Wireless sensor network data collection by connected cooperative UAVs

Peng Wei[1], Quanquan Gu[2] and Dengfeng Sun[1]

***Abstract*— *Wireless sensor network* is a prevailing research topic in recent years. It is adopted in the scenario of monitoring environmental parameters, which is normally expensive or even impossible to monitor by human labor or other technologies. At the same time, another popular topic is *Unmanned Aerial Vehicle* (UAV), which is widely used in military, commercial and civilian activities. In this paper cooperative UAVs form a team to accomplish the data collection task on wireless sensor network, where the technologies in wireless sensor network and UAV are integrated together. We study the novel wireless sensor network data collection with UAVs by considering the cluster load balancing and the connectivity of UAVs. We implement an *Iterative Balanced Assignment with Integer Programming* (IBA-IP) algorithm for efficient UAV deployment and sensor assignment. The authors analyze the advantages of IBA-IP compared to the Iterative and Adaptive (ITA) algorithm developed in [1]. In order to approximate the performance bound, we solve the problem by applying the *Genetic Algorithm* (GA). Finally, simulation results are presented under different parameter settings and the performances of the IBA-IP algorithm and the Genetic Algorithm are evaluated.**

I. Introduction

One of the wireless sensor network (WSN) applications is environmental monitoring because in some cases, it is impossible or very costly to monitor a field by human. The diverse applications need low cost, long life time, durable wireless sensors which can be deployed and installed throughout the field.

After the deployment of the wireless sensors, it is necessary to retrieve the sensing data and information from the sensors. The traditional way is to collect the data to a stationary base station by multi-hop routing [2], [3]. The base station usually has longer transmitting range and much more memory space than sensors. Then the base station can process the data and transmit the collected data to user end. However, sometimes it is not feasible or efficient to collect data by such a stationary single sink multi-hop routing scheme.

An emerging alternative is to use a single UAV or a team of UAVs to collect data from the sensors. One reason is that using air mobile elements for data collection in a huge area has more advantages than traditional multi-hop communication schemes in terms of the sensor battery lifetime [4]. On the other hand, UAV is the ideal means for collecting data from sensors at inaccessible locations or in scenarios where ground vehicles have reduced mobility [5]. Despite these significant advantages, nevertheless, the literature research work on UAVs for WSN data collection are still rare.

Yuan et al. formulate the problem of sensor data collection using a single robot as a special instance of the Travelling Salesperson Problem with Neighborhood (TSPN) [6]. Dunbabin et al. present an underwater data muling system with multiple robots in [7]. Tekdas et al. focus on energy efficiency issues of land-based system [8]. Some other related work can be found in the area of mobile backbone network deployment. In this paper, we study the wireless sensor data collection facilitated by a team of connected UAVs with the load balancing consideration.

A team of connected UAVs is more robust for failures because others can still perform data collection task when one or several UAVs malfunction. Furthermore, a team of UAVs can collect data faster than a single UAV. Based on [9], [10] one of the major delay factors in a random access based wireless network is the hop number from the transmitting node to the receiving node. Without multiple UAVs scheme, a sensor uploads data to the single base station or the single UAV. Hence there will be more hops between the source and the only sink, which will introduce high transmission delay.

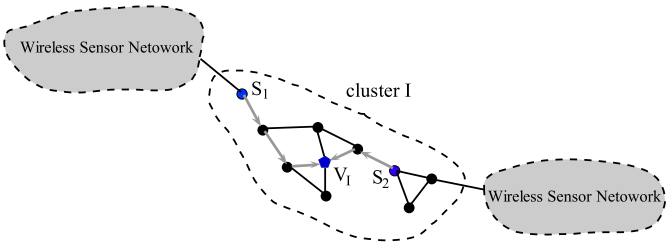

Fig. 1: Data upload inside cluster I.

Therefore, it is desirable to introduce a team of UAVs in order to provide multiple data collection sinks. Each sensor can be assigned to a relatively closer UAV, which ensures fewer transmission hops and shorter data collection time. To do that, wireless sensors will be divided into several clusters and each UAV is set as the data sink of one cluster. In this way the data collection can be done in a parallel style, which makes the data collection process faster. As shown in Fig. 1, the wireless sensor network is divided into multiple clusters. Inside cluster I, sensor s_1 uploads data to its assigned UAV v_I through 3 hops and sensor s_2 needs 2 hops. Similarly, all the other sensors in cluster I upload data to v_I through their own paths.

Normally a team of UAVs should form a connected network. First they need to know whether other UAVs are functioning properly and to share information about which

[1] P. Wei and D. Sun are with the School of Aeronautics and Astronautics, Purdue University, West Lafayette, IN 47907, USA {weip,dsun} @purdue.edu

[2] Q. Gu is with the Department of Computer Science, University of Illinois at Urbana-Champaign, IL 61801, USA qgu3@illinois.edu

sensors are associated to which UAVs. Second, the UAVs need to keep connected to assure synchronization.

Given a wireless sensor network layout, the aim of this work is to optimally determine the deployment positions of the UAVs and the wireless sensor assignment to UAVs. Our objective is to minimize the data collection time while satisfying the UAV connection constraint. The difficulty here is twofold. On one hand the minimal data collection time not only comes from fewer hops from sensor to its UAV but also the load balancing issue in each cluster. On the other hand, the deployed sensor network is an unweighted graph while the links in the connected UAV network are established based on Euclidean distance weight, which makes the two networks cannot be formulated in the same domain. We propose the Iterative Balanced Assignment with Integer Programming (IBA-IP) algorithm to solve the problem. In addition, in order to approximate the performance bound, we also solve the problem by implementing Genetic Algorithm.

The rest of the paper is organized as follows. In Section II, we formulate the problem and discuss its complexity. In Section III, we present the IBA-IP algorithm. We solve the problem by Genetic Algorithm in Section IV. In Section V we evaluate and compare the performances of our algorithms via simulations. Section VI concludes this paper.

II. Problem Formulation

According to [1], node i can reach node j if the *Signal to Interference and Noise Ratio* $\text{SINR}_{i,j}$ is larger than the threshold, which depends on the decoding capability of j. For a node pair of i and j, if i can reach j and j can reach i, there is a link between i and j. A path $p_{i,j}$ exists between two nodes i and j if they can reach each other directly through one link or over multiple links with other relay nodes. $p_{i,j}$ might not be unique.

n wireless sensors are given in the 2D plane constructing a connected network $G_S = (N_S, L_S)$. The set N_S contains all sensors, and $|N_S| = n$ is the size of the wireless sensor network. We denote the sensors as numbers $1, 2, ..., n$. The link between i and j is named as l_{ij}, and the set L_S contains all the links in the sensor network.

There are k UAVs to be deployed to collect the data of the wireless sensor network. Each UAV has one communication interface and one data transmission interface. The communication interface is used to communicate with other UAVs, and the data transmission interface is used to download the data from sensors. The two interfaces are tuned to different frequencies so communications can be carried in the wireless sensor network and the UAV network concurrently. After the deployment, the UAV network can be denoted as $G_V = (N_V, L_V)$, where N_V is the set of UAVs with size $|N_V| = k$, and L_V is the set of UAV links. We denote the UAVs as $v_1, v_2, ..., v_k$.

A. Hopping Time and Competing Time

For each sensor i, there will be an assigned UAV $v(i)$ for data upload. As the upload time of a sensor is directly impacted by the number of hops in the path to the assigned UAV, we consider the number of hops $h(i)$ between the sensor i and its UAV $v(i)$ as *hopping time*. Besides hopping time, if too many sensors upload their data to the same UAV, that UAV will become the *hot spot*, resulting in large *competing time*. In this paper, minimizing the hopping time is set as our objective function and the competing time will be restricted by load balancing constraint.

For the sensor i, $v(i) = v_I$ means that the sensor i is assigned to UAV v_I, where i and I are the indices for sensors and UAVs respectively. Let $h(i, v_I)$ be the hop number from sensor i to UAV v_I, and $h(i, v(i))$ be the hop number from sensor i to its assigned UAV $v(i)$.

The sensors assigned to UAV v_I form a cluster, and $|v_I|$ represents the number of sensors in this cluster. v_I is called the cluster head and $|v_I|$ is the size of this cluster. With k UAVs, there will be k clusters of sensors. In this work, every sensor is assigned to exactly one UAV, thus $\sum_{I=1}^{k} |v_I| = n$.

Before the deployment of UAVs, the actual signal strength can not be measured, therefore we adopt the transmission ranges (R for UAVs and r for sensors) to guide the UAV deployment. In general, a UAV obtains a longer transmission range than a wireless sensor ($R > r$). More accurately, SINR_{v_i,v_j} between two UAVs will be calculated based on Eq. (1) in [1], checked by a threshold at the receiving UAV to ensure the connectivity under fading conditions.

To capture the impact of hopping time, the upload time cost factor c_i between a sensor i and its assigned UAV $v(i)$ can be represented as:

$$c_i = h(i, v(i)). \tag{1}$$

To evaluate the overall UAV network deployment and sensors assignment, we consider the average UAV upload time cost factor $\bar{c}$ of all sensors, and the objective of the UAV network deployment and sensors assignment is to minimize $\bar{c}$ defined as:

$$\bar{c} = \frac{1}{n} \sum_{i=1}^{n} c_i = \frac{1}{n} \sum_{i=1}^{n} h(i, v(i)). \tag{2}$$

In order to avoid the hot spot and restrict the competing time, we formulate the load balancing constraint:

$$(\frac{n}{k} - \beta_-) \leq |v_I| \leq (\frac{n}{k} + \beta_+), \forall v_I \in N_V, \tag{3}$$

where smaller parameters β_- and β_+ are used to keep all the clusters more balanced. Larger β_- and β_+ can relax load balancing constraint to achieve better objective function value.

B. The Problem

Our problem is to deploy k connected UAVs and assign each sensor to one UAV, with the objective of minimizing the average UAV upload time cost and the constraint of

balancing each cluster:

$$\begin{array}{ll} & \min \bar{c} \\ \text{s.t.} & v(i) \in N_V, \forall i \in N_S, \\ & (\frac{n}{k} - \beta_-) \leq |v_I| \leq (\frac{n}{k} + \beta_+), \forall v_I \in N_V \text{ and} \\ & \forall v_I, v_J \in N_V, \exists P_{I,J}. \end{array} \quad \text{(P)}$$

A solution to this problem includes two parts: the deployment of k UAVs and the assignment of each sensor to a UAV.

In order to provide uploading target to a sensor, each sensor should be able to access exactly one UAV, either directly or through multi-hop sensor relays. Each UAV will approach to one sensor as reference location, which results in a limited number $\binom{n}{k}$ candidate deployment combinations in total. With the UAV network connection constraint, only some of these candidate deployment locations are feasible. So there will be less than $\binom{n}{k}$ types of deployment. Because k is known as a constant, the deployment solution has a polynomial complexity.

After the deployment of UAVs, the next step is to assign each sensor to a UAV. Since the sensor network topology is already known, it is possible to maintain the following hop number matrix H:

$$H_{(n\times n)} = \begin{pmatrix} h(1,1) & h(1,2) & \dots & h(1,n) \\ h(2,1) & h(2,2) & \dots & h(2,n) \\ \vdots & \vdots & \ddots & \vdots \\ h(n,1) & h(n,2) & \dots & h(n,n) \end{pmatrix}, \quad (4)$$

where the item $h(i,j)$ is the shortest path hop number between sensors i and j, and $h(i,i)$ is 0. H is a natural number symmetric matrix. Since our UAVs $\{v_1, v_2, ..., v_k\}$ have been already deployed very close to k selected sensors, the hop number from a sensor to every UAV can be found in the matrix H.

Note that given the deployment of UAVs as the candidate facility locations, and the hop number $h(i, v_I)$ from a sensor i to each UAV v_I as the connection cost, if we let $\beta_- = \frac{n}{k}$ and $\beta_+ = \frac{(k-1)n}{k}$ in Eq. (3), the problem after reduction is equivalent to the NP-hard k-facility location problem introduced in [11]. As a result, the assignment part of the problem is NP-hard and thus the entire problem in (P) is NP-hard.

III. Iterative Balanced Assignment with Integer Programming

In order to minimize the average UAV upload time cost while maintaining the load balancing inside each cluster and enforcing the connectivity of k UAVs, it will be ideal to keep UAVs as close as possible to the "center" of each cluster without violating the UAV connectivity constraint. Instead of exhaustively searching through all the possible $\binom{n}{k}$ combinations for UAV deployment and all the possible assignments between n sensors and k UAVs, which may take significantly long time, the authors propose a new algorithm.

A. Initial UAV Positions

A straightforward solution of initial UAV deployment is to randomly pick k sensors as the reference locations. However, this cannot guarantee that k UAVs are connected. Since the n sensors already form a connected network, we randomly select one sensor as the first UAV reference location $\boldsymbol{P}_{(0)1}$ (root) and then perform a *Breadth First Search* to traverse the sensor network to deploy the remaining $k-1$ UAVs. The $k-1$ UAVs are denoted as $\boldsymbol{P}_{(0)2}, \boldsymbol{P}_{(0)3}, ..., \boldsymbol{P}_{(0)k}$. As the transmission range of a UAV is longer than the one of a wireless sensor, the connectivity of the wireless sensor network ensures that the initial deployed UAVs form a connected network.

B. Balanced Assignment with Integer Programming

Given the UAV positions in round t, $\boldsymbol{P}_{(t)1}$,$\boldsymbol{P}_{(t)2}$,...,$\boldsymbol{P}_{(t)k}$, there is a need to optimally associate each sensor to a UAV to reduce the upload time to the UAV network. Since k-means clustering cannot deal with load balancing clustering, we propose the *Balanced Assignment with Integer Programming* (BA-IP) algorithm assigns the n sensors to k UAVs in each iteration. Although spectral clustering [12] can provide balanced clusters [13] and it can be computed in a decentralized manner [14], [15], the BA-IP algorithm is developed in this work as an alternative method.

Let $x_{i,j} = 1$ if sensor i is associated to UAV j and $x_{i,j} = 0$ if not. There are n sensors in total. For each sensor, we select one out of k UAVs to assign to. Constraint (5) shows that each sensor can only be assigned to one UAV.

$$\begin{array}{rcl} x_{1,1} + x_{1,2} + ... + x_{1,k} &=& 1, \\ x_{2,1} + x_{2,2} + ... + x_{2,k} &=& 1, \\ & \vdots & \\ x_{n,1} + x_{n,2} + ... + x_{n,k} &=& 1. \end{array} \quad (5)$$

Cluster load balancing is guaranteed in constraint (6), which makes sure that clusters have appropraciate balanced sizes. β_- and β_+ are used to adjust the trade-off between load balancing and better objective function value.

$$\begin{array}{ccccc} \frac{n}{k} - \beta_- & \leq & x_{1,1} + x_{2,1} + ... + x_{n,1} & \leq & \frac{n}{k} + \beta_+, \\ \frac{n}{k} - \beta_- & \leq & x_{1,2} + x_{2,2} + ... + x_{n,2} & \leq & \frac{n}{k} + \beta_+, \\ & \vdots & & \vdots & \\ \frac{n}{k} - \beta_- & \leq & x_{1,k} + x_{2,k} + ... + x_{n,k} & \leq & \frac{n}{k} + \beta_+. \end{array} \quad (6)$$

The objective function is minimizing the hop number from each sensor to its assigned UAV. Therefore, the integer programming formulation is:

$$\begin{array}{ll} & \min \sum_{i=1}^{n} \sum_{j=1}^{k} h(i, v_j) x_{i,j} \\ \text{s.t.} & (5), (6), \\ & x_{i,j} = \{0, 1\} \end{array} \quad (7)$$

where $h(i, v_j)$ can be looked up from the table in (4). This binary integer programming problem can be solved efficiently since n in our application is not large.

C. Update UAV Positions

After having the association problem settled, the UAV (cluster head) positions need to be updated. To do this, inside each cluster we select the sensor position which has minimal summation of hop numbers from other sensors in the same cluster. The hop numbers can be found in (4).

D. Checking the UAV network connectivity

After we update the UAV positions, there are two methods to check whether the UAV network is connected: constructing a k-node spanning tree and constructing the adjacent matrix. In this work, we take the first method as it has a lower computation complexity. The spanning tree construction can root from an arbitrary UAV. If the constructed spanning tree has a size k, the UAV network is connected; otherwise it is not connected.

As the determination of the UAV deployment is a virtual process, the connection between two neighboring UAVs needs to be calculated based on the link model in Eq. (1) of [1] with a safety threshold to ensure the connectivity under fading conditions.

E. Outlier Trim

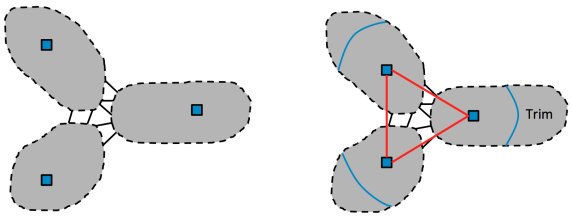

Fig. 2: Outlier Trim method establishes UAV connection.

In each iteration, after solving the balanced assignment with integer programming, the UAV positions are updated as the 3 blue squares shown in Fig. 2a. If the UAVs are far away to each other and not connected, the far end sensor nodes will be trimmed temporarily for UAV position update. As a result, the updated UAV positions will be pushed close enough to each other to establish connection (as shown in Fig. 2b).

Similarly as the hop number matrix (4), an Euclidean distance hop matrix $D_{n\times n}$ is built, in which each item $d(i,j)$ is the shortest Euclidean hop distance from sensor i to j. Inside each cluster, the q Outlier Trim method finds the q furthest sensors to its cluster head in terms of Euclidean hop distance and trims them temporarily for next iteration's BA-IP and UAV positions update.

F. Complete Algorithm

The complete algorithm is an iterative process through BA-IP, UAV positions update, connectivity check and q Outlier Trim as shown in Algorithm 1.

From lines 1 to 3, the positions of k UAVs are initialized, and BA-IP is performed in line 5 for sensor association. In line 6 the updated positions of the UAVs are determined. If the UAV network is not connected, the q Outlier Trim in line 9 is utilized to push the UAVs close to each other to obtain connectivity. At the end, another BA-IP is performed with all n sensors to determine the final sensor assignment.

Algorithm 1 Iterative Balanced Assignment with Integer Programming

1: n sensors, k UAVs
2: $\boldsymbol{P}_{(0)1}$ is randomly picked
3: obtain $\boldsymbol{P}_{(0)2}, ..., \boldsymbol{P}_{(0)k}$ by BFS
4: **while** $\exists \boldsymbol{P}_{(t)I}$!= $\boldsymbol{P}_{(t-1)I}$ **do**
5: run BA-IP
6: update $\boldsymbol{P}_{(t)1}, ..., \boldsymbol{P}_{(t)k}$
7: check UAV network connectivity (1 is connected and 0 is not)
8: **if** connectivity == 0 **then**
9: run q Outlier Trim
10: **end if**
11: $t = t + 1$
12: **end while**
13: output UAV deployment result
14: run BA-IP with all n sensors
15: output sensor assignment result

Compared to the *Iterative and Adaptive (ITA)* algorithm developed in [1], there are two major advantages of IBA-IP. First, the inside and after adjustment processes in ITA are removed, which would take long computation time. Second, by substituting *Random Greedy Assignment (RGA)* in [1] with BA-IP, the uncertainty is eliminated and the algorithm's performance becomes more consistent.

IV. Genetic Algorithm for Performance Bound

Genetic Algorithm [16] is a stochastic optimization algorithm based on the mechanisms of natural selection and natural genetic operation. It starts with a fixed-size population of S solutions $\{s_q|q = 1, 2, \ldots, S\}$ as the original generation. The solutions in GA evolve generation by generation. In this section, we present the implementation of GA to look for the global optimal solution, i.e., the performance bound of our algorithm.

As discussed earlier, since $R > r$ and also the objective of the deployment is to minimize the average UAV upload time cost, the UAVs will be deployed at the reference sensor positions. The total possible combinations of the deployment is $O(n^k)$, which can be checked exhaustively in polynomial time. We will find all the possible UAV deployment combinations first and then use GA to search in the assignment combinations to look for the optimal solution. In the following we introduce our design of applying genetic algorithm for achieving the optimal assignment for a given deployment combination, which consists of several typical steps.

A. Coding

Each solution s_q corresponds with a string of integer numbers with string length n, which represents one of the assignment results of n sensors. In a solution each sensor is assigned to one of the k UAVs from v_1 through v_k, and we use $1, 2, ..., k$ to represent the UAV which a sensor is assigned to. For each solution s_q, we can find out the hop number from each sensor to a UAV by looking up (4). As mentioned above, in each generation, there will be a fixed-size of S solutions.

B. Initialization

To start the GA, we need to set up an initial population of solutions. Normally, the initial solutions will be generated randomly, but in this case, total randomness may cause no sensor assignment to one or more UAVs. To avoid this problem, we first randomly pick k sensors and assign them to UAV from v_1 to v_k individually, and then assign the remaining $(n-k)$ sensors to the set of UAVs randomly.

C. Selection

GA selects the next generation with population size S from the previous generation of solutions, and pick one each time with the probability related to the fitness functions of the solutions in the previous generation. For example, the solution s_q is picked with the selection probability of $p_{\text{s}}(s_q)$:

$$p_{\text{s}}(s_q) = \frac{F(s_q)}{\sum_{i=1}^{p} F(s_q)}, \tag{8}$$

where $F(s_q)$ is the fitness function of solution s_q. We set $F(s_q)$ by applying the σ-truncation method [17] to the average upload time cost and the cluster load balancing constraint resulted by the assignment s_q. Similarly like [1], the fitness function of s_q is given by

$$F(s_q) = \begin{cases} g(s_q), & \text{if } g(s_q) \geq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

D. Crossover

Since we consider the deployment of a UAV network based on the existence of a sensor network, the objective function is not dynamic [18]. We thus decide not to adopt the crossover operation, because it will cause a longer running time and much more generations to converge to the optimum [18].

E. Mutation

For every generation, we divide the whole generation into two halves. We apply the *directional mutation* on the first half in order to speed up the convergence of the good solutions to their nearby local optimal value. At the same time, we apply the *random mutation* on the second half solutions to have some solutions get out of the bad local optimal.

1) Directional Mutation: For a solution s_q, the mutation of the jth item happens if the assignment for the jth sensor changes from its current UAV to another UAV v_I. The directional mutation probability p_{dm} is given by Eq. (11) in [1]. The directional mutation is greedy in probability and can speed up GA convergence.

2) Random Mutation: To prevent our solution being trapped to a local optimal, we apply the random mutation to let the solution move out of the local optimum. For a solution s_q, each item has a random mutation probability p_{rm} to change from its current UAV assignment to another, and probability $1 - p_{\text{rm}}$ to stay with the current UAV. Therefore, if some sensor of s_q is assigned to UAV v_I, then it will have the probability $\frac{p_{\text{rm}}}{k-1}$ to be assigned with $k-1$ other UAVs except the current v_I.

F. The Complete GA

Algorithm 2 Genetic Algorithm

```
while untested UAV deployment combination exists do
  pick an untested deployment combination
  if this UAV network is not connected then
    continue
  else
    set up an initial population P_0 with S solutions
    i = 1
    for i = 1 to G_t do
      P'_i = selection(P_{i-1})
      update the best c̄(s_q)
      split P'_i into P'_{i1} and P'_{i2}
      P_{i1} = Directional Mutation(P'_{i1})
      P_{i2} = Random Mutation(P'_{i2})
      P_i = P_{i1} ∪ P_{i2}
      i = i + 1
    end for
  end if
end while
output the best c̄(s_q)
```

GA has several ways of termination. We set G_t as the limit generation number, beyond which the algorithm will stop and report the best solution it has ever found. The GA solution is used as our simulated optimal bound to evaluate our IBA-IP's performance.

V. Simulation

Simulations are run to evaluate the performance of our proposed IBA-IP algorithm. 60 sensors are generated one by one at random locations. Each new sensor is ensured to get connected with the deployed ones. Therefore these sensors form a connected network with a random topology. To demonstrate our result, the UAV number k is set to 5. The UAV transmission range R is six times of the sensor transmission range r ($\frac{R}{r} = 6$). The coefficients β_- and β_+ in Eq. (3) are set to $0.2n/k$. And q in Outlier Trim method is set to 1.

A. Discussion on parameter k

With $R/r = 3$, the impact of UAV number k is shown in Fig. 3. The UAV number varies from 3 to 7 while other parameters stay the default values. IBA-IP has the close performance to the simulated lower performance bound provided by GA. When k increases, both the average upload time costs of IBA-IP and performance bound decrease because if there are more UAVs, the average sensor cluster size will be smaller thus each sensor can reach its assigned UAV with fewer hops.

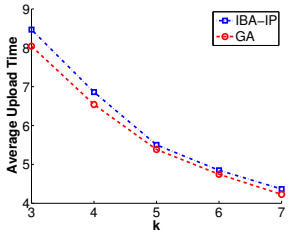

Fig. 3: The average upload time decreases with k increasing.

B. Discussion on transmission range ratio R/r

The impact of transmission range ratio R/r is shown in Fig. 4 with R/r varying from 2 to 7 and $k = 4$. Both curves of the average upload time decrease as R/r increases since with large R, the UAV network connectivity constraint is loosen up and more deployment candidates can be explored. When R/r becomes larger and larger ($R/r \geq 6$), the average upload time of IBA-IP and GA will achieve the same value. This is because the sensor network topology is fixed, no matter how much more R increases, the average upload time cost will stay at the same optimal value provided by the free UAV deployment without the connection constraint.

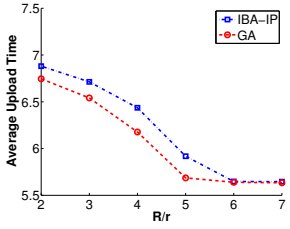

Fig. 4: The impact of R/r to the average upload time.

VI. Conclusion

In this paper we propose the Iterative Balanced Assignment with Integer Programming (IBA-IP) algorithm and the Genetic Algorithm to facilitate the wireless sensor network data collection with a team of cooperative UAVs. The objective of the UAV deployment and sensors assignment is to minimize the average upload time cost of all the sensors under the cluster load balancing and UAV connectivity constraints. We exploit the Genetic Algorithm to obtain the lower performance bound of the problem. We have run simulation to study the impact of different parameters on the performance of the IBA-IP algorithm and compared it with GA. The simulation indicates that the IBA-IP algorithm can achieve the performance close to that obtained by Genetic Algorithm. Our study shows that the proposed algorithm is efficient for the UAV deployment and for the balanced assignment of the wireless sensors. Our future work is to develop the exact mathematical performance bound for the problem in formulation (P).

References

[1] P. Wei, S. Chu, X. Wang, and Y. Zhou, "Deployment of a reinforcement backbone network with constraints of connection and resources," in *The 30th International Conference on Distributed Computing Systems*, Genoa, Italy, June 2010.

[2] M. Chayon, T. Rahman, M. F. Rabbi, and M. Masum, "Automated river monitoring system for bangladesh using wireless sensor network," in *10th International Conference on Computer and Information Technology*, 2007, pp. 1–6.

[3] G. Werner-Allen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh, "Fidelity and yield in a volcano monitoring sensor network," in *7th USENIX Symposium on Operating Systems Design and Implementation*, Seattle, WA, Nov 2006.

[4] A. Somasundara, A. Ramamoorthy, and M. Srivastava, "Mobile element scheduling for efficient data collection in wireless sensor networks with dynamic deadlines," in *Proceedings of 25th IEEE International Real-Time Systems Symposium*, 2004, pp. 296–305.

[5] J. A. Cobano, J. R. Martinez-Dios, R. Conde, J. M. Sanchez-Matamoros, and A. Ollero, "Data retrieving from heterogeneous wireless sensor network nodes using uavs," *Journal of Intelligent and Robotic Systems*, March 2010.

[6] B. Yuan, M. Orlowska, and S. Sadiq, "On the optimal robot routing problem in wireless sensor networks," *IEEE Trans. on Knowl. and Data Eng.*, vol. 19(9), pp. 1252–1261, 2007.

[7] M. Dunbabin, P. Corke, I. Vasilescu, and D. Rus, "Data muling over underwater wireless sensor networks using an autonomous underwater vehicle," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2006, pp. 2091–2098.

[8] O. Tekdas, N. Karnad, and V. Isler, "Efficient strategies for collecting data from wireless sensor network nodes using mobile robots," in *In 14th International Symposium on Robotics Research (ISRR)*, 2009.

[9] R. M. de Moraes, H. R. Sadjadpour, and J. J. Garcia-Luna-Aceves, "On mobility-capacity-delay trade-off in wireless ad hoc networks," in *Proc. IEEE Computer Society's 12th Annual International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS'04)*, Oct. 2004.

[10] B. Liu, P. Thiran, and D. Towsley, "Capacity of a wireless ad hoc network with infrastructure," in *Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*, 2007, pp. 239–246.

[11] P. Zhang, "A new approximation algorithm for the k-facility location problem," *Theoretical computer Science*, vol. 384, no. 1, pp. 126–135, Sep 2007.

[12] U. von Luxburg, "A tutorial on spectral clustering," *Journal Statistics and Computing*, vol. 17, no. 4, Dec 2007.

[13] R. Andersen, D. Gleich, and V. Mirrokni, "Overlapping clusters for distributed computation," in *Proceedings of the fifth ACM international conference on Web search and data mining*, New York, NY, USA, 2012, pp. 273–282.

[14] D. Kempe and F. McSherry, "A decentralized algorithm for spectral analaysis," *Journal of Computer and System Sciences*, vol. 74, no. 1, pp. 70–83, 2 2008.

[15] T. Sahai, A. Speranzon, and A. Banaszuk, "Hearing the clusters in a graph: A distributed algorithm," *Automatica*, vol. 48, no. 1, pp. 15–24, 2012.

[16] J. H. Holland, *"Adaptation in natural and artificial systems"*. MI: Univ. of Michigan Press, 1975.

[17] D. E. Goldberg, *"Genetic Algorithms in Search, Optimization, and Machine Learning"*. Addison-Wesley Professional, 1989.

[18] W. Rand, R. Riolo, and J. H. Holland, "The effect of crossover on the behavior of the ga in dynamic environments: a case study using the shaky ladder hyperplane-defined functions," in *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, Jul. 2006, pp. 1289–1296.