for its optimal parameter selection by learning and adapting to its observations. To that end, we develop a learning mechanism, called MAX-k-GPSO policy, which evaluates potential solutions and updates them toward

C. Joo is the corresponding author.

The work of J. Yun and C. Joo is supported in part by the NRF grant funded by the Korea government (MSIT) (No. NRF-2017K1A3A1A19070720 and No. NRF-2017R1E1A1A03070524), and in part by a Korea University Grant.

The work of A. Eryilmaz is supported by the ONR Grant N00014-19-1-2621; NSF grants: CNS-NeTS-1717045, CNS-ICN-WEN-1719371, CNS-SpecEES-1824337, CNS-NeTS-2007231; and the DTRA grant: HDTRA1-18-1-0050.

the optimal solution. We demonstrate that this new policy can converge significantly faster than its UCB-based counterpart.

The remainder of the paper is organized as follows. The system model is described in Section II. We describe two different threshold-based policies and develop a generalized policy, MAX-k, and empirically observe their performance in Section III. Then, the algorithm to optimize parameters in MAX-k policy, MAX-k-GPSO, is developed in Section IV. In Section V, we evaluate the performance of MAX-k-GPSO and compare it with its UCB-based counterpart through simulations. Finally, in Section VI, we summarize our contributions.

II. System Model

We consider a remote estimation system with N sensors with randomly evolving states at a node that a remote receiver wants to track. The value of each sensor follows an i.i.d Wiener Process. The transmitter has access to sensor state values and sends them to the receiver through a communication channel at a cost(defined below).

The transmitter can decide *when* and *of which sensors* it sends the state value to the receiver. Using the most recently received value for each sensor i, the receiver tracks all N states. Associated with the updates, there are two different types of system cost. One is the estimation cost due to stale sensor values, and the other is the update cost due to transmissions. Our goal is to minimize the total average system cost given that the transmission cost is a nonlinear function of the number of simultaneous updates, which is the case for many scenarios (as will be discussed).

We now formally describe our problem in the following. Let $X_1(t), X_2(t), \cdots, X_N(t)$ be independent standard Wiener Processes evolving in continuous time, where $X_i(t)$ is the true state value of sensor i at time t. Let $u_t^i \in \{0,1\}$ denote the decision variable for the transmitter to update the receiver with the state of sensor i at time t: if $u_t^i = 1$, the transmitter sends $X_i(t)$ to the receiver, and if $u_t^i = 0$, it does not. Multiple sensor values can be updated simultaneously. Let n_t denote the number of sensor values transmitted at time t, i.e., $n_t = \sum_{i=1}^{N} u_t^i$. Let $\hat{X}_i(t)$ denote an estimated value for $X_i(t)$ at the receiver, which evolves as

$$\hat{X}_i(t) = \begin{cases} X_i(t), & \text{if } u_t^i = 1, \\ X_i(t'), & \text{if } u_t^i = 0, \end{cases}$$

where t' be the time of the latest update for sensor i. We assume $\hat{X}_i(0) = 0$. The estimation cost reflects information staleness through the estimation error, which is defined as $\mathcal{E}_i(t) = X_i(t) - \hat{X}_i(t)$. We use the squared error sum for the estimation cost at time t as $\sum_i \mathcal{E}_i^2(t)$. It can be reduced by frequently updating the sensor values, which, however, incurs a competing cost of transmission. In particular, we consider a transmission cost function that is sublinear in the number of transmitted sensor values. Theoretically, this is supported by the information theoretic foundation that encoding more information into the transmission increases the coding rate (since the codeblock length increases). Practically, this is supported by the fact that sending multiple updates at once allow the use of less overhead in each transmission (for synchronization, probing, channel estimation, etc.). Accordingly, we model the sublinear transmission cost as

$$f(n_t) = \bar{c} \cdot (n_t)^{\alpha} \cdot \delta(t), \tag{1}$$

where $\bar{c} > 0$ and $0 < \alpha \leq 1$ are some constants, and $\delta(t)$ is the impulse function. We set zero cost for no update $f(0) = 0$.

Our objective is to minimize the expected average system cost R that combines both the estimation cost and the transmission cost, defined as

$$R = \lim_{T \to \infty} \mathbb{E}\left[\frac{1}{T} \int_{t=0}^{T} \left(\sum_{i=1}^{N} \mathcal{E}_i^2(t) + f(n_t) \right) \mathrm{d}t \right]. \tag{2}$$

Similar, but different, problems have been addressed in the literature. It is shown that a threshold policy achieves the optimal performance in different one-dimensional stochastic processes [4]–[10]. In the multi-dimensional case, a threshold policy is shown to be optimal when it samples multiple Wiener Processes all together [11]. From earlier results, we are motivated to define and optimize our solution in a broader class of threshold policies that accommodates the above cases but also extends them to improve the performance in our generic setting. Our formulation differs from the previous works in that we consider the unexplored case of sublinear form of the transmission cost, which is of important in practice and also generates interesting new characteristics in the design.

III. Benchmark Threshold-based Policies

We first consider two threshold policies that appeared in the literature. Then, we generalize them with a parameter, which makes the two policies are extreme cases. We show that neither extreme policy is optimal with transmission cost (1). For ease of explanation, we introduce a notion of *predicted estimation error* $\tilde{\mathcal{E}}_i(t)$, which is the predicted error before the transmitter makes a decision, assuming that it does not update the value of sensor i at time t, i.e., $\tilde{\mathcal{E}}_i(t) = X_i(t) - X_i(t')$.

A. All-at-once policy [11]

All-at-once policy that appeared in [11] updates all sensor values simultaneously when the transmitter updates the receiver, i.e., it has the restriction of $n_t \in \{0, N\}$. It has been shown in [11] that an optimal policy makes an update decision only if $\sqrt{\sum_i \tilde{\mathcal{E}}_i(t)^2} \geq \gamma^{all}$ with $\gamma^{all} = \sqrt[4]{2(N+2)f(N)}$. Under the policy, the optimal average cost denoted by R^{all} can be obtained as $R^{all} = \sqrt{\frac{2N^2 f(N)}{N+2}}$.

B. Multi-threshold policy

Another natural extreme policy that we consider is Multi-threshold policy, under which an update decision for each sensor is determined independently of the others. In Multi-threshold policy, $u_t^i = 1$ when $|\tilde{\mathcal{E}}_i(t)| \geq \gamma^{mul}$. Otherwise, $u_t^i = 0$. In this case, clearly the optimal policy for the single sensor is of threshold type, and its behavior has been well studied in [10]. We have the optimal average cost under Multi-threshold policy, which is denoted by R^{mul}, as $R^{mul} = \frac{\sqrt{6}}{3} \sum_{i=1}^{N} \sqrt{f(1)}$ with optimal threshold $\gamma^{mul} = \sqrt[4]{6f(1)}$.

Comparing the two costs of R^{all} and R^{mul} along with (1), All-at-once policy has a lower cost when $\alpha < \log_N \frac{N+2}{3}$. Note that All-at-once policy is restricted in its updates and thus it would have higher estimation cost than Multi-threshold policy. Despite the handicap, it can achieve overall better performance if the benefit of simultaneous updates is large (i.e., when α is small). This observation motivates us to investigate the trade-off between the estimation cost and the transmission cost through a generalized version of the two policies.

C. A generalized threshold-based policy: MAX-k policy

We generalize All-at-once policy and Multi-threshold policy as follows. At time t, let $\{\pi_t^i\}_{i=1}^N$ denote the permutation of sources in the order of the largest predicted estimation error first, which means $\tilde{\mathcal{E}}_{\pi_t^i} \geq \tilde{\mathcal{E}}_{\pi_t^{i+1}}$. Also, let $A_t(k)$ denote the first k elements of the permutation, i.e., $A_t(k) = \{\pi_t^1, \pi_t^2, \ldots, \pi_t^k\}$.

Definition 1. MAX-k policy *is a threshold-based policy that updates k out of N sensor values as*

$$u_t^i = \begin{cases} 1, & \text{if } \sqrt{\sum_{j\in A_t(k)} \tilde{\mathcal{E}}_j(t)^2} \geq \gamma \text{ and } i \in A_t(k), \\ 0, & \text{otherwise}. \end{cases}$$

The optimal threshold $\gamma^{(k)}$ is defined as the value of γ, with which MAX-k policy achieves the minimum average cost.

We note that MAX-k policy becomes All-at-once policy when $k = N$, and Multi-threshold policy when $k = 1$, i.e., $\gamma^{(N)} = \gamma^{all}$ and $\gamma^{(1)} = \gamma^{mul}$. An interesting question is whether this generalization can improve the average cost performance of the other two policies, and if so under what circumstances. It is also of interest to develop methods that can find the optimal k and $\gamma^{(k)}$ without knowledge of sensor dynamics or transmission cost functions. While we will investigate these questions in the rest of the paper, we first numerically evaluate the performance of MAX-k policy with different transmission cost functions. Interestingly, with a certain cost function, we can minimize the total cost by selecting an appropriate $k \in \{2, \cdots, N-1\}$.

We conduct simulations for a system with N sensors. For All-at-once policy and Multi-threshold policy, we use their optimal threshold levels γ^{all} and γ^{mul}, respectively. For MAX-k policy, we empirically find the best-performing integer threshold γ for each $k \in [N]$ where $[N] := \{1, 2, \cdots, N\}$.

Fig. 1 shows the average cost of MAX-k policy with different α when $N = 7$. For each α, the best-performing k value of MAX-k policy is marked by a cross. Recall that Multi-threshold policy and All-at-once policy is the same as MAX-1 policy and MAX-7 policy, respectively. We highlight that in general, neither Multi-threshold policy nor All-at-once policy is optimal, and MAX-k policy depends on α, i.e., $k = 2$ when $\alpha = 0.75$, $k = 4$ when $\alpha = 0.5$, and $k = 5$ when $\alpha = 0.25$: best-performing k increases to N as α decreases to 0. All-at-once policy would be the best policy if the transmission cost is irrelevant to the number of simultaneous transmissions, i.e., when $\alpha = 0$.

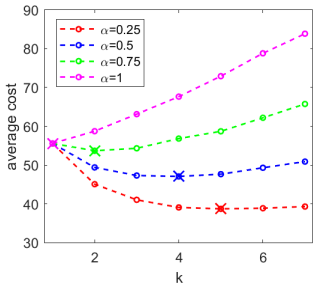


Figure 1. Average cost of MAX-k policy.

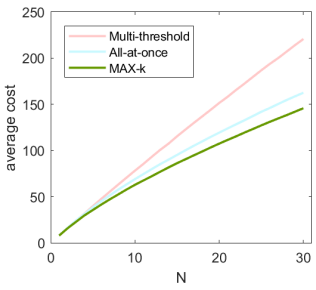


Figure 2. Average cost of three policies with $\alpha = 0.5$.

We further compare the performance of the policies with a larger number of sensors. Fig. 2 shows that as the number of sensors increases, the average cost linearly increases under Multi-threshold policy. In contrast, the cost increases sublinearly under All-at-once policy and MAX-k policy. It is expected that MAX-k policy achieves the lowest average cost, since MAX-k policy is a generalization of the others. Nevertheless, we highlight that the performance gap enlarges as the number of sensors increases.

IV. Optimizing MAX-k Policy

For the generalized MAX-k policy, it is essential to find the best-performing k and the corresponding threshold level $\gamma^{(k)}$ for the optimal performance. Let k^* and $\gamma^{(k^*)}$ denote the best-performing k and the corresponding optimal threshold, respectively. Unfortunately, it is not tractable to find an analytical solution to this problem due to the complex and highly coupled dynamics that exist in the system evolution. Moreover, we are interested in finding the solution without knowledge of the transmission cost function or the statistics of the Wiener Processes. This motivates us to approach the problem as an online learning problem for $(k^*, \gamma^{(k^*)})$, where $k^* \in [N]$.

In our scenarios, there is a certain structure of average cost function. For example, we empirically observed that for a fixed k, if $\gamma' < \gamma'' \leq \gamma^{(k)}$, then the average cost of (k, γ') is higher than the average cost of (k, γ''). This implies that we may develop learning mechanisms faster if we use the underlying structure of the cost function. In our setup, however, the structure of the cost function is not explicitly known. Moreover, the space (k, γ) over which we must find the optimal parameter is a mixed (discrete, continuous) space. These characteristics prevent us from using existing approaches. In the next section, we investigate a novel learning strategy that employs multiple points to search the space.

A. MAX-k policy optimization with GPSO

Assuming that the underlying average cost function is smooth, we employ the Particle Swarm Optimization (PSO) technique [12]. The PSO method is a population-based optimization method. It maintains a population of particles which are constitutive of a swarm where each particle represents a potential solution. Each particle moves around the search space, guided by two positions: the best position among the visits of the particle itself, and the best position among the

visits of all particles. Repeating the guided movements in this manner allows the swarm of particles to pursue an optimal solution as a group. PSO has several attractive features: (i) it is easy to implement, (ii) it needs no gradient information of the objective function, and (iii) it allows real-value parameters. Despite these advantages, the original PSO method cannot be directly applied to our problem of finding $(k^*, \gamma^{(k^*)})$, since PSO is designed for deterministic objective functions. In our problem, the average cost function is stochastic, in the sense that for given (k, γ), the cost is randomly drawn from some unknown fixed distribution. To this end, we modify the PSO algorithm to incorporate it with MAX-k policy.

We consider a bounded search space of integer k and real value γ, where particles move around. We partition the search space into a grid of small pieces called *cell*s, and collect the statistics for each cell according to the particles' visits to the cell. This cell level integration intends to approximate the statistical value of our stochastic objective function with the real-number search space. The particles' movement that incurs their visits to a cell (except the cell level integration) is the same as that of the original PSO. We denote this modified PSO as Grid-PSO (GPSO), and combine it with MAX-k policy, denoted by MAX-k-GPSO. Detailed operations are shown in Algorithm 1 and described as follows.

We divide the time horizon into rounds, where a round is the time interval in which total U sensor updates occur under MAX-k policy. At the end of each round, we update all the sensors and reset the estimation error to 0, which incurs additional communication cost by replacing update cost $f(k)$ with $f(N)$ for the last update in the round. For MAX-k policy, this additional cost is unavoidable for practical online learning algorithm. However, it becomes negligible for a reasonably long round, e.g., $U = N^2$, which roughly corresponds to N updates for each sensor. Let Δ be the length of a round (random variable) and $\hat{r}$ be the average cost during the round, i.e.,

$$\hat{r} = \frac{1}{\Delta} \int_{\Delta} \left(\sum_{i=1}^{N} \mathcal{E}_i^2(t) + f(n_t) \right) \mathrm{d}t. \tag{3}$$

We consider the two-dimensional search space (k, γ) with $k \in [N]$ and real-valued $\gamma \in [\gamma^{mul}, \gamma^{all}]$. Note that we have a bounded range $[\gamma^{mul}, \gamma^{all}]$ for $\gamma^{(k^*)}$ since $\gamma^{(k^*)}$ increases as k^* increases. We divide it into a grid with cells, by partitioning the range of γ into M intervals denoted by $\mathcal{I} = \{I_m\}_{m=1}^{M}$, each of which centered at $\{\gamma_m\}_{m=1}^{M}$ with $M < \infty$. There are total $N \times M$ cells, where each cell c includes an integer k_c (for k) and a range $[\underline{\gamma}_c, \overline{\gamma}_c) \in \mathcal{I}$ (for γ). Let C be the set of cells. We define function $h : [N] \times \mathbb{R} \to C$ that maps a position (k, γ) to cell c with $k = k_c$ and $\gamma \in [\underline{\gamma}_c, \overline{\gamma}_c)$. Also, we define function $g : C \to \mathbb{R}$ that maps cell c to the average cost over the cell, i.e., $g(c) = \mathbb{E}_c[R]$, where the expectation is over $\gamma \in [\underline{\gamma}_c, \overline{\gamma}_c)$. Our goal is to identify the cell c^* that minimizes $g(c)$, assuming that $g(c^*)$ is close to the average cost under MAX-k policy with $(k^*, \gamma^{(k^*)})$. For a given c, the function $g(c)$ is unknown. Thus, we use empirical mean incurred by particles' visits to a cell. Let $\tilde{g}(c)$ denote the empirical mean of average costs, each of which is generated by a particle's visit to cell c.

Algorithm 1 MAX-k-GPSO policy

Input: p_1, p_2, w, U, swarm size $|S|$, set of cells C

Create a swarm of size $|S|$
For all $s \in S$, initialize $\mathbf{x}_s$ and $\mathbf{v}_s \leftarrow 0$
while stopping condition is not true **do**
  for each $s \in S$ **do**
    $\mathbf{x}_s \leftarrow \mathbf{x}_s + \mathbf{v}_s$
    $x_{s1} \leftarrow \lfloor 0.5 + x_{s1} \rfloor$
    Update sensors under MAX-k policy with $\mathbf{x}_s$ by U times, and obtain $\hat{r}$
    $V^h(\mathbf{x}_s) \leftarrow V^h(\mathbf{x}_s) + 1$
    $\tilde{g}^h(\mathbf{x}_s) \leftarrow \tilde{g}^h(\mathbf{x}_s)(1 - \frac{1}{V^h(\mathbf{x}_s)}) + \frac{\hat{r}}{V^h(\mathbf{x}_s)}$
    if $\tilde{g}^h(\mathbf{x}_s) < \tilde{g}^h(\mathbf{y}_s)$ **then**
      $\mathbf{y}_s \leftarrow \mathbf{x}_s$
    end if
  end for
  $s^* \leftarrow \mathrm{argmin}_s\{\tilde{g}^h(\mathbf{y}_s)\}$, and $\hat{\mathbf{y}} \leftarrow \mathbf{y}_{s^*}$
  Calculate ρ using (5)
  for $d \in \{1, 2\}$ **do**
    Draw e_3 uniformly at random in range $(0, 1)$
    $v_{s^*d} \leftarrow -x_{s^*d} + \hat{y}_d + w v_{s^*d} + \rho(1 - 2e_3)$
  end for
  for all $s \in S \setminus \{s^*\}$ and $d \in \{1, 2\}$ **do**
    Draw e_1, e_2 uniformly at random in range $(0, 1)$
    $v_{sd} \leftarrow w v_{sd} + p_1 e_1 [y_{sd} - x_{sd}] + p_2 e_2 [\hat{y}_d - x_{sd}]$
  end for
end while
return $\hat{\mathbf{y}}$

Let S be the set of particles in the swarm. Each particle $s \in S$ has position $\mathbf{x}_s = (x_{s1}, x_{s2})$ with $x_{s1} \in [N]$ and $x_{s2} \in [\gamma^{mul}, \gamma^{all}]$. Suppose that the initial location of each particle is determined. For a particle s located at $\mathbf{x}_s$, we set $(k, \gamma) = (x_{s1}, x_{s2})$, and run one round of U updates under MAX-k policy starting from $\mathcal{E}_i(t) = 0$ for all sensor i's. This procedure is called as the *evaluation of position* $\mathbf{x}_s$. For the round, an average cost can be computed as (3). We repeat the evaluation of each particle's position $\mathbf{x}_s$ for all $s \in S$.

Then, we compute the empirical mean of average cost of each cell as follows. Let $\mathbf{x}^j$ be the position of the particle that is chosen at round j and $\hat{r}_j$ be the corresponding empirical average cost obtained at the end of round. Let $V_J(c)$ denote the number of visits of any particles to cell c up to round J, i.e.,

$$V_J(c) = \sum_{j=1}^{J} \mathbb{I}_{\{h(\mathbf{x}^j) = c\}},$$

where $\mathbb{I}_{\{A\}}$ is an indicator function of event A; $\mathbb{I}_{\{A\}} = 1$ if A is true and 0 otherwise. Initially, $V_0(c) = 0$ for all c. The empirical mean $\tilde{g}_J(c)$ for cell c after round J is given as

$$\tilde{g}_J(c) = \frac{1}{V_J(c)} \sum_{j=1}^{J} \mathbb{I}_{\{h(\mathbf{x}^j) = c\}} \cdot \hat{r}_j.$$

We set $\tilde{g}_J(c) = 0$ if $V_J(c) = 0$. We also define composite functions $V_J^h(\mathbf{x}) = V_J(h(\mathbf{x}))$ and $\tilde{g}_J^h(\mathbf{x}) = \tilde{g}_J(h(\mathbf{x}))$, that maps location $\mathbf{x}$ to the number of visits and the empirical mean

of average cost of the cell that $\mathbf{x}$ belongs to, respectively. For brevity, we omit subscript J.

We denote a group of $|S|$ rounds as a phase, in which the position of each particle $s \in S$ has been evaluated for exactly one round of updates. After phase l, we now make the particles move. Let $\mathbf{y}_s$ denote the *local best position* of particle s based on its visit history, and also let $\hat{\mathbf{y}}$ denote *global best position* of all particles based on all the visit history. They can be formally defined as

$$\mathbf{y}_s(l) = (y_{s1}, y_{s2}) = \underset{\mathbf{x}\in\{\mathbf{x}_s(l),\mathbf{y}_s(l-1)\}}{\operatorname{argmin}} \tilde{g}^h(\mathbf{x}),$$
$$\hat{\mathbf{y}}(l) = (\hat{y}_1, \hat{y}_2) = \underset{\mathbf{x}\in\{\mathbf{y}_s(l)\}_{s\in S}}{\operatorname{argmin}} \tilde{g}^h(\mathbf{x}),$$

where $\mathbf{x}_s(l)$ is the location of particle s in phase l. These two variables are used to compute the *velocity* $\mathbf{v}_s(l+1) = (v_{s1}(l+1), v_{s2}(l+1))$ of particle s at phase $l+1$, which will be used to calculate new position of particle s as

$$x_{s1}(l+1) = \operatorname{nint}(x_{s1}(l) + v_{s1}(l+1)),$$
$$x_{s2}(l+1) = x_{s2}(l) + v_{s2}(l+1),$$

where $\operatorname{nint}(a)$ returns the integer closest to a. The velocity is computed separately for each dimension. In our case, for each $d \in \{1, 2\}$, we have

$$\begin{aligned} v_{sd}(l+1) = w \cdot v_{sd}(l) &+ p_1 \cdot e_1 \cdot [y_{sd}(l) - x_{sd}(l)] \\ &+ p_2 \cdot e_2 \cdot [\hat{y}_d(l) - x_{sd}(l)], \end{aligned} \tag{4}$$

where inertia weight w and acceleration coefficients p_1, p_2 are constants, and e_1 and e_2 are an independent random variable drawn uniformly in range $(0, 1)$. Once new position $\mathbf{x}_s(l+1)$ is determined for all particle s's, then new phase $l+1$ starts from the evaluation of the position for each particle.

The above description is based on the original PSO algorithm, which can be possibly stuck at a suboptimal solution. A remedy has been developed by modifying the movement of the best-performing particle in [13]. We also incorporate it. At each phase l, let s^* denote the best-performing particle satisfying $\mathbf{y}_{s^*}(l) = \hat{\mathbf{y}}(l)$, and we call s^* the *global best particle*. The velocity of particle s^* is computed as $v_{s^*d}(l+1) = -x_{s^*d}(l) + \hat{y}_d(l) + w v_{s^*d}(l) + \rho(l)(1-2e_3)$, where e_3 is another independent random variable drawn uniformly in range $(0, 1)$, and $\rho(l)$ is determined as

$$\rho(l+1) = \begin{cases} 2\rho(l), & \text{if } \mathbf{N}_{succ} > \lambda_s, \\ 0.5\rho(l), & \text{if } \mathbf{N}_{fail} > \lambda_f \text{ and } \rho(l) > \rho_{min}, \\ \rho(l), & \text{otherwise}, \end{cases} \tag{5}$$

where $\rho_{min}, \lambda_s, \lambda_f$ are pre-determined parameters. Letting a 'success' denote the case when $\tilde{g}^h(\hat{\mathbf{y}}(l+1)) < \tilde{g}^h(\hat{\mathbf{y}}(l))$ and a 'failure' for the other case, $\mathbf{N}_{succ}$ is the number of phase-consecutive successes and $\mathbf{N}_{fail}$ is the number of phase-consecutive failures. The new velocity makes particle s^* moves to a point that is uniformly sampled from a square region with side lengths $2\rho(l)$ centered around $\hat{\mathbf{y}}(l) + w\mathbf{v}_{\mathbf{s}^*}(l)$, where the region size depends on the event history of successes and failures. This behavior contributes to the convergence of the algorithm.

B. Convergence of Grid-PSO

We investigate the convergence of GPSO algorithm toward an optimal solution under an assumption that $g(c)$ is convex. It has been shown in [14]–[16] that all particles converge to a point $\hat{\mathbf{y}}$ in the search space with appropriate parameters of w, p_1, and p_2, i.e., $\lim_{l\to\infty} \mathbf{x}_s(l) = \hat{\mathbf{y}}$ for all $s \in S$. We show that if the converging point $\hat{\mathbf{y}}$ is not the optimal solution, then the particles can move toward the optimal solution. For given $\epsilon > 0$, let G^*_ϵ be the set of cells whose average cost is ϵ-close to the minimum g^* where $g^* := \min_c g(c)$, i.e., $G^*_\epsilon = \{c \mid g(c) - g^* < \epsilon\}$.

Theorem 1. *If average cost function $g(c)$ is convex, then for any $\epsilon > 0$, the particles converge to an optimal region under the GPSO algorithm, i.e., with probability* 1*,*

$$\lim_{l\to\infty} h(\mathbf{x}_s(l)) \in G^*_\epsilon, \quad \textit{for all } s \in S.$$

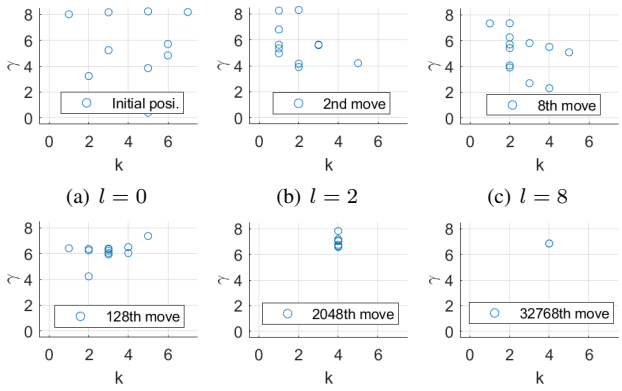
We omit the proof due to lack of space. The MAX-k-GPSO policy can successfully find the cell in G^*_ϵ for any $\epsilon > 0$ under a convex average cost function $g(c)$ from Theorem 1. By using sufficiently small $\epsilon > 0$, it can achieve near-optimal solution. In the following section, we evaluate the performance of MAX-k-GPSO policy and compare it with its UCB-based counterpart.

V. Simulations

For the comparison, we introduce another approach of online learning to find the best-performing (k, γ), which applies a well-known algorithm of UCB [17] for all possible arms (k, γ). We present this approach that combines MAX-k policy with the UCB algorithm, denoted by MAX-k-UCB. We again partition the range of γ into M intervals of $\mathcal{I}$ as before. Otherwise, we have infinite number of arms due to real-valued threshold level γ. We assume $\gamma \in \{\gamma_m\}_{m=1}^M$ for MAX-k-UCB policy. At the beginning of each round, MAX-k-UCB selects an arm (k, γ) with smallest UCB index and we make U update decisions using MAX-k policy with these parameters. Then, UCB index for the arm is updated using the resulting average cost. This process continues until a stopping condition is true.

We set up parameters for simulations as follows. For MAX-k-GPSO policy, we create a swarm with $|S|$ number of particles, and set $|S| = 10 + \lfloor 2\sqrt{D} \rfloor$ where D is the number of dimensions in the search space, according to [18]. In our case of $D = 2$, the size of swarm, $|S|$, becomes 12. We also set inertia weight and acceleration coefficients in (4) as $w = 0.7298$ and $p_1 = p_2 = 1.49618$ as in [19]. For the velocity equation of global best particles s^*, we set $\rho_{min} = 4\delta$, $\rho(0) = 1$, $\lambda_s = 5$, and $\lambda_f = 5$ in (5). We set parameters $U = N^2$ and $N = 7$, and use the transmission cost function with $\alpha = 0.5$, and $\bar{c} = 100$.

For MAX-k-GPSO policy, we first observe the convergence of all particles to the optimal solution. Fig. 3 shows the positions of all particles after l-th improvement phase when J is set to be a million and $M = 1000$. Initially, the particles are randomly located in the search space as shown in Fig. 3(a) and move toward the optimal solution. Finally, the particles



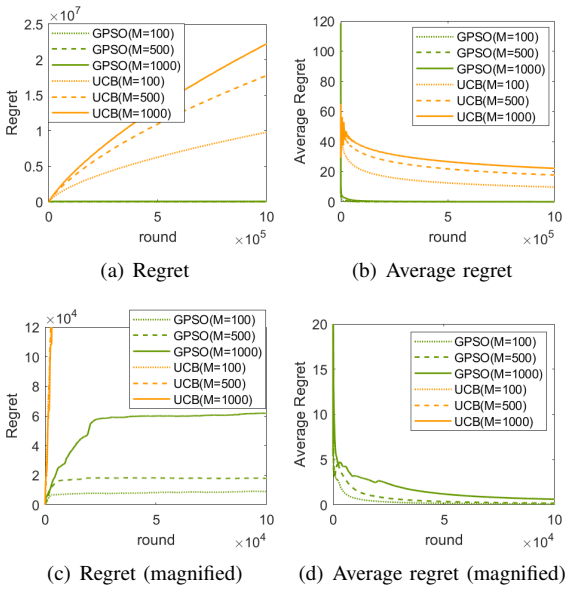
(a) $l = 0$ (b) $l = 2$ (c) $l = 8$

(d) $l = 128$ (e) $l = 2048$ (f) $l = 32768$

Figure 3. Positions of particles ($|S| = 12$), after l-th phase.

converge to the $(k^* = 4, \gamma^{(k^*)} = 6.8530)$ as shown in Fig. 3(f).

We now compare the performance of MAX-k-GPSO policy and MAX-k-UCB policy observing the *regret* of policies, which is the performance metric widely used in the learning area. We first empirically find the optimal average cost using MAX-k-GPSO policy, then we draw (accumulated) regret and per-round average regret with respect to the optimal average cost. Fig. 4 shows the regret and average regret up to $J = 10^6$. The orange lines represent the results under MAX-k-UCB policy and the green lines for MAX-k-GPSO policy. The solid lines are for the case when $M = 1000$, the dashed lines are for $M = 500$, and the dotted lines are for $M = 100$. Since the regret under MAX-k-GPSO policy compared with MAX-k-UCB policy is too small to recognize, we magnify them as shown in Fig. 4(c) and 4(d). We observe that the speed of convergence toward the optimal solution under MAX-k-GPSO policy is drastically faster than that under MAX-k-UCB policy. Although UCB algorithm is known to achieve asymptotically optimal performance in MAB problem [17], it does not exploit the underlying structure of the objective function, which makes the speed of convergence slow under MAX-k-UCB policy.



(a) Regret (b) Average regret

(c) Regret (magnified) (d) Average regret (magnified)

Figure 4. Regret performance of MAX-k-GPSO & MAX-k-UCB.

VI. Conclusions

We developed a novel strategy, MAX-k policy, which improved the average system cost over All-at-once and Multi-threshold policies as it exploits the sublinear structure of the communication cost function. We then developed the GPSO optimization technique for the online learning algorithm that adaptively optimizes the parameters of MAX-k policy with reasonable speed of convergence.

References

[1] M. Costa, M. Codreanu, and A. Ephremides, "Age of information with packet management," in *2014 IEEE International Symposium on Information Theory*. IEEE, 2014, pp. 1583–1587.

[2] Q. He, D. Yuan, and A. Ephremides, "Optimizing freshness of information: On minimum age link scheduling in wireless systems," in *2016 14th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*. IEEE, 2016, pp. 1–8.

[3] A. M. Bedewy, Y. Sun, S. Kompella, and N. B. Shroff, "Age-optimal sampling and transmission scheduling in multi-source systems," in *Proceedings of the Twentieth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 2019, pp. 121–130.

[4] G. M. Lipsa and N. C. Martins, "Remote state estimation with communication costs for first-order lti systems," *IEEE Transactions on Automatic Control*, vol. 56, no. 9, pp. 2013–2025, 2011.

[5] A. Nayyar, T. Başar, D. Teneketzis, and V. V. Veeravalli, "Optimal strategies for communication and remote estimation with an energy harvesting sensor," *IEEE Transactions on Automatic Control*, vol. 58, no. 9, pp. 2246–2260, 2013.

[6] O. C. Imer and T. Basar, "Optimal estimation with limited measurements," in *IEEE Conference on Decision and Control*, 2005.

[7] X. Gao, E. Akyol, and T. Başar, "On remote estimation with multiple communication channels," in *American Control Conference (ACC)*, 2016.

[8] J. Chakravorty, J. Subramanian, and A. Mahajan, "Stochastic approximation based methods for computing the optimal thresholds in remote-state estimation with packet drops," in *American Control Conference (ACC)*, 2017.

[9] J. Yun, C. Joo, and A. Eryilmaz, "Optimal real-time monitoring of an information source under communication costs," in *IEEE Conference on Decision and Control (CDC)*, 2018.

[10] Y. Sun, Y. Polyanskiy, and E. Uysal-Biyikoglu, "Remote estimation of the wiener process over a channel with random delay," in *IEEE International Symposium on Information Theory (ISIT)*, 2017.

[11] K. Nar and T. Başar, "Sampling multidimensional wiener processes," in *IEEE Conference on Decision and Control*, 2014.

[12] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *IEEE ICNN*, vol. 4, 1995.

[13] F. van den Bergh and A. P. Engelbrecht, "A new locally convergent particle swarm optimiser," in *IEEE International conference on systems, man and cybernetics*, vol. 3, 2002.

[14] F. Van den Bergh and A. P. Engelbrecht, "A convergence proof for the particle swarm optimiser," *Fundamenta Informaticae*, vol. 105, no. 4, pp. 341–374, 2010.

[15] R. Poli, "Mean and variance of the sampling distribution of particle swarm optimizers during stagnation," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 4, pp. 712–721, 2009.

[16] C. W. Cleghorn and A. P. Engelbrecht, "Particle swarm convergence: Standardized analysis and topological influence," in *International Conference on Swarm Intelligence*. Springer, 2014, pp. 134–145.

[17] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Mach. Learn.*, vol. 47, no. 2-3, May 2002.

[18] M. Clerc, *Particle swarm optimization*. John Wiley & Sons, 2010, vol. 93.

[19] F. Van Den Bergh and A. P. Engelbrecht, "An analysis of particle swarm optimizers," Ph.D. dissertation, 2002.