Age-Optimal UAV Scheduling for Data Collection with Battery Recharging

Ghafour Ahani, Di Yuan, Senior Member, IEEE, and Yixin Zhao

Abstract—We study route scheduling of a UAV for data collection from remote sensor nodes (SNs) with battery recharging. The freshness of the collected information is captured by the metric of age of information (AoI). The objective is to minimize the average AoI cost of all SNs over a scheduling time horizon. We prove that the problem is NP-hard via a reduction from the Hamiltonian path. Next, we prove tractability of the problem for a symmetric scenario. For problem solving, we develop an algorithm based on graph labeling. Finally, we show the effectiveness of our algorithm in comparison to greedy scheduling.

Index Terms—Age of information, data collection, path planning, scheduling, UAV.

I. INTRODUCTION

A. Motivations

Recently, UAVs are becoming employed for data collection from sensor nodes (SNs) in remote areas [1]. A UAV can travel to hard-to-reach areas, collect information, and carry them back to a base station (BS) for data analysis. The information should be delivered to the BS in a timely manner. To this end, age of information (AoI) is a relatively newly introduced performance metric that captures the freshness of received information. It is defined as the amount of time elapsed with respect to the generation time of latest received information [2].

The works [3]–[7] have studied data collection via UAV with objectives related to AoI. In [3], assuming Euclidian distances between the SNs, maximum and average Aois are minimized via dynamic programming and genetic algorithms. In [4], the authors extended the system model in [3] to the scenario where the UAV can collect information from a set of SNs at a so-called data collection point. They proposed an SN association and trajectory planning policy to minimize the maximum AoI of all SNs. In [5], the authors minimize the average AoI under energy constraint of SNs. They proposed a new data acquisition model for uploading data, consisting of three modes, namely hovering, flying, and hybrid. In [6], an AoI-optimal data collection and dissemination problem on graphs is studied, where a UAV flies along the randomized trajectory to minimize the average and maximum Aois. In [7], UAV is used as a mobile relay between a source-destination pair to minimize the maximum AoI.

The aforementioned works assumed that all SNs must be visited by the UAV before flying back to the BS. However, this may not be optimal. For example, consider two SNs such that the BS is at the middle point of the straight line between the two SNs. It is straightforward to see that it is not optimal to visit both SNs first and then fly to the BS for delivering the collected data. In addition, these studies considered the scenario of SNs of equal importance, whereas in reality the SNs, depending on the application, may have SN-specific cost functions of AoI. Hence, which SNs to visit for data collection and when to fly to the BS for delivering the collected data are two key questions in optimal data collection via UAV. In addition, the battery consumption of UAV has to be considered as the amount of battery energy may allow for visiting only some subset of the SNs, before the UAV has to get to the BS for charging. Waiving these limitations calls for further investigation.

B. Contributions

We study optimal scheduling of a UAV for collecting data from a set of SNs over a given time horizon. The UAV is of limited battery capacity, and for each SN a specific AoI cost function is defined. Our contributions are as follows:

- We provide complexity analysis. First, we formally prove that the problem is NP-hard via a reduction from the Hamiltonian path problem. Next, we prove when all SNs have uniform travel time from the BS and a common AoI cost function, the problem is polynomial-time solvable.
- For problem solving, we develop a polynomial-time solution method based on the concept of graph labeling and time slicing. The number of labels in the algorithm naturally acts as a control parameter for the trade-off between computational effort and solution optimality.
- We conduct simulations to show the effectiveness of our solution approach, by comparing it to a greedy schedule. Our solution approach outperforms the greedy solution.

II. SYSTEM MODEL

The system scenario consists of a BS, a UAV with limited battery capacity, and a set of $S$ SNs with index set $S = \{1, \ldots, S\}$. The UAV is utilized for collecting information from the SNs and carry them to the BS over a time horizon of length $T$. The system scenario is shown in Fig. 1.

We define $S^+ = \{0\} \cup S$ the index set of all SNs and the BS, in which index 0 is reserved for the BS. Traveling from $i \in S^+$ to $j \in S^+$ takes $t_{ij}$ amount of time and consumes $b_{ij}$ amount of energy. The UAV repeatedly departs from the BS, visits a subset of SNs $S' \subseteq S$, collects information, and flies back to the BS for information delivery. For each visited SN, the UAV hovers over the SN and establishes a communication link with the SN. The SN senses the information, generates a data packet, and transmits it to the UAV. The corresponding
time and energy required for these operations, without loss of
generality, are embedded into \(t_{ij}\) and \(b_{ij}\), respectively.

The UAV has to go back to the BS before its battery energy
becomes exhausted. Denote by \(B\) the battery capacity. Each
time the UAV returns to the BS, it has the choice of getting
partially or fully recharged. Denote by \(g(\cdot)\), the recharging
function of the battery. Note that we do not assume that \(g(\cdot)\)
has to be linear. At time \(t\), \(u(s, t)\) stands for the timestamp
of the most recently received information of SN \(s\) available in
the BS. Denote by \(a(s, t)\) the AoI of SN \(s\) at time \(t\). Thus, the AoI
at time \(t\) can be calculated as \(a(s, t) = t − u(s, t)\). The AoI
vector of all SNs at time \(t\) is represented by \(a(t)\). Notation
\(f_s(\cdot)\) is used for the AoI cost function of SN \(s \in S\). The problem
consists in age-optimal UAV scheduling (AUS) for
data collection from the SNs over the time horizon of duration
\(T\), with the objective of minimizing the overall average AoI
cost of all SNs.

Remark. We say \(a(t) < a'(t)\) if and only if \(a(s, t) \leq a'(s, t)\)
for \(s = 1, 2, \ldots, S\) and there exists at least one index \(j\) for
which \(a(j, t) < a'(j, t)\).

### III. COMPLEXITY ANALYSIS

**Theorem 1.** AUS is NP-hard.

**Proof.** The proof is based on a polynomial-time reduction
from the Hamiltonian path problem that is NP-complete [3].
In the Hamiltonian path problem, there are a set of nodes \(N\)
and a set of edges \(E\). The task is to determine if there is a
path visiting every node exactly once.

We construct a reduction from the Hamiltonian path
problem as follows. We set \(S = N\). Consider any two SNs \(i\) and
\(j, i, j \in S\). If there is a link in \(E\) between the corresponding
nodes of \(i\) and \(j \in N\), we set \(t_{ij} = 4\), otherwise \(t_{ij} = 16\).
We define an edge between the BS and each SN \(s \in S\) with
\(t_{0s} = 8\). The value of \(b_{ij}\) can be any positive value,
e.g., \(b_{ij} = t_{ij}\). Let \(B = \sum_{(i, j) \in E} b_{ij}\). The time horizon is
\(T = 4S + 14\) and an AoI cost function is defined for all SNs
as follows:

\[
f_s(x) = \begin{cases} 
0 & x \leq 4S + 13 \\
100 & x > 4S + 13 
\end{cases}
\]  

(1)

Now, solving the defined instance of AUS will solve
the Hamiltonian path problem. Because, if the overall AoI cost
at the optimum is zero, it means that the UAV departs from the
BS at time zero, visits each SN \(s \in S\) exactly once, flies back
to the BS, and delivers the collected data at time \(4S + 12\).
For any other tour, either the data of at least one SN is not
collected within the time horizon and hence the AoI is 100,
or an SN is visited twice in which the UAV can not deliver
the collected data before time point \(4S + 15\), or the tour is
not using the edges of the original graph and the UAV can not
deliver the collected data before time point \(4S + 24\). Therefore,
the AoI cost of at least one SN is 100 in time interval \([4S + 13, 4S + 14]\), and the overall average AoI cost has to be at
least \(\frac{100}{S(4S + 14)}\). Hence the conclusion.

In the following we consider a special case of AUS,
referred to as symmetric AUS (S-AUS), for which we prove its
tractability. This problem corresponds to the scenario where
SNs are located on a circle and the BS is at the center.

**Definition.** In S-AUS all SNs have uniform travel time \(r\) from
the BS and a common AoI cost function \(f(\cdot)\). The battery,
when fully charged, allows for an operation time of \(2r\).

We use the term trip to refer to getting to one SN from the
BS, collecting the information, and returning to the BS. By
the definition of S-AUS, a schedule of the UAV has to consist
of a sequence of trips.

**Lemma 2.** For S-AUS, if the UAV performs one trip starting
at time \(t_0\) in time interval \([t_0, t_1]\) where \(t_1 - t_0 \geq 2r\), then the
trip to the SN having the largest AoI is optimal.

**Proof.** Denote by \(h_i\) the average AoI cost of SN \(i \in S\) for
interval \([t_0, t_1]\) with duration \(\Delta t = t_1 - t_0\). This cost can be
calculated as:

\[
h_i = \frac{1}{\Delta t} \left( \sum_{s \in S \setminus \{i\}} \int_{a(s, t_0)}^{a(s, t_0) + \Delta t} f(x) dx + \int_{a(i, t_0)}^{a(i, t_0) + 2r} f(x) dx + \int_{r}^{\Delta t - r} f(x) dx \right)
\]

(2)

Denote by \(i^*\) the SN with largest AoI. It can be verified
that:

\[
h_{i^*} - h_i = \frac{1}{\Delta t} \left( \int_{a(i, t_0) + 2r}^{a(i, t_0) + \Delta t} f(x) dx - \int_{a(i^*, t_0) + 2r}^{a(i^*, t_0) + \Delta t} f(x) dx \right)
\]

(3)

Since \(a(i, t_0) \leq a(i^*, t_0)\) and \(f\) is monotonically increasing,
the result of (3) is non-negative and we have \(h_{i^*} \leq h_i\).

As all trips are of same length for S-AUS, we consider
scheduling the UAV for any number of consecutive trips. In
the following, we present and prove the optimal solution.

**Theorem 3.** Consider S-AUS and \(M\) consecutive trips, visiting
the SN with the largest AoI in each trip results in optimum
average AoI.

**Proof.** Denote by \(t_1, t_2, ..., t_M\) the time points of departures
from the BS for trips 1, 2, ..., \(M\), respectively. Consider any solution \(q\) in which \(k\) is the index of the first trip that the
UAV does not visit the SN with largest AoI. Consider another
solution \(q'\) which is the same as \(q\) except that trip \(k\) goes
to the SN with largest AoI. We show that \(q'\) gives a lower
cost than that of \(q\). Consider trips 1, 2, ..., \(k\). Both solutions
Moreover, arc \(q\) gives a lower cost than that of \(q'\). Therefore, \(q'\) gives a lower cost for trips 1, 2, ..., \(k\). It is easy to see that 
\[a'(t_{k+1}) \geq a'(t_{k+1})\] 
where \(a'(t_{k+1})\) is the age value associated with solution \(q'\). For trip \(k + 1, ..., M\), as AoI vector of \(q\) is larger than that of \(q'\) and the same SN are visited in both \(q\) and \(q'\), the AoI cost of \(q\) obviously will not be lower than that of \(q'\) after any of the trips. Hence, \(q'\) gives a lower overall AoI cost. This establishes the theorem. 

IV. ALGORITHM DESIGN

The complexity of AUS motivates the use of sub-optimal algorithms. However, it is desirable to design an algorithm that inherently enables to turn optimality against complexity. To this end, we develop a graph labeling algorithm (GLA) enabled by time slicing. For AUS, we construct a graph in which finding an optimal path provides an approximate solution of AUS. GLA is shown in Algorithm 1.

A. Graph Representation

We slice the time horizon \(T\) into a set of \(N = \{1, 2, \ldots, N\}\) time slots, each of length \(\tau = \frac{T}{N}\). Slot \(n \in N\) is defined for time interval \([(n-1)\tau, n\tau]\). The AoI of SN \(s \in S\) for slot \(n \in N\) is evaluated at the beginning of the slot, and the AoI remains for the entire slot. Hence, the approach provides an approximation of AUS where the solution accuracy depends on the granularity of time slicing.

We define a directed and acyclic graph \(G = (V, A)\). For slot \(n \in N\), we define \(S + 1\) nodes, giving in total \((S + 1) \times N\) nodes. Denote by \(v_{sn}\) the node defined for location \(s \in S^+\) and slot \(n \in N\) is evaluated at the beginning of the slot, and the AoI remains for the entire slot. Hence, the approach provides an approximation of AUS where the solution accuracy depends on the granularity of time slicing.

B. Label Creation

A label represents a partial solution. The idea is to create labels at graph nodes and store the promising ones. A label \(\ell\) in GLA is defined as a tuple of format \((B_\ell, M_\ell, z_\ell, a_\ell, h_\ell, \hat{h}_\ell)\) in which \(B_\ell\) is the remaining energy, \(M_\ell\) is a set recording the visited SNs since the most recent departure from the BS, \(z_\ell\) is a vector containing the timestamps of collected information of the SNs in \(M_\ell\), \(a_\ell\) is a vector containing the AoIs of SNs, \(h_\ell\) is the overall average AoI cost for the time slot of the label, and \(\hat{h}_\ell\) is another AoI related cost to be explained later.

We use matrix \(L\) of \((S + 1) \times N\) elements to store the labels. Entry \(L_{sn}\) is a set that stores the labels for location \(s\) and time slot \(n\). Denote by \(L_{sn}\) the number of stored labels in this entry. GLA stores a maximum of \(K\) labels for each entry. Using a larger \(K\), GLA stores more partial candidate solutions and hence potentially improves the overall solution. Thus, \(K\) can be tuned for the trade-off between solution quality and complexity.

GLA creates labels as follows. Consider node \(v_{sn}\) with label \(\ell\). If \(s = 0\) and the UAV stays for \(w\) time slots: If \(w \geq w_{min}\), battery is recharged, and \(B_\ell = B_\ell + g(w)\). If \(x < w_{min}\), battery will not be recharged, and this case is not considered. For the former, \(z_\ell\) and \(M_\ell\) are the same as those in \(\ell\), i.e., \(z_{\ell'} = z_\ell\) and \(M_{\ell'} = M_\ell\). All elements of \(a_\ell\) increase by \(w\tau\), i.e., \(a_{\ell'} = a_\ell + w\tau\). Finally, \(h_\ell = h_\ell + \tau + \sum_{\tau_{s} \in S} f_\tau(a_{\ell}(s) + i\tau)\). Here, \(h_\ell = h_\ell\). This case corresponds to Lines 6-11 in Algorithm 1.

Figure 2: Graph representation and an example with three SNs.
In summary, each time an arc is traversed that corresponds to moving from one node to another, a label of the starting node of the arc is used to result in a label at the end node. The content of the latter depends on the used label and the arc.

C. Label Domination

Instead of arbitrarily storing labels, GLA utilizes domination rules to eliminate labels that are either evidently non-optimal or less promising. The dominance check algorithm (DCA) is given in Algorithm 2. Consider a node \( v_{sn} \) defined for the BS in slot \( n \) and two labels \( \ell \) and \( \ell' \). If \( B_{\ell'} \leq B_\ell \) and \( h_{\ell'} > h_\ell \) (see Line 5 in Algorithm 2), then \( \ell' \) cannot lead to a better solution than \( \ell \), hence it should not be stored. In such a case, \( \ell \) dominates \( \ell' \).

For the labels defined for an SN, one can not conclude domination based on their actual Aol or Aol costs. For example, consider node \( v_{sn} \), and two labels \( \ell' \) and \( \ell \). Label \( \ell \) corresponds to a partial solution in which the UAV has stayed in the BS during time interval \([0, n - t_{0s}]\), and then it moved from BS to SN \( s \) during time interval \([n - t_{0s}, n]\). Label \( \ell' \) corresponds to a partial solution in which the UAV has visited many SNs during time interval \([0, n]\) without any return to the BS. Since in neither of the solutions any data is not delivered to the BS, both \( \ell \) and \( \ell' \) have the same Aol vectors and Aol costs, but \( \ell \) has more battery than \( \ell' \). Thus, \( \ell \) seemingly dominates \( \ell' \). However, \( \ell \) is in fact less promising in leading to a better solution than \( \ell' \). Because, in \( \ell \) the UAV only stays in the BS for most of time duration, while in \( \ell' \) data of many SNs are collected which intuitively should give a better solution than \( \ell \), even though one can not guarantee this. To deal with such scenarios, we use \( \hat{a} \) and \( \hat{h} \) for comparison instead of \( a \) and \( h \), and remove the labels that are less promising. We again use the term “domination”. Here, \( \hat{a} \) is a vector defined for the SNs, and for each SN, it contains the amount of time passed since data collection took place for this SN, i.e., \( \hat{a}(t) = t - z(t) \). \( \hat{h} \) is the cost calculated based on \( \hat{a} \) (see Line 2 in Algorithm 2), namely \( \hat{h} \) captures the Aol cost if the collected information were delivered to the BS. Note that for \( s = 0 \), we have \( u(t) = z(t) \), and hence \( \hat{a} = a \) and \( \hat{h} = h \).

DCA is shown in Algorithm 2. If a stored label \( \ell \) dominates a new label \( \ell' \), DCA discards \( \ell' \), see Line 5. Conversely, if \( \ell' \) dominates \( \ell \), DCA removes \( \ell \), see Lines 8-10. If \( \ell' \) is not discarded and less than \( K \) labels are stored, DCA stores \( \ell' \), see Lines 11-12. If none of these applies, there is no capacity to store more labels. In this case, if \( \hat{h}_\ell < \max_{\ell' \in \mathcal{L}_{s,n'}} \{\hat{h}_{\ell'}\} \), DCA removes the label with the maximum Aol cost and stores \( \ell' \) instead, see Lines 13-15. Otherwise, \( \ell' \) is discarded.

For complexity of DCA, calculating \( h_{\ell} \) is of complexity \( O(ST) \). Also, as a new label \( \ell' \) needs to be compared with a maximum of \( K \) labels, and each comparison concerns two vectors of size \( S \), the complexity is of \( O(KS) \). Thus, DCA is of complexity \( O(\max(ST, KS)) \). For the complexity of GLA, for each slot and node, there are a maximum of \( K \) labels, and for each label a maximum of \( S + N - 1 \) choices exist in which \( S \) is the number of candidate nodes to visit and \( N - 1 \) corresponds to staying in the BS of different lengths of time. As for each choice, the value of \( h_{\ell} \) needs to be calculated and DCA needs to be run, the complexity of GLA is of \( O((N - 1)(S + 1)K(S + N - 1)ST\max\{ST, KS\}) \).

**Algorithm 1: Graph labeling algorithm (GLA)**

**Input:** \( K, B, b, t, w_{\min}, f_s(\cdot) \) for \( s \in S \), \( g(\cdot) \)
**Output:** A schedule for AUS

\[
L_{00} \leftarrow \{B, \{0, \ldots, 0\}, \{0, \ldots, 0\}, \{\}, 0\}
\]

1. for \( n = 1 : N - 1 \) do
2. for \( s \in S^+ \) do
3. for \( \ell \in L_{sn} \) do
4. for \( s' \in S^+ \setminus \{M_n\} \) do
5. if \( s = 0 \) and \( s' = 0 \) then
6. if \( n' = n + w_{\min} : N \) do
7. if \( B_{\ell'} > B_\ell + g(n' - n) \)
8. \( z_{\ell'} \leftarrow z_{\ell}, a_{\ell'} \leftarrow a_\ell + (n' - n)\tau \)
9. if \( h_{\ell'} > h_\ell + \sum_{s \in S} \sum_{i=1}^{w_s} f_s(a_\ell(s) + i\tau) \)
10. \( \mathcal{L}_{\ell'} \leftarrow \text{DCA}(\ell', s, n, s', n', \{\ell, s, n\}) \)
11. if \( \tau(n + t_{0s} + t_{0s}') \leq T \) and \( b_{s'n'} + b_{s'n} \leq B_{\ell'} \)
12. \( h_{\ell'} \leftarrow h_\ell + \sum_{s \in S} \left( \sum_{i=1}^{w_s} f_s(a_\ell(s) + i\tau) + f_s(a_{s'n}(s)) \right) \)
13. \( \mathcal{L}_{\ell'} \leftarrow \text{DCA}(\ell', s, n, s', n', \{\ell, s, n\}) \)
14. \( \ell \leftarrow \text{arg max}_{\ell \in L_{0n}} f_\ell \)

**Algorithm 2: Dominance Check Algorithm (DCA)**

**Input:** \( l', s, s', n', L_{s,n'} \)
**Output:** \( L_{s',n'} \)

\[
L_{s',n'} \leftarrow \begin{cases} 1 & \ell \leftarrow 1, \hat{a}_{\ell'} \leftarrow \tau n' - z_{\ell}, i \leftarrow 1 \end{cases}
\]

\[
L_{s',n'} \leftarrow h_\ell + \sum_{s \in S} \left( \sum_{i=1}^{w_s} f_s(a_\ell(s) + i\tau) + f_s(\hat{a}_{\ell}(s)) \right) \)
\]

3. while \( (X = 1 \) and \( i \leq L_{s',n'} \)) do
4. \( \ell \leftarrow L_{s',n'}(i), \hat{a}_{\ell} \leftarrow \tau n' - z_{\ell}, i \leftarrow i + 1 \)
5. if \( (B_{\ell'} \geq B_{\ell} \) and \( a_{\ell'} \geq a_\ell \) \) or \( (B_{\ell'} < B_{\ell} \) and \( \hat{a}_{\ell'} \geq \hat{a}_\ell \) \) or \( (B_{\ell'} \leq B_{\ell} \) and \( \hat{a}_{\ell'} < \hat{a}_\ell \) \) then \( h_{\ell'} \geq h_\ell \)
7. \( X \leftarrow 0 \)
8. if \( X = 1 \) then
9. for \( \ell \in L_{s',n'} \) do
10. if \( (B_{\ell'} \geq B_{\ell} \) and \( \hat{a}_{\ell'} \leq \hat{a}_\ell \) \) or \( (B_{\ell'} > B_{\ell} \) and \( a_{\ell'} \geq a_\ell \) \) or \( (B_{\ell'} < B_{\ell} \) and \( \hat{a}_{\ell'} \leq \hat{a}_\ell \) \) then \( h_{\ell'} < h_\ell \)
11. Delete label \( \ell \) from \( L_{s',n'} \)
12. if \( L_{s',n'} < K \) then \( L_{s',n'} \leftarrow L_{s',n'} \cup \{\ell'\} \)
13. \( \mathcal{L}_{s',n'} \leftarrow \mathcal{L}_{s',n'} \cup \{\ell'\} \)
14. \( \mathcal{L}_{s',n'} \leftarrow \mathcal{L}_{s',n'} \cup \{\ell'\} \)
15. \( \ell \leftarrow \text{arg max}_{\ell \in L_{s',n'}} \{\hat{h}_\ell\} \)

V. PERFORMANCE EVALUATION

We evaluate the performance of GLA against a greedy algorithm (GA). The core idea of GA is that the data of the SN with the largest Aol and the smallest travel time will
be collected first. In each step, all time-feasible SNs that are not visited since the most recent departure from the BS are considered. Among them, GA selects the SN with the highest AoI cost with respect to the travel time. The UAV goes to the SN if this leads to a lower overall AoI cost than going back to the BS. Otherwise, the UAV examines the next SN. If none of the SNs leads to a lower AoI cost, the UAV goes to the BS for data delivery.

We consider a UAV with battery capacity of 25 min of flying time, and a recharging time of 50 min [9]. As in [3] we consider a circular area of radius 5000 m. The SNs are randomly located such that the travel time between any two locations is at least 0.5 min. The travel times are calculated based on a UAV velocity of 1200 m/min [3], [5]. The length of slots is set to \( \tau = 1 \) min. We have used several linear and non-linear functions [10], [11] to model the AoI costs of SNs.

Figs. 3-4 show the performance results. Fig. 5 shows the impact of duration of scheduling time horizon. As it can be seen the AoI cost for both algorithms increases with respect to \( T \) where after some time the AoI cost become stable. When \( T = 25 \), GLA outperforms GA by 12\% and this increases to about 28\% when \( T = 150 \).

Fig. 4 shows the impact of number of SNs. Clearly, as can be seen larger number of SNs results in higher AoI cost. When \( S = 5 \), GLA outperforms GA by about 9\%, and this increases to about 35\% when \( S \) increases to 25. Because a larger number of SNs results in a more difficult problem, and hence more difficult for GA to maintain the quality of solution.

Fig. 5 shows how normalized solution quality and solution time increase with respect to \( K \). Increasing \( K \), which means a higher computing time, has very clear impact on solution improvement. There is however a saturation effect (when \( K \) grows beyond 10), which is likely attributed to that the performance of GLA is approaching global optimality.

VI. CONCLUSION

This paper has studied UAV scheduling for data collection while accounting for the battery capacity of the UAV. The objective is to minimize the overall average AoI cost over a given time horizon. In addition to complexity analysis, we have proposed a tailored graph labeling algorithm featuring a mechanism for a trade-off between complexity and solution quality, and with a performance that clearly goes beyond that of greedy scheduling.

REFERENCES

[1] Y. Zeng, R. Zhang, and T. J. Lim, “Wireless communications with unmanned aerial vehicles: opportunities and challenges,” IEEE Communications Magazine, vol. 54, no. 5, pp. 36–42, 2016.

[2] S. Kaul, R. Yates, and M. Gruteser, “Real-time status: How often should one update?”, in IEEE INFOCOM, 2012, pp. 2731–2735.

[3] J. Liu, X. Wang, B. Bai, and H. Dai, “Age-optimal trajectory planning for UAV-assisted data collection,” in IEEE INFOCOM WKSHPS, 2018, pp. 553–558.

[4] P. Tong, J. Liu, X. Wang, B. Bai, and H. Dai, “UAV-enabled age-optimal data collection in wireless sensor networks,” in IEEE ICC WKSHPS, 2019, pp. 1–6.

[5] Z. Jia, X. Qin, Z. Wang, and B. Liu, “Age-based path planning and data acquisition in UAV-assisted IoT networks,” in IEEE ICC WKSHPS, 2019, pp. 1–6.

[6] V. Tripathi, R. Talak, and E. Modiano, “Age optimal information gathering and dissemination on graphs,” in IEEE INFOCOM, 2019, pp. 2422–2430.

[7] M. A. Abd-Elmagid and H. S. Dhillon, “Average peak age-of-information minimization in UAV-assisted IoT networks,” IEEE Trans. Veh. Technol., vol. 68, no. 2, pp. 2003–2008, 2019.

[8] M. R. Garey and D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman & Co., 1990.

[9] B. Galkin, J. Kibilda, and L. A. DaSilva, “UAVs as mobile infrastructure: Addressing battery lifetime,” IEEE Commun. Mag., vol. 57, no. 6, pp. 132–137, 2019.

[10] Y. Sun, E. Uysal-Biyikoglu, R. D. Yates, C. E. Koksal, and N. B. Shroff, “Update or wait: How to keep your data fresh,” IEEE Trans. Inf. Theory, vol. 63, no. 11, pp. 7492–7508, 2017.

[11] Y. Sun and B. Cyber, “Sampling for data freshness optimization: Non-linear age functions,” IEEE J. Commun. Networks, vol. 21, no. 3, pp. 204–219, 2019.