MHCDMC Rouding

Qinghui Zhang[1], Weidong Li[2], Qian Su[1] Xuejie Zhang[1, *]

1. School of Information Science and Engineering, Yunnan University, Kunming 650504, China

2. School of Mathematics and Statistics, Yunnan University, Kunming 650504, China

November 8, 2022

Abstract

abstract

keywords: high performance computing; resource allocation; scheduling; approximation algorithm; bag-of-tasks

*Correspondence: xjzhang@ynu.edu.cn (X. Zhang)

1 Introduction

With their maneuverability and increasing affordability, unmanned aerial vehicles (UAVs) have many potential applications in wireless communication systems [1]. In particular, UAV-mounted mobile base stations (MBSs) can be deployed to provide wireless connectivity in areas without infrastructure coverage such as battlefields or disaster scenes. Unlike terrestrial base stations (BSs), even those mounted on ground vehicles, UAV-mounted MBSs can be deployed in any location and move along any trajectory constrained only by their aeronautical characteristics, in order to cover the ground terminals (GTs) in a given area based on their known locations.

After a major natural disaster, the ground-based communication facilities are usually destroyed and communication is interrupted, and important communication information is blocked, which endangers the lives of the affected people and aggravates the difficulty of post-disaster rescue. UAVs have wide application prospects in the field of emergency communication because of their advantages such as rapid deployment and the ability to provide effective air-ground line-of-sight links to cover the affected areas by equipping emergency base stations [1].

In order to protect people's life and property and speed up the post-disaster reconstruction and recovery work, we need to provide communication security for users as soon as possible. According to the actual communication demand, some potential optional UAV deployment locations are selected by relying on information such as population distribution and disaster situation. Selecting the minimum number of UAVs to restore the communication network in that area under the constraints of communication needs is a critical issue. Since larger UAVs have larger energy reserves compared to smaller UAVs, larger UAVs can transmit signals with higher power and have greater bandwidth capacity to obtain better signal coverage performance. Therefore, in this paper we assume that the UAV base station with higher signal transmitting power has a larger bandwidth capacity.

contributions:

- We improved the algorithm proposed by Bandyapadhyay in [1] when the require of user is real.We also analyze the relationship between SINR and power variation on ε in the wireless communication model is analyzed.
- We propose a center-of-gravity-based hierarchical position presetting algorithm to preset the UAV position.

- Since the number and location of deployed UAVs are unknown, the SINR between each user and UAV is preset in this paper. After multiple solving, the gap between the preset value and the real value can be reduced significantly. We evaluated the gap between the pre-set SINR and the real SINR as an indication of the accuracy of the UAV base station deployment sites.

2 Related work

related work

3 System model

In a disaster area, we have pre-planned m loci and possible MBSs to be deployed in. Although deploying MBSs in each location can satisfy the communication needs of all users, it is very inefficient and impractical. We need to select as few loci as possible to deploy the corresponding MBSs as soon as possible to restore communication services for n users. We denote the set of users and MBS loci by U and A, respectively. For each user $u_j \in U$, there is a bandwidth requirement BR_j. The signal transmit power of MBS $a_i \in A$ is p_i, and the bandwidth resource is limited to BW_i. In order to resume communication for all users as soon as possible while ensuring that the communication signal-to-noise ratio of all users is not less than $SINR_{min}$, we need to select as few deployed MBSs as possible. We define the problem as a minimum metric capacitated signal coverage (MMCSC) problem. We introduce decision variables x_{ij} and y_i. y_i denotes whether to choose a_i as the final MBS deployment policy, and when $y_i = 1$ indicates that MBS a_i is chosen. x_{ij} is used to indicate whether to make a_i allocate bandwidth resources for u_j to provide services, and when $x_{ij} = 1$ indicates that u_j is served by a_i.

We can regard MMCSC problem as a capacitated bipartite graph cover (CBGC) problem. For a bipartite graph $G = (A, U, E)$, where A and U are the two sets of vertex, $E = \{e_{ij} | SINR_{ij} \geq SINR_{\min}, a_i \in A, u_j \in U\}$.The capacity of each vertex $a_i \in A$ is BW_i, the request of each vertex $u_j \in U$ is BR_j. The definition of CBGC problem is that choosing a subset $A' \subseteq A$ which is capacitated to cover all vertex in U. In this problem, each vertex $a_i \in A$ is like the source of *flow*, and each vertex $u_j \in U$ has flow demand. The flow of a_i can only flow to the connected u_j in G.

The communication between the UAV-enabled MBS and the user uses an air-to-ground communication link in the sub-6 GHz band, where Line of Sight (LoS) dominates. A The path loss between u_j and a_i can be expressed as:

$$L_{ij}(\mathrm{dB}) = 20\lg(d_{ij}) + 20\lg(\frac{4\pi f}{c}) + \eta_{LoS}, \tag{1}$$

where d_{ij} denotes the distance between a_i and u_j, f denotes the carrier frequency, c represents the speed of light, and η_{LoS} represents the shadow fading loss of LoS, which is a constant. the signal-to-noise ratio between a_i and u_j is:

$$SINR_{ij} = \frac{G_{ij}p_i}{N_I + N_0}, \tag{2}$$

G_{ij} denotes the channel gain between a_i and u_j, N_I denotes the interference noise power in this environment, and N_0 denotes the white noise power. The channel gain G_{ij} is affected by the path loss and satisfies the following relationship:

$$G_{ij}p_i(\mathrm{dB}) = p_i(\mathrm{dB}) - L_{ij}(\mathrm{dB}). \tag{3}$$

According to the above relationship, u_j's data rate DR_j can be expressed as

$$DR_j = BR_j \log_2(1 + SINR_{ij}). \tag{4}$$

Based on the above definition, we can get the integer programming form of the problem.

$$\begin{aligned}
\min \quad & \textstyle\sum_{a_i \in A} y_i && && (5)\\
s.t. \quad & x_{ij} \le y_i && \forall u_j \in U,\ \forall a_i \in A. && (5a)\\
& \textstyle\sum_{u_j \in U} (x_{ij} \cdot BR_j) \le y_i \cdot BW_i, && \forall a_i \in A. && (5b)\\
& \textstyle\sum_{a_i \in S} x_{ij} = 1, && \forall u_j \in U. && (5c)\\
& x_{ij} = 0, && \forall u_j \in U, \forall a_i \in A \text{ such that } SINR_{ij} < SINR_{\min}, && (5d)\\
& x_{ij} \in \{0,1\}, && \forall u_j \in U, \forall a_i \in A && (5e)\\
& y_i \in \{0,1\}, && \forall a_i \in A. && (5f)
\end{aligned}$$

Constraint (5a) means that u_j can be served by a_i only after the MBS a_i is selected. Constraint (5b) is the bandwidth resource capacity resource constraint of each MBS, the sum of bandwidth demand of users it serves cannot exceed its own capacity. Constraint (5c) of indicates that every user must be served. Constraint (5d) means that if u_j is served by a_i, then the SINR of u_j has to be greater than $SINR_{min}$. Constraints (5e) and (5f) are two integer decision variable constraints. We relax the integer programming (5) to be able to obtain its linear programming (6):

$$
\begin{aligned}
\min \quad & \sum\nolimits_{a_i \in A} y_i && && (6)\\
s.t. \quad & x_{ij} \le y_i, && \forall u_j \in U, \forall a_i \in A. && (6a)\\
& \sum\nolimits_{u_j \in U} (x_{ij} \cdot BR_j) \le y_i \cdot BW_i, && \forall a_i \in A. && (6b)\\
& \sum\nolimits_{a_i \in S} x_{ij} = 1, && \forall u_j \in U. && (6c)\\
& x_{ij} = 0, && \forall u_j \in U, \forall a_i \in A \text{ such that } SINR_{ij} < SINR_{\min}, && (6d)\\
& x_{ij} \ge 0, && \forall u_j \in U, \forall a_i \in A && (6e)\\
& 0 \le y_i \le 1, && \forall a_i \in A. && (6f)
\end{aligned}
$$

We can obtain the optimal solution $\sigma = (x, y)$ of LP 6 in polynomial time, which is a fractional solution. In this solution, for any $x_{ij} > 0$, we can ensure that $SINR_{ij} \ge SINR_{\min}$. In a real scenario, MBS can enhance its own signal coverage through spectrum multiplexing and other methods. We define $SINR'_{\min}(< SINR_{\min})$ as a new minimum SINR after using the multiplexing technique. When $SINR'_{min} \le SINR_{ij} \le SINR_{\min}$, a_i can also provide services with better signal quality for u_j by adopting multiplexing techniques. In addition, MBSS can also enhance the signal coverage by increasing the signal power. Let p'_i be the power after enhancement.

For the two variables $SINR'_{\min}$ and p'_i, they have a limited variation. In this paper, we do not focus on how small $SINR'_{\min}$ is or how large p'_i is; we only give the definition of the magnitude of their variable as follows.

$$\varepsilon^p = \max_{a_i \in A} \frac{{p_i}' - p_i}{p_i}, \tag{7}$$

$$\varepsilon^{SINR} = \frac{SINR_{\min} - SINR'_{\min}}{SINR_{\min}}. \tag{8}$$

Algorithm 1 Hierarchical Rounding

Input: $(U, A,)$.

Output: A set of selected MBS A^*.

1: $\sigma \leftarrow$ the solution of LP(6).

2: $\overline{\sigma} \leftarrow DSIS(\sigma)$

3: $\widehat{\sigma}, \mathcal{C} \leftarrow CoS(\overline{\sigma})$.

4: $A^* \leftarrow SFS(\widehat{\sigma}, \mathcal{C})$.

5: **return** A^*.

4 Rounding

In this section, we present an efficient rounding algorithm for the solution $\sigma = (x, y)$ of LP (6). The rounding algorithm consists of three parts, namely "determining superior and inferior servers", "clustering of servers" and "selecting the final servers". With Alg.1, we can finally get the set of the selected servers A^*. Finally, we demonstrate through theoretical analysis that the approximation ratio of the algorithm is () when the SINR and the power are varied by ε^p and ε^{SINR}, respectively.

Before presenting the above algorithm, we give here two important definitions.

Definition 4.1. *For a fraction solution $\sigma = (x, y)$ of LP.(6), given a parameter $0 < \alpha \leq 1/2$, and each server with non-zero y_i can be classified according to the relationship with α. If $y_i = 1$, we call a_i a superior server and denote their set by $\mathcal{S}$. If $0 < yi \leq \alpha$, we call a_i an inferior server and denote their set by $\mathcal{I}$.*

Definition 4.2. *Reroute the flow from a subset of servers $A' \subseteq A$ to a server $a_k \notin A'$, it means that the users $U' \subseteq U$ served by A' will be allocated to a_k to serve. A new solution $(\overline{x}, \overline{y})$ will get from current solution (x, y) as flowing way: (1) $\overline{x}_{kj} = x_{kj} + \sum_{a_i \in A'} x_{ij}, \forall u_j \in U'$; (2) $\overline{x}_{ij} = 0, \forall u_j \in U', \forall a_i \in A'$.*

4.1 Determining Superior and Inferior Servers

Since the number of inferior servers is directly related to the result of the rounding algorithm, this subsection is to trim the replaceable servers in $\mathcal{I}$. In addition, servers with $\alpha < y_i < 1$ are

Algorithm 2 Determining Superior and Inferior Servers (DSIS)

Input: A fraction solution of LP(6) σ, a parameter α and the magnitudes of variable of power and SINR ε^p and ε^{SINR}.

Output: A fraction solution $\overline{\sigma} = (\overline{x}, \overline{y})$ containing only superior and inferior servers.

Initialize $\overline{x} \leftarrow x$, $\overline{y} \leftarrow y$.
$\mathcal{S} \leftarrow \{a_i | \forall a_i \in A \text{ and } \overline{y}_i = 1\}$, $\mathcal{I} \leftarrow \{a_i | \forall a_i \in A \text{ and } 0 < \overline{y}_i \leq \alpha\}$.
for all $u_j \in U$ **do**
  if $\sum_{a_i \in \mathcal{I}: \overline{x}_{ij} > 0} \overline{y}_i > \alpha$ **then**
    Construct an set $\mathcal{I}_j$ of inferior servers serving u_j such that $\alpha < \sum_{a_i \in \mathcal{I}_j} \overline{y}_i \leq 2\alpha$.
    $r \leftarrow \max\limits_{a_i \in \mathcal{I}_j} r_i$, $\mathcal{T} \leftarrow \{a_i | r_i \leq \varepsilon r/2, \forall a_i \in \mathcal{I}_j\}$.
    Divide the set $\mathcal{I}_j \backslash \mathcal{T}$ of servers into $\lceil \log(1/\varepsilon) \rceil$ levels such that each lth level server a_i has $2^{l-1} r\varepsilon < r_i \leq 2^l r\varepsilon$, $0 \leq l \leq \lceil \log(1/\varepsilon) \rceil$.
    for $l = 0$ to $\lceil \log(1/\varepsilon) \rceil$ **do**
      Cut the square centered at u_j with side length $2^{l+2} r\varepsilon$ to grid cells, and each grid cell has side length $2^{l-2} r\varepsilon^2$.
      For each grid cell that contains at least one server, creating a group G_l consisting of all servers contained in the grid cell.
      $\mathcal{G}_l \leftarrow$ all groups in the lth level.
      for all $G_l \in \mathcal{G}_l$ **do**
        $a_m \leftarrow \arg\max_{a_i \in G_l} r_i$, reroute the flow from each $a_i \in G_l \backslash \{a_m\}$ to a_m.
        $\overline{y}_m \leftarrow 1$; $\overline{y}_i \leftarrow 0, \forall a_i \in G_l \backslash \{a_m\}$.
      end for
    end for
    Reroute the flow from $\mathcal{T}$ to server has maximum radii in $\mathcal{I}_j$.
  end if
end for
$\overline{y}_i \leftarrow 1, \forall a_i \in A, \alpha < y_i < 1$.
return $\overline{\sigma}$.

also added to $\mathcal{S}$ at the end of Alg.2.

4.2 Clustering of Servers

Clustering of Servers

Clustering of Servers

Clustering of Servers

Clustering of Servers

Clustering of Servers

Clustering of Servers

Clustering of Servers

Clustering of Servers

Clustering of Servers

Clustering of Servers

Clustering of Servers

Clustering of Servers

Clustering of Servers

Clustering of Servers

Clustering of Servers

4.3 Selecting the Final Servers

Selecting the Final Servers

5 Experimental Results

6 Conclusion and future work

We feel that the local assignment algorithm in Section will find application in related areas.

Acknowledgement

The work is supported in part by the National Natural Science Foundation of China [Nos. 61662088, 61762091], the Program for Excellent Young Talents, Yunnan University, and IRT-STYN.

References

[1] S. Bandyapadhyay, S. Bhowmick, T. Inamdar, and K. Varadarajan, "Capacitated covering problems in geometric spaces," *Discrete & Computational Geometry*, vol. 63, no. 4, pp. 768–798.

Algorithm 3 Clusting of Servers

Input: A fraction solution from Alg.2 $\overline{\sigma}$

Output: A fraction solution $\widehat{\sigma} = (\widehat{x}, \widehat{y})$, a set of clusters $\mathcal{C}$.

Initialize $\widehat{x} \leftarrow \overline{x}$, $\widehat{y} \leftarrow \overline{y}$, $\mathcal{O} \leftarrow \emptyset$.
$\mathcal{S}' \leftarrow \{a_i | \forall a_i \in A \text{ and } \widehat{y}_i = 1\}$; $\mathcal{I}' \leftarrow \{a_i | \forall a_i \in A \text{ and } 0 < \widehat{y}_i \leq \alpha\}$.
$C_i \leftarrow \{a_i\}, \forall a_i \in \mathcal{S}'$; $\mathcal{C} \leftarrow \{C_i | \forall a_i \in \mathcal{S}'\}$
while $\mathcal{I}' \neq \emptyset$ **do**
  for all $a_i \in \mathcal{S}'$, $a_t \in \mathcal{I}'$ **do**
    if a_t intersects a_i and $RB_i \geq \sum_{u_j \in U} \widehat{x}_{tj} BR_j$ **then**
      $C_i \leftarrow C_i \cup \{a_t\}$; $\mathcal{I}' \leftarrow \mathcal{I}' \setminus \{a_t\}$.
    end if
  end for
  if $\mathcal{I}' \neq \emptyset$ **then**
    $A_i \leftarrow \{u_j | \widehat{x}_{ij} > 0, \forall u_j \in U\}$, $\forall a_i \in \mathcal{I}'$.
    $k_i \leftarrow \min \left\{ BW_i, \sum_{u_j \in A_i} BR_j \right\}$; $a_t \leftarrow \arg\max_{a_i \in \mathcal{I}'} k_i$; $\mathcal{O} \leftarrow \mathcal{O} \cup \{a_t\}$; $\mathcal{I}' \leftarrow \mathcal{I}' \setminus \{a_t\}$.
    if $k_t \equiv \sum_{u_j \in A_t} BR_t \leq BW_t$ **then**
      For each $u_j \in A_i$, reroute the flow from $A \setminus \mathcal{O}$ to a_t.
      Update RB_t and $RB_i, \forall a_i \in \mathcal{S}'$.
    end if
    if $k_t \equiv BW_t < \sum_{u_j \in A_t} BR_t$ and $k_t \geq 2BR_t^{\min}$ **then**
      while Exists $u_j \in A_t : (1 - x_{tj}) BR_j \leq RB_t$ **do**
        $u_k \leftarrow \arg\min_{u_j \in A_t} [RB_t - (1 - x_{tj}) BR_j]$, reroute the flow of u_k from $A \setminus \mathcal{O}$ to a_t.
      end while
    end if
    if $k_t \equiv BW_t < \sum_{u_j \in A_t} BR_t$ and $BR_t^{\min} \leq k_t < 2BR_t^{\min}$ **then**
      $u_k \leftarrow \arg\max_{u_j \in A_t : BR_j \leq BW_t} BR_j$, reroute the flow of u_k from $\mathcal{I}'$ to a_t.
      $f \leftarrow \sum_{a_i \in \mathcal{O}} \widehat{x}_{ik}$, reroute the $\min\{RB_t / BR_j, (1 - f)\}$ amount of flow from $\mathcal{C}$ to a_t.
    end if
  end if
end while
$\widehat{y}_i \leftarrow 1$; $C_i \leftarrow \{a_i\}, \forall a_i \in \mathcal{O}$; $\mathcal{C} \leftarrow \mathcal{C} \cup \{C_i | \forall a_i \in \mathcal{O}\}$.
return $\widehat{\sigma}$, $\mathcal{C}$.

Algorithm 4 Selecting the Final Servers

Input: A fraction solution from Alg.3 $\widehat{\sigma}$, a set of clusters $\mathcal{C}$.

Output: Final solution $\widetilde{\sigma}$.

Initialize $\widetilde{x} \leftarrow \widehat{x}$, $\widetilde{y} \leftarrow \widehat{y}$.
$\mathcal{S}' \leftarrow \{a_i | \forall a_i \in A \text{ and } \widetilde{y}_i = 1\}$; $\mathcal{I}' \leftarrow \{a_i | \forall a_i \in A \text{ and } 0 < \widetilde{y}_i \leq \alpha\}$.
for all $a_h \in \mathcal{S}'$ **do**
  if $C_h \equiv \{a_h\}$ **then**
    $\widetilde{y}_h \leftarrow 1$.
  else
    $\mathcal{A}_1 \leftarrow \{a_i | r_i > r_h, \forall a_i \in C_h\}$; $\mathcal{A}_2 \leftarrow C_h \backslash \mathcal{A}_1$.
    $r_1 \leftarrow \max_{a_i \in \mathcal{A}_1} r_i$; $a_1 \leftarrow \arg\max_{a_i \in \mathcal{A}_1} r_i$.
    $\mathcal{T}_1 \leftarrow \{a_i | r_i \leq \varepsilon r_1/4, \forall a_i \in \mathcal{A}_1\}$; $\mathcal{T}_2 \leftarrow \{a_i | r_i \leq \varepsilon r_h/2, \forall a_i \in \mathcal{A}_2\}$.
    Using the hierarchical technique in alg.2, $\mathcal{A}_1 \backslash \mathcal{T}_1$ is divided into $O(\varepsilon^{-2} \log_2 1/\varepsilon)$ groups, and the flow from each group is reroute to the leader server.
    Reroute the flow from $\mathcal{T}_1$ to a_1.
    Cut the square containing $\mathcal{A}_2 \backslash \mathcal{T}_2$ with side length $O(r_h)$ to grid cells, and each grid cell has side length $\varepsilon^2 r_h/4$.
    For each grid cell that contains at least one server, creating a group G consisting of all servers contained in the grid cell.
    $\mathcal{G}_2 \leftarrow$ all groups from $\mathcal{A}_2 \backslash \mathcal{T}_2$.
    for all $G \in \mathcal{G}_2$ **do**
      Order the servers in G by non-increasing radii.
      $G' \leftarrow$ the first $\lceil \sum_{a_i \in G} y_i \rceil$ servers in this ordering.
      Reroute the flow from $G \backslash G'$ to G'.
    end for
    Reroute the flow from $\mathcal{T}_2$ to a_h.
  end if
end for
$\widehat{y}_i \leftarrow 1, C_i \leftarrow \{a_i\}, \forall a_i \in \mathcal{O}$;
return $\widehat{\sigma}$, $\mathcal{C}$.