BRAVE: Benefit-Aware Data Offloading in UAV Edge Computing Using Multi-Agent Reinforcement Learning

Odyssefs Diamantopoulos Pantaleon[a], Aisha B Rahman[a], Eirini Eleni Tsiropoulou[a]

[a] *Performance and Resource Optimization in Networks – PROTON Lab, School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, 85282, AZ, United States*

Abstract

Edge computing has emerged as a transformative technology in public safety and has the potential to support the rapid data processing and real-time decision-making during critical events. This paper introduces the BRAVE framework, a cutting-edge solution where the UAVs act as Mobile Edge Computing (MEC) servers, addressing users' computational demands across disaster-stricken areas. An accurate UAV energy consumption model is introduced, including the UAV's travel, processing, and hover energy. BRAVE accounts for both the users' Quality of Service (QoS) requirements, such as latency and energy constraints, and UAV energy limitations in order to determine the UAVs' optimal path planning. The BRAVE framework consists of a two-level decision-making mechanism: a submodular game-based model ensuring the users' optimal data offloading strategies, with provable Pure Nash Equilibrium properties, and a reinforcement learning-driven UAV path planning mechanism maximizing the data collection efficiency. Furthermore, the framework extends to collaborative multi-agent reinforcement learning (BRAVE-MARL), enabling the UAVs' coordination for enhanced service delivery. Extensive experiments validate the BRAVE framework's adaptability and effectiveness and provide tailored solutions for diverse public safety scenarios.

Keywords: Edge Computing, Data Offloading, Public Safety, Response Management, Multi-Agent Reinforcement Learning

1. Introduction

Natural disasters like earthquakes, landslides, and wildfires can result in extensive loss of life and property, along with significant economic repercussions for the impacted regions. The integration of Unmanned Aerial Vehicles (UAVs) with Artificial Intelligence (AI) technology is transforming disaster response strategies [1]. Due to their affordability, maneuverability, and ease of deployment, the UAVs can quickly access areas that are challenging to reach by other means. UAVs can act as Mobile Edge Computing (MEC) servers to collect data from the disaster areas and process them on board to support the search and rescue operations [2]. In this paper, the BRAVE framework is introduced to enable the UAVs to optimize their path planning for the effective data collection and processing in disaster-stricken areas. Specifically, the users in each disaster area autonomously determine their optimal amount of offloaded data to the visiting UAV, while the UAVs exploit a wide variety of reinforcement learning algorithms to determine their optimal path trajectories while balancing the objectives of maximizing the disaster response efficacy with their physical energy limitations.

1.1. Related Work

The problem of *data offloading to UAV-MEC-enabled systems* by exploiting AI-driven mechanisms has been thoroughly studied in the recent literature [3]. A collaborative MEC framework using multiple UAVs is introduced in [4] to minimize the task execution delays and the energy consumption by jointly optimizing the UAVs' trajectories, task allocation, and resource management through a cooperative multi-agent deep reinforcement learning (DRL) approach. A similar approach is followed in [5] by mainly focusing on the optimal placement of the UAVs to manage unexpected high traffic demands by proposing a low-complexity rule-based method and a Markov Decision Process (MDP)-based reinforcement learning (RL) approach. Several research works have focused on the energy efficient optimization of the data offloading process in UAV-MEC-enabled systems [6]. The authors in [7] focus on balancing the power consumption and the UAVs' flight efficiency while supporting the service offloading in smart city environments, by proposing a Mixed Integer Linear Programming optimization model. The problem of minimizing the combined communication, computation, and UAV flight energy costs is addressed in [8] based on the block coordinate descent and convex approximation methods. A generative neural networks-based data-driven heuristic framework is designed in [9] to efficiently optimize the deployment of UAV-based aerial MEC servers for optimizing the energy-efficient data offloading process.

Recent research efforts have been mainly focused on the problem of *determining the UAVs' optimal path planning to optimize the data collection and processing* while supporting a wide range of applications [10]. These approaches are organized in *collaborative and distributed methods* among the UAVs, promoting the UAVs' coordination [11] and distributed decision-making [12], respectively. A collaborative path planning algorithm is introduced in [13] based on the multi-agent Deep Q-Networks aiming at maximizing the data collection efficiency and minimizing the coverage overlap for the UAVs serving a set of areas. A similar collaborative path planning model is proposed in [14] based on an independent proximal policy optimization-based model targeting the minimization of the UAVs' energy consumption. Focusing on the UAVs' distributed decision-making regarding their optimal path planning, a single-level DRL model is analyzed in [15] to support the UAVs' energy-efficient path planning, optimal resource allocation and task offloading. A two-level DRL framework is designed in [16] to determine the UAVs' optimal trajectory to support high data acquisition success rates during the data collection process in complex environments with mobile and emergency nodes. A similar approach is analyzed in [17] by introducing a multi-objective optimization algorithm for UAV-enabled offloading that balances the UAVs' energy efficiency and safe path planning.

A small portion of the recent literature has extended its efforts in designing *tailored data collection and processing solutions for UAV-MEC systems assisting public safety operations* [18]. A multi-stage data offloading algorithm is proposed in [19] for optimizing the UAV-assisted MEC networks in disaster scenarios by balancing the computing power, user association, and service quality under various constraints. A similar approach is followed in [20] by proposing a gravitational search algorithm-based data offloading strategy to optimize the UAVs' energy, delay, and load balancing. A UAV scheduling scheme is designed in [21] to support emergency response operations by reducing the age of information and packet loss rate in the data collection process.

1.2. Contributions and Outline

Despite the significant advancements in UAV-MEC systems for data offloading, the existing literature primarily focuses on energy efficiency, task execution delay, and general-purpose data collection [22], without focusing on the design of tailored approaches specifically for disaster-stricken environments. Additionally, several collaborative and distributed decision-making methods have been explored, however, few research works introduce autonomous user-driven data offloading decisions coupled with UAV path optimization [23], especially under reinforcement learning frameworks aimed at maximizing disaster response efficacy.

The BRAVE framework addresses these gaps by enabling the users' autonomous decisions on data offloading in disaster areas, while the UAVs leverage diverse reinforcement learning algorithms to optimize their path planning and balance critical disaster response needs with their own energy constraints. The main contributions of this research work are summarized as follows.

1. A realistic public safety environment is introduced, where the UAVs act as flying MEC servers supporting the users' computing needs by visiting different disaster areas. An accurate and thorough UAVs' energy consumption model is analyzed, incorporating the UAVs' travel energy, processing energy, and hover energy. Also, the users' Quality of Service (QoS) requirements are captured accounting for their latency and energy consumption constraints.
2. The BRAVE framework is introduced to support the users' benefit-aware data offloading process and determine the UAVs' optimal path planning in public safety scenarios in order to maximize the data collection from multiple Areas of Interest (AoIs) while considering the UAVs' energy constraints. The BRAVE framework consists of a two-level decision-making mechanism.
3. The first level of the BRAVE's framework models the users' interactions in each AoI as a submodular game enabling them to autonomously determine their optimal data offloading strategies to the UAV that serves the AoI. The existence of a Pure Nash Equilibrium are shown.
4. The second level of the BRAVE's framework consists of a reinforcement learning-based optimal path planning mechanism that enables the UAVs to optimize their trajectory in order to maximize the data collection and processing support offered to the users, while considering the UAVs' energy constraints. A wide variety of RL mechanisms is proposed ranging from probabilistic mechanisms accounting for the AoIs needs for data processing (BRAVE-PRO) to softmax-like proba-

bility distributions (BRAVE-EXPO) and Q-learning mechanisms (Q-BRAVE).

5. The BRAVE framework is further extended to a collaborative setting where multiple UAVs coordinate with each other to support the public safety operations by engaging in a multi-agent RL mechanism (BRAVE-MARL) to determine the UAVs' optimal path planning in order to maximize the efficacy of the provided edge computing services to the disaster-stricken areas.
6. Detailed experiments have been performed to demonstrate the operational characteristics and benefits of the different BRAVE framework's variants to enable the Emergency Control Centers to adopt the solution that better fits the needs of each public safety scenario.

The remainder of this paper is organized as follows. The BRAVE system model is analyzed in Section 2, while the users' optimal data offloading mechanism is discussed in Section 3. The BRAVE-GREEDY framework, which is used for benchmarking purposes, is designed in Section 4, while the RL-based BRAVE variants are introduced in Section 5. The collaborative BRAVE-MARL framework is proposed in Section 6. Section 7 discusses the relevance of our work to the simulation modeling and practice, providing a detailed discussion regarding the simulation modeling performed in this paper. Detailed experiments are presented in Section 8 and Section 9 concludes the paper.

2. System Model

A public safety scenario is considered, where a natural or man-made disaster has occurred in a set of Areas of Interest (AoIs) $\mathcal{A} = \{1, \ldots, a, \ldots, A\}$. In each area a, a set of users $\mathcal{N}_a = \{1, \ldots, n_a, \ldots, N_a\}$ resides, who collect data and offload them to a flying UAV u, which acts a MEC server, for further data processing. The users' collected data are of interest to the Emergency Control Center (ECC), which coordinates the emergency response operation.

Each user $n_a \in \mathcal{N}_a, \forall a \in \mathcal{A}$ has a computation task to be completed $\mathcal{T}_{n_a} = (B_{n_a}, \Phi_{n_a})$, where B_{n_a} [bits] denotes the user's data and $\Phi_{n_a}[\frac{CPU-Cycles}{bit}]$ is the computational intensity of the user's task. Each user can choose how much data to offload to the UAV, each time the UAV visits the AoI where the user resides, from its strategy set $\mathcal{S}_{n_a} = \{s_{n_a,min}, \ldots, s_{n_a,j}, \ldots, s_{n_a,max}\}$ where $s_{n_a,j} \in [0, 1]$ is the percentage of the overall data of the user.

The UAV is equipped with a limited energy supply $\mathcal{E}$, starting from an initial AoI a_0, and must traverse a sequence of AoIs to gather and process data in order to support the emergency response operation, ending at a destination node a_{final}. The objective of the UAV is to identify a data collection and processing path $P = \{a_0, a_1, a_2, \ldots, a_{final}\}$, such that the total data gathered along the path, $B_P = \sum\limits_{a \in P} B_a$, is maximized while ensuring that the total cost of the path, $E_P = \sum_{k=1}^{k-1} w(a_k, a_{k+1}) \leq \mathcal{E}$, does not exceed the UAV's available energy $\mathcal{E}$. Specifically, consider a fully connected weighted graph $G = (\mathcal{A}, L)$, where $\mathcal{A}$ denotes the set of AoIs and L represents the set of links. Each link $(a_k, a_{k+1}) \in L$ has an associated weight $w(a_k, a_{k+1})$, and every AoI $a \in \mathcal{A}$ has an amount of data $B_a = \sum\limits_{n_a \in \mathcal{N}_a} B_{n_a} \geq 0 \in \mathbb{R}^+$. The associated weight captures the UAV's energy spent to hover over the AoI for a short time period to collect the users' data, travel from AoI a_k to AoI a_{k+1} and the energy spent to process the users' offloaded data in the AoIs a_k and a_{k+1}. Thus, the corresponding weight is defined as $w(a_k, a_{k+1}) = E_{UAV}^{hover} + E_{UAV}^{trav} + E_{UAV}^{proc}$. It is noted that in the extreme scenario where $a_0 = a_{final}$, the UAV starts and finishes at the same AoI. Also, we assume that the AoIs are far enough from each other, thus, while the UAV traverses from one AoI to another, the users' offloaded data at the AoI a_k are processed until the UAV reaches the AoI a_{k+1}.

The UAV's energy cost to hover over an AoI for a fixed duration of time t_{hover} [sec] in order to collect the users' data is estimated as follows [24]:

$$E_{UAV}^{hover} = (4.917H - 275.204)t_{hover} \tag{1}$$

where H [m] is the altitude at which the UAV is hovering at each AoI a.

The UAV has to expend a certain amount of energy to move from one AoI a_k to another a_{k+1}. According to [24] the energy expended to fly from one location to another is equal to:

$$E_{UAV}^{trav} = 308.709t_{trav} - 0.85 \tag{2}$$

where t_{trav} [sec] is the time needed for the UAV to cover the corresponding distance $d_{a_k,a_{k+1}}$ [m]. We consider that the UAV hovers above the center of the AoI, which has (x_a, y_a, z_a) coordinates. Additionally, the UAV is characterized by a velocity $\mathbf{v} = (x_v, y_v, z_v)$ [m/s]. Therefore, the UAV's travel time among the AoI a_k and a_{k+1} is $t_{trav} = \frac{d_{a_k,a_{k+1}}}{v}$, where $d_{a_k,a_{k+1}} =$

$\sqrt{(x_{a_k} - x_{a_{k+1}})^2 + (y_{a_k} - y_{a_{k+1}})^2 + (z_{a_k} - z_{a_{k+1}})^2}$ and $v = \sqrt{x_v^2 + y_v^2 + z_v^2}$ [m/s].

The UAV can potentially collect $B_a = \sum_{n_a \in \mathcal{N}_a} B_{n_a}$ data from an AoI a that it visits. Considering the users' optimal data offloading strategies $s^*_{n_a,j}$ (as analyzed in Section 3 below), the UAV ultimately processes $B^*_a = \sum_{n_a \in \mathcal{N}_a} B^*_{n_a}$ amount of data, where $B^*_{n_a} = s^*_{n_a,j} B_{n_a}$. At the time that the UAV makes its decision regarding its optimal path planning, the exact amount of data offloaded by the users is unknown and becomes available only upon the UAV's arrival at the AoI. Thus, the UAV accounts for a worst-case scenario regarding the energy expenditure required to process the users' total data in the AoI, represented as follows [25], [26]:

$$E^{proc}_{UAV} = \xi f^2_{UAV} \sum_{\forall n_a \in \mathcal{N}_a} \Phi_{n_a} B_{n_a} \tag{3}$$

where $f_{UAV}[Hz]$ is the frequency of the CPU of the UAV's MEC server and $\xi[\frac{J \cdot s}{\frac{CPU-cycles}{s}}]$ is the energy coefficient.

3. Users' Optimal Data Offloading

The users at each AoI collect data from the disaster area and they offload them to the UAV, when the latter one visits the AoI, for further processing. The data rate experienced by each user $n_a \in \mathcal{N}_a$ is derived as follows:

$$R_{n_a} = W \log_2\left(1 + \frac{p_{n_a} g_{n_a}}{I_0 + \sum_{\forall n'_a \neq n_a \in \mathcal{N}_a,} p_{n'_a} g_{n'_a}}\right) \tag{4}$$

where W [Hz] denotes the available bandwidth for the communication between the UAV and the users, I_0 [dBm/Hz] represents the power spectral density of zero-mean Additive White Gaussian Noise (AWGN), p_{n_a} [W] and g_{n_a} are the user's transmission power and channel gain in its communication link with the UAV, respectively.

Each user n_a experiences a time overhead to offload its data to the UAV and process them, while considering the UAV's shared computing capacity among all the users residing in the AoI. Thus, the user's experienced latency is derived as follows:

$$L_{n_a} = \frac{s_{n_a,j} B_{n_a,j}}{R_{n_a}} + \frac{\Phi_{n_a} s_{n_a,j} B_{n_a}}{[1 - \frac{\sum_{n'_a \neq n_a} s_{n'_a,j'} B_{n'_a}}{B_{UAV}}] \cdot f_{UAV}} \quad (5)$$

the first term captures the data offloading time overhead and the second term represents the data processing time overhead at the UAV. Also, B_{UAV} [bits] denotes the total amount of data that the UAV can process in parallel. It is noted that the second term of Eq. 5 shows that the UAV allocates its computing resources in a fair manner among the users, while accounting for the amount of their offloaded data.

Focusing on the users' experienced energy overhead due to the data offloading process to the UAV, its corresponding experienced energy consumption is derived as follows.

$$E_{n_a} = \frac{s_{n_a,j} B_{n_a}}{R_{n_a}} p_{n_a} \quad (6)$$

Based on Eq. 5 and Eq. 6, each user's normalized overhead is derived as follows:

$$O_{n_a} = \frac{L_{n_a}}{T} + \frac{E_{n_a}}{e_{n_a}} \quad (7)$$

where T [sec] is the duration of a timeslot within the examined system and $e^{(t)}_{n_a}$ [Joules] denotes the user's mobile device's available energy.

Considering the user's experienced latency and energy overhead, its utility can be formulated as follows:

$$U_{n_a}(s_{n_a,j}, \mathbf{s}_{-n_a,j}) = b e^{\frac{s_{n_a,j}}{\sum_{\forall n'_a \neq n_a} s_{n'_a,j'}}} - c e^{O_{n_a}} \quad (8)$$

where $\mathbf{s}_{-n_a,j} = [s_{1,j}, \ldots, s_{n_a-1,j}, s_{n_a+1,j} \ldots, s_{n_a,j}]$ represents the offloading strategies of all other users in the system, excluding the user n_a. The physical meaning of the user's utility captures its satisfaction from offloading data to a UAV-mounted MEC server, while also incurring costs in terms of time and energy. Furthermore, the satisfaction and cost experienced by each user are affected by the data offloading decisions made by other users within the system. The parameters b and c, within the range $[0, 1]$, adjust the weight each user assigns to its satisfaction from processing its data at the UAV (the first part of Eq. 8) against the cost associated with this choice (the second part of Eq. 8). Additionally, due to the sensitivity of the distributed system's stability to slight shifts in offloading strategies across numerous users, an exponential form is employed in Eq. 8 to effectively model the satisfaction-cost balance and associated patterns.

Each user seeks to optimize its utility (Eq. 8) in order to balance the benefits of offloading data to the UAV-assisted MEC server against the associated overhead (Eq. 7). Therefore, the problem for each user is structured as a utility maximization problem, and is formulated as follows:

$$\max_{s_{n_a,j} \in \mathcal{S}_{n_a}} U_{n_a}(s_{n_a,j}, \mathbf{s}_{-n_a,j}) = be^{\frac{s_{n_a,j}}{\sum_{\forall n'_a \neq n_a} s_{n'_a,j'}}} - ce^{O_{n_a}} \tag{9}$$

The optimization in Eq. 9 reveals the interdependencies in offloading strategies among the users, leading to competitive behavior over the UAV-mounted MEC server's resources. Thus, this optimization problem is modeled as a non-cooperative game $G = \{\mathcal{N}_a, \{\mathcal{S}_{n_a}\}_{\forall n_a \in \mathcal{N}_a}, \{U_{n_a}\}_{\forall n_a \in \mathcal{N}_a}\}$, where $\mathcal{N}_a$ indicates the users in AoI a, $\mathcal{S}_{n_a}$ is each user's offloading strategy set, and U_{n_a} represents each user's utility.

The solution to this game should identify a Pure Nash Equilibrium (PNE) where each user optimizes its utility by choosing an optimal strategy $s^*_{n_a,j}$. If a PNE exists, no user benefits by unilaterally changing its chosen strategy $s_{n_a,j}$ while others remain fixed. The PNE concept is formalized as follows.

Definition 1. (Pure Nash Equilibrium) A data offloading vector $\mathbf{s}^* = [s^*_{1,j}, \dots, s^*_{N_a,j}], s^*_{n_a,j} \in \mathcal{N}_a$ is a PNE of the non-cooperative game G if for every user n_a, the following holds: $U_{n_a}(s^*_{n_a,j}, \mathbf{s}^*_{-n_a,j}) \geq U_{n_a}(s_{n_a,j}, \mathbf{s}^*_{-n_a,j})$, for all $s_{n_a,j} \in \mathcal{S}_{n_a}$.

According to Definition 1, a PNE guarantees the stable system performance of the interaction among the users and the UAV, and the users achieve their maximum perceived utility at the PNE. If no PNE exists, then, the system becomes unstable and the users cannot determine their optimal offloading strategies. We apply the principles of S-modular games to demonstrate that a PNE exists in the non-cooperative game G. Specifically, we show that the non-cooperative game is submodular, meaning that as one user increases its offloading to the UAV-mounted MEC server, the other users tend to reduce theirs, due to increasing competition. The submodular games inherently have at least one PNE [27]. We formalize this analysis in the theorem below.

Theorem 1. (Submodular Game) The non-cooperative game $G = \{\mathcal{N}_a, \{\mathcal{S}_{n_a}\}_{\forall n_a \in \mathcal{N}_a}, \{U_{n_a}\}_{\forall n_a \in \mathcal{N}_a}\}$ is submodular if, for all $n_a \in \mathcal{N}_a$, the following conditions are satisfied: (i) For each $n_a \in \mathcal{N}_a$, the strategy set $\mathcal{S}_{n_a}$ is a compact subset of Euclidean space; (ii) U_{n_a} is smooth within $\mathcal{S}_{n_a}$ and exhibits non-increasing differences, i.e., $\frac{\partial^2 U_{n_a}}{\partial s_{n_a,j} \cdot \partial s_{n'_a,j'}} \leq 0, \forall n_a, n'_a \in \mathcal{N}_a, n_a \neq n'_a, \forall j, j'$.

Proof. To show that the non-cooperative game G is submodular, we assume each user can offload any portion of its data to the UAV-MEC server. Thus, the strategy space $\mathcal{S}_a = [0, 1]$ is continuous and compact, while U_{n_a} is smooth. Furthermore:

$$\frac{\partial^2 U_{n_a}}{\partial s_{n_a,j} \cdot \partial s_{n'_a,j'}} = b \cdot \left(\frac{-1}{\left(\sum_{n'_a \neq n_a} s_{n'_a}\right)^2} + \frac{-1}{\left(\sum_{n'_a \neq n_a} s_{n'_a,j'}\right)^3} \cdot s_{n_a,j} \right) \cdot e^{\frac{s_{n_a}}{\sum_{n'_a \neq n_a} s_{n'_a,j'}}}$$
$$- c e^{O_{n_a}} \cdot \left(\frac{\Phi_{n_a} \cdot B_{n_a} \cdot \frac{B_{n'a}}{B_{UAV}}}{\left[1 - \frac{\sum_{n'_a \neq n_a} s_{n'_a,j'} B_{n'_a}}{B_{UAV}}\right]^2 \cdot f_{UAV} \cdot T} \right) \cdot (1 + O_{n_a}) \leq 0 \tag{10}$$

Thus, the game is submodular. □

Since the submodular games have a non-empty set of PNE points, we have shown that the non-cooperative game G has at least one PNE $\mathbf{s}^* = [s^*_{1,j}, \ldots, s^*_{N_a,j}]$.

Toward determining the PNE, we design a Best Response Dynamics (BRD) algorithm, where the users iteratively adjust their strategies to converge to the PNE. The best response strategy for each user is defined as follows:

$$BR_{n_a}(\mathbf{s}_{-n_a,j}) = s_{n_a,j} = \underset{s_{n_a,j} \in \mathcal{S}_{n_a}}{\arg\max} U_{n_a}(s_{n_a,j}, \mathbf{s}_{-n_a,j}) \tag{11}$$

In summary, the asynchronous BRD algorithm, presented in Algorithm 1, determines a PNE for the non-cooperative game G. The algorithm's complexity is $O(N_a \cdot Ite)$, where Ite is the number of iterations for convergence. Specifically, the complexity for the part of the utility calculation is $O(N_a)$ given the iterative nature of the algorithm over all the users N_a residing in an area a. Also, considering that the algorithm is iteratively repeated over Ite iterations until it converges, the overall complexity is $O(N_a \cdot Ite)$.

Algorithm 1 Asynchronous BRD Algorithm

```
Require: $\mathcal{N}_a$, $R_{n_a}$, $B_{n_a}$, $f_{UAV}$, $B_{UAV}$, $T$, $e_{n_a}$, $\mathcal{S}_{n_a}$, $\forall n_a \in \mathcal{N}_a$
Ensure: Pure Nash Equilibrium: $\mathbf{s}^*$
 1: Initialization: $ite = 0$, $Convergence = 0$, $\mathbf{s}|_{ite=0}$
 2: while $Convergence == 0$ do
 3:     $ite = ite + 1$
 4:     for $n_a = 1$ to $N_a$ do
 5:         Each user $n_a$ determines $s_{n_a,j}|_{ite}$
                w.r.t. $\mathbf{s}_{-n_a,j}|_{ite}$ (Eq. 11) and receives $U_{n_a}^{(ite)}\big(s^*_{n_a,j}|_{ite}, \mathbf{s}^{(t)}_{-n_a,j}|_{ite}\big)$
 6:     end for
 7:     if $s^*_{n_a,j}|_{ite} - s^*_{n_a,j}|_{ite-1} <= 10^{-15}$ then
 8:         $Convergence = 1$
 9:     end if
10: end while
```

4. BRAVE-GREEDY

The UAV u, acts as a MEC server, which travels among the AoIs $a, \forall a \in \mathcal{A}$, to collect the users' data B_a. The problem is to determine the optimal path of the UAV in order to collect the maximum amount of data, thus, optimize the benefit of the UAV-MEC server, considering the energy constraints of the UAV to perform the flight and process the data. We formulate the above problem as an Integer Linear Programming (ILP) problem, where $\delta_{a,a'}$ is a binary variable, showing if the AoIs a, a' belong to the UAV's path and the UAV moves from a to a', i.e., $\delta_{a,a'} = 1$, while $\delta_{a,a'} = 0$, otherwise. To manage the AoIs' sequence, we introduce positional variables $a_k \in \mathcal{A}\backslash\{a_0\}$, which indicate the order in which the AoIs are visited. We set a_0 as the starting AoI of the UAV's path and $a_k < a_{k+1}$ denotes that AoI a_k is visited before a_{k+1}. The ILP problem is formulated as follows:

$$\max \sum_{a \in \mathcal{A}\backslash\{a_{final}\}} \sum_{a' \in \mathcal{A}\backslash\{a_0\}} B_a \cdot \delta_{a,a'} \tag{12}$$

$$\text{s.t.} \qquad \delta_{a,a'} \in \{0,1\}, \forall a, a' \in \mathcal{A} \tag{13}$$

$$\sum_{a' \in \mathcal{A}\backslash\{a_0\}} \delta_{a_0,a'} = \sum_{a \in \mathcal{A}\backslash\{a_{final}\}} \delta_{a,a_{final}} = 1 \tag{14}$$

$$\sum_{a \in \mathcal{A}\backslash\{a_{final}\}} \sum_{a' \in \mathcal{A}\backslash\{a_0\}} w_{a,a'} \cdot \delta_{a,a'} \leq \mathcal{E} \tag{15}$$

$$2 \leq a_a \leq A, \quad \forall a \in \mathcal{A} \backslash \{a_0\} \tag{16}$$

$$a_a - a_{a'} + 1 \leq (A-1) \cdot (1 - \delta_{a,a'}) \tag{17}$$

The goal is to maximize the total amount of collected and processed data from the users residing in each visited AoI considering the UAV's resource constraints, as captured in Eq. 12. Eq. 13 shows that the UAV will or will not visit a path from an AoI a to an AoI a'. Eq. 14 ensures the path begins at AoI a_0 and ends at AoI a_{final}. Eq. 15 restricts the total energy cost of the path to stay within the given UAV's energy budget $\mathcal{E}$. Finally, Eq. 16–17 are the Miller-Tucker-Zemlin (MTZ) inequalities, ensuring that no subpaths occur and all selected AoIs are part of a single connected path [28].

The BRAVE-GREEDY Algorithm (Algorithm 2) prioritizes visiting the AoIs with the highest available data to be offloaded by the users, while the UAV stays within its energy availability constraints. Initially, all AoIs are sorted in descending order based on their amount of data (step 4). The algorithm proceeds iteratively (steps 5-14) where in each iteration, given the current AoI a_c and the remaining energy of the UAV $\mathcal{E}_c$, it evaluates if any AoI which still has data to offload to the UAV can be reached considering the UAV's energy availability (step 6). If such AoIs exist, the one characterized by the maximum amount of data is selected, and the corresponding information, such as the path, energy cost, amount of collected data so far, remaining available energy of the UAV, and next current AoI, is updated (steps 7-12). If multiple AoIs have the same amount of data, a random choice is made. The process terminates either when all AoIs have offloaded all their available data or when none of the remaining AoIs can be visited due to the limited energy availability of the UAV or, alternatively, if the UAV has already arrived at the final destination a_{final} (step 5). At this point, the algorithm proceeds to the destination AoI a_{final} and returns the final path, the remaining energy of the UAV, and the total amount of collected data (steps 15-19). The algorithm has a time complexity of $O(A^2)$. Specifically, the sorting process of the AoIs requires $O(A \cdot \log A)$, while the worst case scenario of the main iterative loop (steps 5-14) is $O(A^2)$, if the algorithm traverses each AoI (worst case scenario). Thus, the overall time complexity of the BRAVE-GREEDY algorithm is dominated by the main iterative loop resulting in the overall complexity $O(A^2)$.

Algorithm 2 BRAVE-GREEDY Algorithm

```
1: Input: $G = (\mathcal{A}, L)$, $a_0$, $a_{final}$, $\mathcal{E}$
2: Output: Path $P$ from $a_0$ to $a_{final}$, $E_P$, $B_P$
3: Initialization: $E_P^{initial} = 0$, $B_P^{initial} = 0$, $\mathcal{A}_0 = \mathcal{A} \backslash \{a_0, a_{final}\}$ set of AoIs
   that have available data, $a_c = a_0$ current visited AoI, $\mathcal{E}_c = \mathcal{E}$ current
   energy availability
4: Sort the AoIs with available data with respect to their amount of data
   $B_a, \forall a \in \mathcal{A}_0$ in decreasing order, with $a_{a_{\max}} = 1$ denoting the AoI with
   the largest amount of data
5: while ($\mathcal{A}_0 \neq \emptyset$ && $\{a | w(n, a_c) + w(a_c, a_{final}) \leq \mathcal{E}\} \neq \emptyset$ && $a_c \neq$
   $a_{final}$) do
6:   if $a_{a_{\max}} \in \{a \mid w(a_c, a) + w(a_c, a_{final}) \leq \mathcal{E}\}$ then
7:     $P = P \cup \{a_{a_{\max}}\}$
8:     $E_P = E_P + w(a_c, a_{a_{\max}})$
9:     $B_P = B_P + B^*_{a_{a_{\max}}}$
10:    $\mathcal{E} = \mathcal{E} - w(a_c, a_{a_{\max}})$
11:    Update $B_a$ based on the amount of the users' offloaded data and
       update $\mathcal{A}_0$ if $a_{\max}$ exhausted its available data
12:    $a_c = a_{a_{\max}}$
13:  end if
14: end while
15: $E_P = E_P + w(a_c, a_{final})$
16: $P = P \cup \{a_{final}\}$
17: $B_P = B_P + B^*_{a_{final}}$
18: $\mathcal{E} = \mathcal{E} - w(a_c, a_{final})$
19: return $E_P, P, B_P, \mathcal{E}$
```

5. Reinforcement Learning-based BRAVE

In this section, the path of the UAV is determined following a reinforcement learning (RL) approach. The UAV acts as an RL agent. The Markov Decision Process (MDP) is formulated to model the decision-making process of the UAV in a dynamic environment, where the UAV must decide on the next AoI to visit while considering both its energy constraints and the data collection requirements. The key components of the MDP are as follows:

- **State (s_t):** The state represents the UAV's current situation, which includes the AoI where the UAV is currently located and the remaining energy available for visiting other AoIs in the future. Thus, the state captures the UAV's position in the environment and its remaining energy, which directly influences its decision-making process.

- **Action (a_t):** The action corresponds to the UAV's decision at time t regarding which AoI to visit next. The action space is discrete and consists of a set of AoIs that the UAV can choose to visit. The UAV's decision is influenced by its energy constraints and the expected benefit of visiting each AoI (i.e., the amount of data to be collected).

- **Reward (r_t):** The reward is a numerical value that quantifies the UAV's experienced benefit by selecting a particular action. In this case, the reward is based on the amount of data collected at the selected AoI. The UAV receives a positive reward for collecting data, which incentivizes it to prioritize AoIs with higher amounts of data. Thus, the reward is designed to reflect the UAV's objective, i.e., the maximization of the amount of collected data from the field, while respecting energy constraints.

- **Policy:** The policy is a mapping from the UAVs' states to their corresponding actions, and it represents the UAV's decision-making strategy. The UAV's policy $\pi(s_t, a_t)$ defines the probability of selecting a particular action given its current state. The policy is learned through the reinforcement learning process, where the UAV iterates through episodes, receives rewards, and adjusts its policy to maximize the cumulative reward over time.

In this section, three alternative methods are analyzed. The first method, BRAVE-PRO, follows a gradient bandit approach where a soft-max distribution is applied. This distribution is proportional to the amount of data collected at the selected Area of Interest (AoI) and scaled probabilistically by the ratio $\frac{B_a}{\sum\limits_{a' \in \mathcal{A}} (B_{a'})}$, where B_a is the data available at AoI a. The second method, BRAVE-EXPO, also employs a gradient bandit approach. However, it uses an experienced benefit model, expressed as $\frac{\exp(\frac{B_a}{\sum\limits_{a' \in \mathcal{A}} (B_{a'})})}{\sum\limits_{a'' \in \mathcal{A}} (\exp(\frac{B_{a''}}{\sum\limits_{a' \in \mathcal{A}} (B_{a'})}))}$. Finally, the third method, Q-BRAVE, relies on a Q-Learning technique, where the UAV selects an AoI based on the learned Q-values. An overview of the BRAVE-PRO, BRAVE-EXPO, and Q-BRAVE is presented in Fig. 1

5.1. BRAVE-PRO

In BRAVE-PRO method, the UAV acts as an RL agent navigating through the AoIs, which have available data to offload to the UAV, aiming

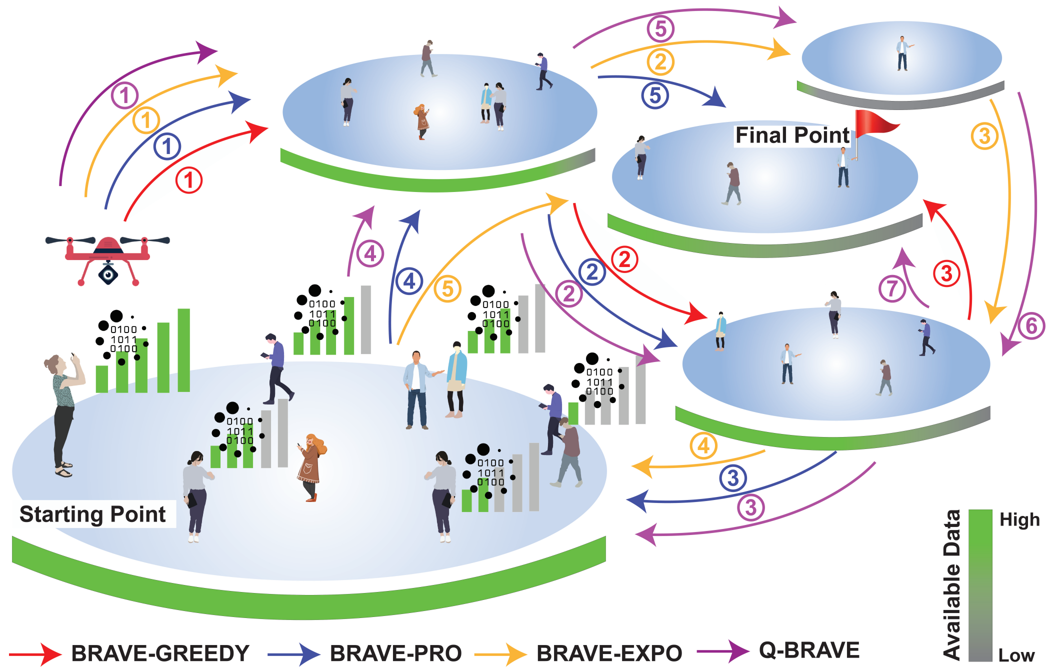


Figure 1: Overview of the BRAVE-PRO, BRAVE-EXPO, and Q-BRAVE frameworks.

at maximizing the amount of collected data. A probabilistic policy governs the action selection of the UAV, where the probability of moving from one AoI to the next one is calculated as follows.

$$\pi(s_t, a_t = a) = \frac{B_a}{\sum\limits_{a' \in \mathcal{A}_0} B_{a'}}, \tag{18}$$

This policy offers a straightforward approach for path selection. By normalizing B_a relative to the total data available across all AoIs, this rule ensures that higher data values directly influence the selection probability. The BRAVE-PRO method is computationally simple, however, this method tends to favor AoIs with the largest amount of available data and it reduces the probability of visiting smaller but still viable alternatives. This bias can result in suboptimal exploration (e.g., early termination of the UAV's navigation among the AoIs), especially in heterogeneous environments with skewed data distributions.

5.2. BRAVE-EXPO

In this section, we propose an advanced reinforcement learning-based approach, BRAVE-EXPO, for the UAV's optimal path planning. Unlike the conventional methods, such as the BRAVE-PRO method, BRAVE-EXPO leverages an exponential probability update mechanism to achieve a balanced trade-off between the exploration and the exploitation. This approach mitigates the biases that exist in the UAV's probabilistic decision-making, particularly in environments with highly skewed data distributions across the available AoIs.

The BRAVE-EXPO algorithm introduces a probability update rule based on a softmax-like function, defined as follows.

$$\pi(s_t, a_t = a) = \frac{\exp\left(\frac{B_a}{\sum\limits_{a' \in \mathcal{A}} B_{a'}}\right)}{\sum\limits_{a'' \in \mathcal{A}} \exp\left(\frac{B_{a''}}{\sum\limits_{a' \in \mathcal{A}} B_{a'}}\right)} \tag{19}$$

The adoption of the softmax-like function ensures a smoother probability distribution and reduces the dominance of AoIs with disproportionally large data values. Therefore, the UAV is encouraged to explore other viable AoIs with non-negligible data amounts, thus, the BRAVE-EXPO improves the overall system efficiency and promotes the exploration. The exponential reduces the influence of extreme values in B_a, consequently, it encourages the UAV to perform more balanced decision-making by performing more exploration. Moreover, by mitigating the over-reliance of the UAV on high-value AoIs, BRAVE-EXPO enables the UAV to maximize its data collection while at the same time maintaining an energy-efficient operation.

The complexity of the BRAVE-PRO and BRAVE-EXPO algorithms is $O(ITE \cdot A)$, where ITE denotes the number of iterations that the algorithms need to converge to the optimal path for the UAV. Specifically, the complexity of the main loop (steps 7-17) is $O(A)$ per iteration, as all the internal calculations of complexity $O(1)$ are performed for each AoI. The algorithm is executed iteratively over ITE iterations, where this number depends on the UAV's energy level, thus, the overall complexity is $O(ITE \cdot A)$.
The BRAVE-PRO and BRAVE-EXPO algorithms are jointly presented in Algorithm 3.

Algorithm 3 BRAVE-PRO/BRAVE-EXPO UAV Path Planning

```
1: Input: $G = (\mathcal{A}, L)$, $a_0$, $a_{final}$, $\mathcal{E}$
2: Output: Path $P$, Total Collected Data $B_P$, Final Energy $\mathcal{E}_c$
3: Initialization: $\mathcal{A}_0 = \mathcal{A} \setminus \{a_0\}$, $a_c = a_0$, $\mathcal{E}_c = \mathcal{E}$, $P = \{a_0\}$, $B_P = 0$
4: while ($\mathcal{A}_0 \neq \emptyset$ && $\{a | w(n, a_c) + w(a_c, a_{final}) \leq \mathcal{E}\} \neq \emptyset$ && $a_c \neq a_{final}$) do
5:     Compute probabilities $\pi(s_t, a_t = a')$ for all $a' \in \mathcal{A}_0$
6:     Select $a'$ using $\pi(s_t, a_t)$
7:     if $w(a_c, a') \leq \mathcal{E}_c$ then
8:         $P = P \cup \{a'\}$
9:         $B_P = B_P + B^*_{a'}$
10:        $B_{a'} = B_{a'} - B^*_{a'}$
11:        Update $\mathcal{A}_0$ if $a'$ exhausted its available data
12:        $\mathcal{E}_c = \mathcal{E}_c - w(a_c, a')$
13:        $a_c = a'$
14:    else
15:        $\mathcal{A}_0 = \mathcal{A}_0 \setminus \{a'\}$
16:    end if
17: end while
18: $P = P \cup \{a_{final}\}$
19: $B_P = B_P + B^*_{a_{final}}$
20: $\mathcal{E}_c = \mathcal{E}_c - w(a_c, a_{final})$
21: return $P, B_P, \mathcal{E}_c$
```

5.3. Q-BRAVE

In this section, we extend the BRAVE framework by introducing a Q-Learning-based path planning approach, i.e., the Q-BRAVE framework. It is noted that the BRAVE-PRO and BRAVE-EXPO frameworks employ probabilistic policies for the action selection, while, the Q-BRAVE framework leverages Q-learning, a model-free reinforcement learning algorithm, to determine the optimal path for the UAV. Q-learning enables the UAV to make decisions based on accumulated experience by learning an action-value function $Q(s_t, a_t)$. From any given state $s_t \in \mathcal{S}$, the UAV selects an action $a_t \in \mathcal{A}$ to move to a new state, while obtaining a reward $r_t(s_t, a_t)$. A policy, represented as $\pi(s_t) : \mathcal{S} \to \mathcal{A}$, maps the current state s_t to an appropriate action a_t. In this context, we assume a deterministic policy where a specific action is chosen based on the state. The agent refines its decision-making by evaluating the effectiveness of actions through the Q-value function $Q(s_t, a_t)$. This value reflects the anticipated discounted total of future rewards when performing action a_t in state s_t, under the assumption of following an optimal policy. The best action for any given state is the one that results in

Algorithm 4 Q-BRAVE UAV Path Planning

```
1: Initialize $Q(s,a) \leftarrow 0$ for all states $s \in \mathcal{S}$ and actions $a \in \mathcal{A}$
2: Set learning rate $\alpha$, discount factor $\gamma$
3: Set episodes, $\epsilon$-greedy policy
4: for iteration = 1 to episodes do
5:     Reset UAV to starting AoI $a_0$ and $G = (\mathcal{A}, L)$
6:     while ($\mathcal{A}_0 \neq \emptyset$ && $\{a | w(n, a_c) + w(a_c, a_{final}) \leq \mathcal{E}\} \neq \emptyset$ && $a_c \neq a_{final}$) do
7:         Select action $a'$ using $\epsilon$-greedy policy
8:         Calculate reward $r_t(s_c, a')$
9:         Update Q-value:
               $Q(s_c, a') \leftarrow (1-\alpha) \cdot Q(s_c, a') + \alpha \cdot [r_t(s_c, a') + \gamma \cdot \max_{a'} Q(s', a')]$
10:        $P = P \cup \{a'\}$
11:        $B_P = B_P + B^*_{a'}$
12:        $B_{a'} = B_{a'} - B^*_{a'}$
13:        Update $\mathcal{A}_0$ if $a'$ exhausted its available data
14:        $\mathcal{E}_c = \mathcal{E}_c - w(a_c, a')$
15:        $a_c = a'$
16:    end while
17: end for
18: return Optimal path and total collected data
```

the highest Q-value. When the RL agent is in state s_t, executes action a_t, and moves to the subsequent state, the value $Q(s_t, a_t)$ is adjusted iteratively using the Bellman equation:

$$Q(s_t, a_t) \leftarrow (1-\alpha) \cdot Q(s_t, a_t) + \alpha \cdot \left[r_t(s_t, a_t) + \gamma \cdot \max_{a'} Q(s'_t, a') \right] \quad (20)$$

In Eq. 20, $0 < \alpha \leq 1$ denotes the learning rate, while $\max_{a'} Q(s'_t, a')$ represents the highest possible reward obtainable from the subsequent state s'_t. The UAV starts at the source AoI a_0 and selects its next move by following the Q-table, specifically choosing the AoI $a' = \text{argmax}_{a'} Q(s', a')$, where the amount of data $B^*_{a'}$ is collected. This process continues until the RL agent arrives at the destination a_{final}. The complexity of the Q-BRAVE algorithm (Algorithm 4) is $O(episodes \cdot |\mathcal{S}| \cdot A)$, where *episodes* is the maximum number of episodes that the Q-BRAVE algorithm is executed. Specifically, the initialization phase (steps 1-3) has complexity $O(|\mathcal{S}| \cdot A)$. Then, the algorithm is executed over multiple *episodes*, following the for loop. The inner loop has an $O(A)$ complexity for the action selection based on the ϵ-greedy policy, while the reward calculation, Q-value update, state updates have all complexity $O(1)$. Thus, by combining the iterative process of the algorithm and considering that the order of magnitude of the states is comparable to the one of the states, the overall complexity is $O(episodes \cdot |\mathcal{S}| \cdot A)$.

6. BRAVE-MARL

In this section, we consider the general scenario, where multiple UAVs $u \in \mathcal{U}$ serve the AoIs and they collaborate among each other to support the search and rescue operation. To achieve this goal a Multi-Agent Reinforcement Learning (MARL) framework is introduced, named BRAVE-MARL. BRAVE-MARL extends Q-BRAVE into a multi-agent framework, structured into two distinct phases. The **autonomous exploration segment**, each RL agent independently follows a reward-oriented action strategy and updates the Q-table. In the **collaborative refinement segment**, the RL agents work together to adjust the reward and matrix entries, ensuring paths with higher amounts of collected data are encoded. An overview of the BRAVE-MARL is presented in Fig. 2.

6.1. Autonomous Exploration Segment

In this segment, the RL agents follow a reward-driven approach and update the Q-table individually. Specifically, when a UAV hovers above an AoI, it selects the next AoI to visit based on one of three options:

1. Greedy selection: The UAV/RL agent chooses the AoI offering the highest immediate reward based on Eq. 21 following an ϵ-greedy policy.

$$a' = \text{argmax}_{a' \in \mathcal{A}_0 \cap \mathcal{F}(r, \mathcal{E}_u)} Q(s_t, a') \tag{21}$$

   where $\mathcal{A}_0$ denotes the set of AoIs that still have data to offload to a UAV, and $\mathcal{F}(r, \mathcal{E}_u)$ is the set of feasible nodes within the agent's budget $\mathcal{E}_u$.
2. Exploration: The RL agent selects the next AoI probabilistically based on an ϵ-greedy policy.
3. Finalization: If no feasible AoIs exist to be visited, the RL agent terminates at the destination a_{final}.

After transitioning from AoI a to AoI a', the UAVs independently update the Q-table according to Eq. 20.

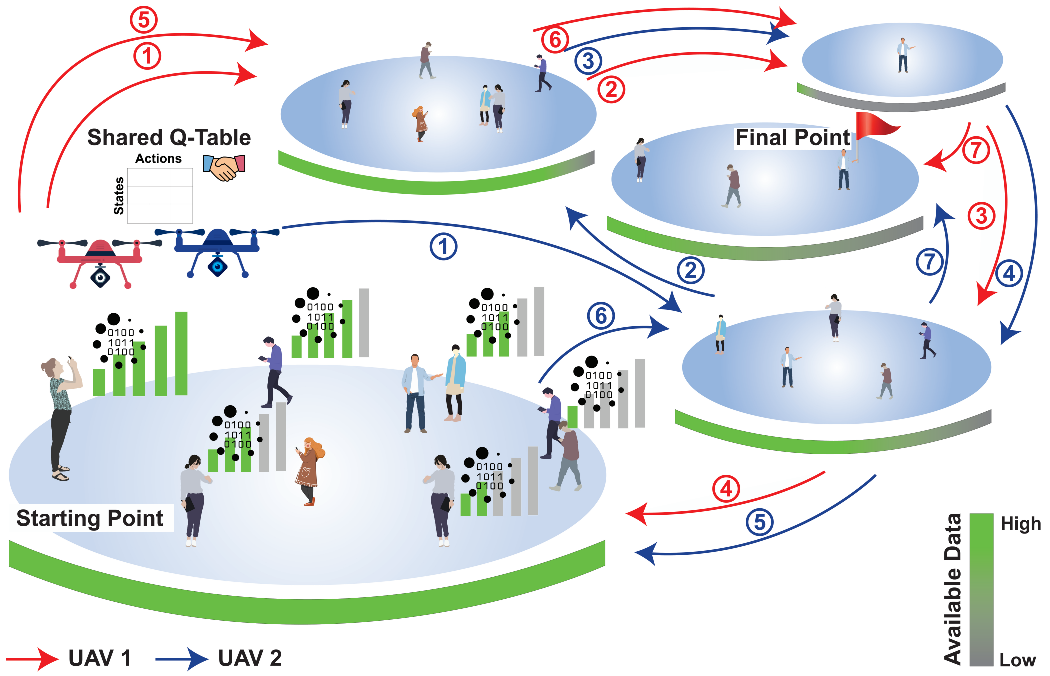


Figure 2: Overview of the BRAVE-MARL framework.

6.2. Collaborative Refinement Segment

In this phase, the UAVs communicate to update their collective Q-table in order to improve their knowledge about the environment. Based on the collaboratively updated Q-table, the UAVs enter the autonomous exploration segment phase to choose the next AoI to be visited. The UAVs repeat the autonomous exploration and the collaborative refinement segments until all the UAVs have reached their terminal conditions, i.e., the AoIs have offloaded all their data or no feasible AoIs can be visited due to energy constraints or the UAV has reached a_{final}.

The BRAVE-MARL algorithm is presented in Algorithm 5. The complexity of the BRAVE-MARL algorithm is $O(episodes \cdot |\mathcal{U}|^2 \cdot A)$. Specifically, during the autonomous exploration segment phase, each UAV operates independently, selects actions, and updates its Q-table. For each UAV, this process involves iterating over all possible actions and updating the Q-value for each state-action pair. This results in a complexity of $O(episodes \cdot A)$. Since the algorithm involves multiple UAVs, the overall complexity for this segment is $O(episodes \cdot |\mathcal{U}| \cdot A)$. Then, during the collaborative refinement segment phase, the UAVs exchange and refine their Q-tables. This involves communication between each pair of UAVs to update the collective Q-table, which introduces a pairwise interaction complexity of $O(|\mathcal{U}|^2)$. The refinement of the Q-tables is dependent on the number of UAVs and the size of the Q-table, resulting in a complexity of $O(episodes \cdot |\mathcal{U}|^2 \cdot A)$. Thus, by combining both segments and considering the number of episodes, the overall complexity is $O(episodes \cdot |\mathcal{U}| \cdot A + episodes \cdot |\mathcal{U}|^2 \cdot A)$ However, since the collaborative refinement segment's complexity scales quadratically with the number of UAVs, the total complexity can be approximated as $O(episodes \cdot |\mathcal{U}|^2 \cdot A)$.

Summarizing the analysis provided in Sections 5 and 6, it is noted that Section 5 introduces three single-agent reinforcement learning methods, i.e., (i) BRAVE-PRO: A gradient bandit approach using a soft-max distribution; (ii) BRAVE-EXPO: An advanced version of BRAVE-PRO using an exponential probability update mechanism; and (iii) Q-BRAVE: A Q-Learning based approach for optimal path planning. Section 6 builds upon these foundations to create BRAVE-MARL, i.e., a multi-agent reinforcement learning framework. BRAVE-MARL extends Q-BRAVE to a multi-UAV scenario by introducing the distinct phases of autonomous exploration and collaborative refinement and incorporating inter-agent communication and collective Q-table updates. BRAVE-MARL handles multiple agents simultaneously and it combines individual learning with collaborative refinement, thus, balancing the autonomy and teamwork among the UAVs. BRAVE-MARL algorithm adapts Q-Learning to a multi-agent environment, which is a non-trivial extension.

Algorithm 5 BRAVE-MARL Algorithm

```
1: Input: Set of UAVs $\mathcal{U}$, Set of AoIs $\mathcal{A}_0$, UAV energy budgets $\mathcal{E}_u$, $episodes$
2: Output: Path $P_u$, Total Collected Data $B_{P_u}$, Final Energy $\mathcal{E}_u$
3: for each iteration $ite = 1, 2, \ldots, episodes$ do
4:     Autonomous Exploration Segment:
5:     for each UAV $u \in \mathcal{U}$ do
6:         Initialize current state $s_t$ for UAV $u$
7:         while UAV $u$ has feasible AoIs $\mathcal{F}(r, \mathcal{E}_u)$ to visit do
8:             Select the next AoI $a'$ based on:
9:             (a) Greedy Selection: $a' = \text{argmax}_{a' \in \mathcal{A}_0 \cap \mathcal{F}(r, \mathcal{E}_u)} Q(s_t, a')$
10:            (b) Exploration: Select $a'$ probabilistically based on $\epsilon$-greedy
               policy
11:            (c) Finalization: If no feasible AoIs exist, terminate at $a_{final}$
12:            Transition from $a$ to $a'$, and update the Q-table $Q(s_t, a')$ using
               Eq. 20
13:            Update UAV state $s_t \leftarrow s_{t+1}$
14:        end while
15:    end for
16:    Collaborative Refinement Segment:
17:    for each UAV pair $(u_1, u_2) \in \mathcal{U}, \forall u_1 \neq u_2$ do
18:        Exchange Q-tables $Q_{u_1}$ and $Q_{u_2}$
19:        Update the collective Q-table by combining individual Q-tables con-
           sidering the largest entries
20:    end for
21:    for each UAV $u \in \mathcal{U}$ do
22:        Update individual Q-table $Q_u$ based on the collaboratively updated
           Q-table
23:    end for
24:    Check Termination Conditions:
25:    if all the UAVs have reached their terminal conditions, i.e., the AoIs
       have offloaded all data or no feasible AoIs can be visited due to energy
       constraints or the UAV has reached $a_{final}$ then
26:        Break the loop
27:    end if
28: end for
29: return $P_u$, $B_{P_u}$, $\mathcal{E}_u$, $\forall u \in \mathcal{U}$
```

7. BRAVE System Simulation and Modeling

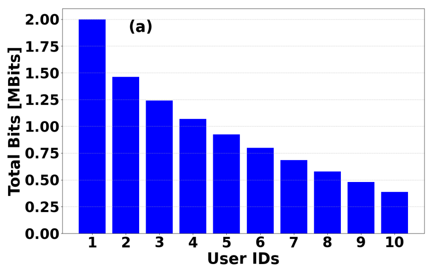
The simulation framework of the BRAVE framework was implemented using Python (3.10.14), leveraging its extensive libraries, such as scipy.minimize (1.14.1) to simulate the submodular game described in Algorithm 1, Jax and

Jaxlib (0.4.21) to execute and accelerate all necessary numerical operations (state-of-the-art technology that takes advantage of the system's GPU), concurrent.futures (Python multiprocessing library introduced in Python 3.2) for Monte Carlo simulations, gymnasium (0.29.1) to simulate the RL environment that the UAV interacts within each episode, and Matplotlib (3.9.2) for result visualization. The choice of Python was driven by its flexibility, scalability, and robust support for algorithm development and integration.

The network topology modeled in the simulation emulates a public safety system consisting of one and multiple UAVs for the BRAVE-GREEDY, BRAVE-PRO, BRAVE-EXPO, and BRAVE-MARL frameworks respectively. The UAVs' characteristics in terms of energy cost were captured in the simulation by providing them as inputs to the simulation, as they were derived from the UAVs' specification documentation that is available online. Parameters such as UAVs altitudes, AoIs positions, and user densities were configured based on real-world benchmarks, referenced from the Emergency Control Center of the City of Albuquerque, New Mexico, USA.

The proposed collaborative edge computing framework was implemented using a multi-agent reinforcement learning algorithm. The model incorporates a Stackelberg game-based optimization approach for resource allocation and task offloading decisions. MARL agents were designed to act as decision-makers for the UAVs, optimizing task scheduling by considering system constraints, such as limited computational power, dynamic user demands, and latency thresholds. The algorithm was initialized with hyperparameters derived from extensive literature review and fine-tuned through iterative testing.

To simulate realistic computing traffic patterns, synthetic task request data were generated following a Poisson distribution in order to reflect varying user request rates. These tasks were characterized by computational intensity and strict latency constraints, which represent realistic edge computing requirements. The evaluation metrics of the BRAVE frameworks included the UAVs' consumed energy, collected data, number of visited AoIs, execution time of the BRAVE frameworks, users' time overhead and total overhead, users' amount of offloaded data, users' devices' consumed energy, and users' achieved data rate. To ensure reproducibility, all parameter settings, and pseudocode for the algorithms are shared in the paper.



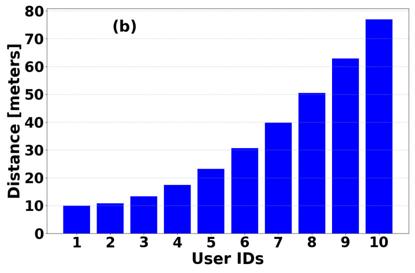


Figure 3: Users' total transmitted bits and distance from the AoI's center.

8. Numerical Results

In this section, a detailed evaluation of the BRAVE framework and its alternative models is presented to capture their pure operation as well as their benefits and drawbacks with respect to supporting search and rescue operations. Specifically, Section 8.1 analyzes the users' optimal data offloading based on the proposed game-theoretic approach. Section 8.2 provides a thorough analysis of the different BRAVE frameworks' alternative implementations by explaining their drawbacks and benefits with respect to the public safety events. Section 8.3 presents the realistic scenario of the Boston Marathon Bombing and how the BRAVE framework can support the search and rescue operation. Finally, Section 8.4 compares the BRAVE-MARL framework to alternative approaches that have been introduced in the literature to quantify its operational superiority. In the rest of the simulations, the following parameters have been adopted, $|\mathcal{A}| = 10$, $N_a = 10$, minimum distance between different AoIs is 100 [m], $H = 30$ [m], $\mathcal{E} = 19,800$ [KJ], $W = 5 \times 10^6$ [Hz], $f_{UAV} = 2 \times 10^9$ [Hz], $\Phi_{n_a} \in [1,2][\frac{KCPU-Cycles}{bit}]$, $\mathbf{v} = (1,1,1)$ [m/s], $T = t_{hover} = 2$ [sec], the maximum length of the path is 50, each user's distance from the center of the AoI is ranging $[10, 77]$ [m], $B_{n_a} \in [3 \times 10^5, 2 \times 10^6]$ [bits], $B_{UAV} = 1 \times 10^9$ [bits], $\xi = 0.1[\frac{J \cdot s}{\frac{CPU-cycles}{s}}]$, $I_0 = 1 \times 10^{-9}$, $p_{n_a} = 1$ [W], $e_{n_a} = 29$ [KJoules], b = 0.74 and c = 0.00043, unless otherwise explicitly stated. Specifically, the hyperparameters of the proposed BRAVE-MARL algorithm are learning rate $\alpha = 0.1$, discount factor $\gamma = 0.9$, the exploration rate ϵ is initialized at 0.95 and decayed over time using an exponential decay factor of 0.995 to encourage exploration during early episodes and exploitation in later stages.

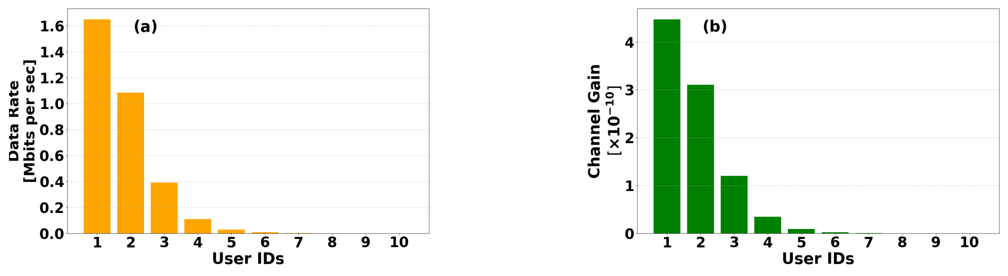


Figure 4: Users' data rate and channel gain.

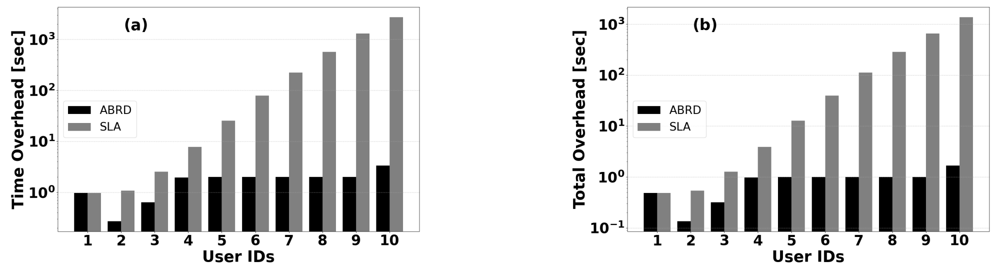


Figure 5: Users' time overhead and total overhead (logarithmic axes).

8.1. Optimal Data Offloading

In this section, the users' optimal data offloading is presented, as discussed in Section 3, following the game-theoretic approach in order to converge to a Pure Nash Equilibrium. Specifically, Fig. 3a – 3b demonstrate the users' total amount of data that they aim to offload to the UAVs in a representative AoI, as well as their distance from the center of the AoI, as a function of the user's ID, respectively. Fig. 4a – 4b illustrate the users' achieved data rate (Eq. 4) and their corresponding channel gain in the communication link with the AoI, as a function of the users' ID, respectively.

Moreover, the proposed game-theoretic approach, i.e., Asynchronous BRD (ABRD) Algorithm, has been compared with a reinforcement learning approach, i.e., Stochastic Learning Automata (SLA), under identical conditions, including the volume of transmitted data, power consumption, and solution cost, to demonstrate the benefits of the game-theoretic-based data offloading. Under the SLA approach, each user n_a has its strategy set $\mathcal{S}_{n_a} = \{s_{n_a,min}, \dots, s_{n_a,j}, \dots, s_{n_a,max}\}$ where $s_{n_a,j} \in [0,1]$, and selects the strategy $s_{n_a,j}$ at iteration t with probability $p_{n_a,j}(t)$. The probability of selecting the same action $s_{n_a,j}$ at time $t+1$ is updated as follows: $p_{n_a,j}(t+1) = p_{n_a,j}(t) + \alpha \cdot r(t) \cdot (1 - p_{n_a,j}(t))$, where $\alpha \in (0,1)$ is the learning

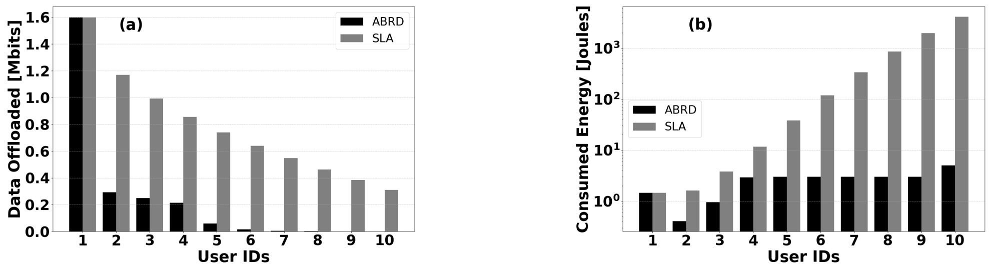


Figure 6: Users' amount of offloaded data and energy consumption.

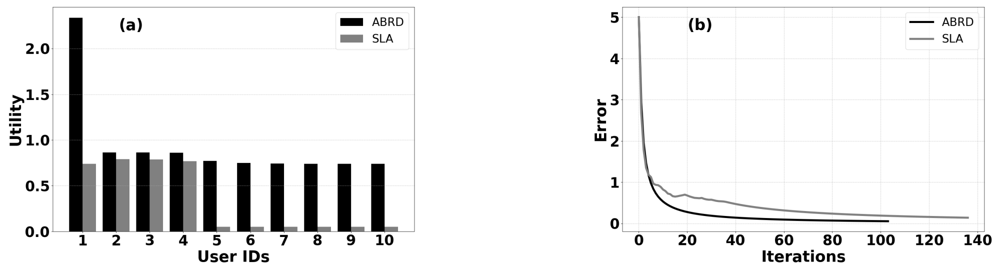


Figure 7: Users' utility at the PNE and convergence of the ABRD Algorithm.

rate, and $r(t) \in [0,1]$ is the reinforcement signal at time t. On the other hand, the probability of selecting a different action $s_{n_a,k}$ (where $k \neq j$) is updated as: $p_{n_a,k}(t+1) = p_{n_a,k}(t) - \alpha \cdot r(t) \cdot p_{n_a,k}(t)$. These updates ensure that the probability vector $\mathbf{p}_{n_a}(t)$ remains normalized, i.e., $\sum_j p_{n_a,j}(t) = 1$ for all t. The reinforcement signal $r(t)$ is defined as the normalized value of the utility function $U_{n_a}(s_{n_a,j}, \mathbf{s}_{-n_a,j})$, given by: $r(t) = \frac{U_{n_a}(s_{n_a,j},\mathbf{s}_{-n_a,j})}{\sum\limits_{s_{n_a,k} \in \mathcal{S}_{n_a}} U_{n_a}(s_{n_a,k},\mathbf{s}_{-n_a,k})}$, $\alpha = 0.7$, and the SLA algorithm converges if the probability of an action of a user exceeds 0.85. It is noted that each user acts as an RL agent, executing the SLA algorithm autonomously at the beginning of each time slot, and the algorithm converges if the same strategies are selected by all the users for a consecutive period of time, i.e., 10 time slots. The following comparative results among the game-theoretic ABRD and the reinforcement learning-based SLA algorithms are presented. Fig. 5a – 5b show the users' experienced time overhead (Eq. 5) and the users' total overhead considering both the latency and energy overhead, as expressed in Eq. 7, as a function of the users' ID, respectively. Fig. 6a – 6b depict the users' optimal amount of offloaded data at the Pure Nash Equilibrium point (Eq. 11) and their corresponding consumed energy as expressed in Eq. 6, with respect to the users' ID, respectively. Also, Fig. 7a – 7b present the users' utility as a function of the user's

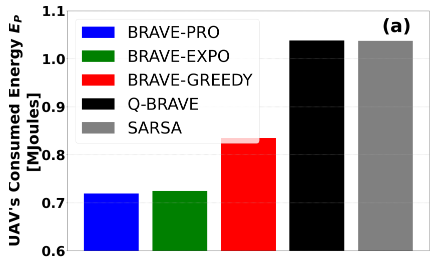
ID (Eq. 8) and the error of ABRD and SLA algorithms as a function of the iterations until they converge to the Pure Nash Equilibrium, respectively.

In the indicative AoI under consideration, we have sorted the user's ID based on the users' total amount of data available to offload to the UAV, as presented in Fig. 3a. The results reveal that the users who reside closer to the UAV (Fig. 3b) and are characterized by a larger amount of data to be offloaded to it (Fig. 3a) achieve better channel gain conditions (Fig. 4b) resulting in an improved data rate (Fig. 4a), thus, experiencing a lower time overhead (Fig. 5a), as well as overall overhead (Fig. 5b) considering both the latency and energy overhead. Also, the results show that the users characterized by favorable channel gain conditions achieve to offload the larger amount of data (Fig. 6a) with a lower energy cost (Fig. 6b) resulting in a higher achieved utility (Fig. 7a). Also, the results demonstrate the efficient convergence of the Asynchronous Best Response Dynamics Algorithm, which converges to the Pure Nash Equilibrium in less than 100 iterations, which is translated to 0.9 seconds. This analysis confirms that the distributed decision-making among the users in terms of deciding their optimal data offloading strategies operates in an optimal manner in terms of respecting the users' communication characteristics, as well as data offloading needs, by converging to the Pure Nash Equilibrium, as presented in Section 3.

Moreover, focusing on the comparative evaluation of the game-theoretic ABRD and the reinforcement learning-based SLA algorithms, the results reveal that ABRD minimizes the overall overhead (Fig. 5b), thus, it makes the data offloading process more scalable and efficient for large-scale systems with multiple users compared to the SLA algorithm. This outcome is achieved given that ABRD determines the PNE and optimizes the resource allocation and task offloading more effectively, thus, it reduces the energy consumption (Fig. 6b), while it optimizes the data offloading considering the users' data availability and priority to offload data to the UAV in order to optimize the UAV's data collection process (Fig. 6a). Based on these observations, the users achieve a higher utility through the data offloading process (Fig. 7a). Also, the results reveal that the SLA algorithm suffers from long convergence time compared to the ABRD which converges fast to the PNE (Fig. 7b).

8.2. BRAVE Frameworks' Operation and Performance

In this section, the pure operation and performance of the BRAVE framework is introduced by providing a comparative evaluation among the differ-



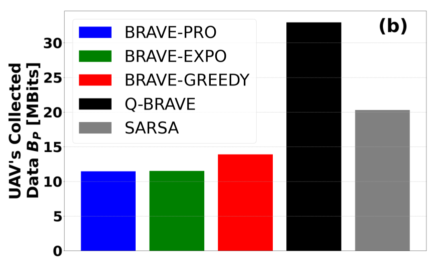
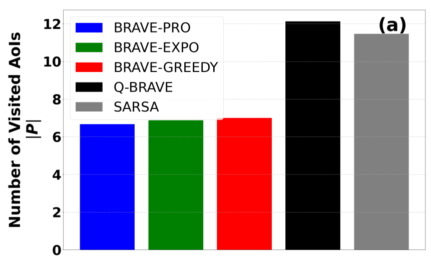


Figure 8: UAV's consumed energy and amount of collected data, for the four BRAVE models.



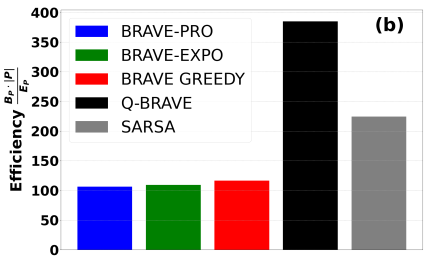


Figure 9: Number of visited AoIs and Efficiency $\frac{B_P \cdot |P|}{E_P}$ of BRAVE models.

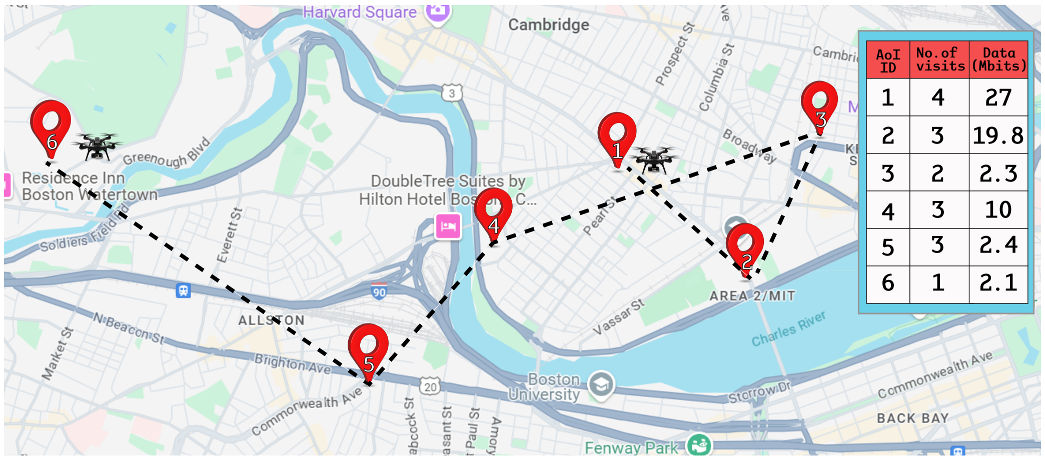
ent BRAVE models, i.e., BRAVE-GREEDY, BRAVE-PRO, BRAVE-EXPO, and Q-BRAVE, and other single-agent reinforcement learning approaches, i.e., SARSA algorithm [29, 30], that have been introduced in the literature to perform the UAV's path planning. Fig. 8a–9a and Fig. 9b present the UAV's consumed energy, amount of collected data, number of visited AoIs, and the efficiency parameter, defined as $\frac{B_P \cdot |P|}{E_P}$ for the four BRAVE models and the SARSA algorithm, respectively. It is noted that a Monte Carlo analysis is performed with 500 executions of each experiment, and the SARSA algorithm adopts the same hyperparameters as the four BRAVE algorithms for fairness in the comparison.

The results show that the Q-BRAVE model outperforms the rest of the models by collecting the largest amount of data, visiting the largest number of AoIs, while consuming a large amount of energy. The overall benefit of the Q-BRAVE model is reflected in the efficiency parameter, showing that the Q-BRAVE model achieves approximately 260%, 250%, and 230% improved efficiency compared to the BRAVE-PRO, BRAVE-EXPO, and BRAVE-GREEDY models, respectively. The superior performance of the Q-BRAVE model stems from the intelligence of the Q-learning algorithm that enables the UAV to avoid the destination AoI a_{final} before exhausting its available energy in order to continue supporting the data collection and processing in the field. In contrast, the rest of the models, either select greedily (BRAVE-GREEDY) or probabilistically an AoI (BRAVE-PRO and BRAVE-EXPO) based on the AoI's available data to be collected and processed, thus, the destination AoI has a higher probability to be visited, even before the UAV's energy is not exhausted. Therefore, we observe that even though under the Q- BRAVE model the UAV's consumed energy is higher compared to the rest of the models, and the benefit provided in the search and rescue mission is quite higher, as reflected by the efficiency metric (Fig. 9b).

Focusing on the comparison of Q-BRAVE compared to SARSA, the results reveal that the Q-BRAVE algorithm outperforms SARSA by 62.36%, 5.83%, and 71.69% regarding the amount of collected data, number of visited nodes, and efficiency, respectively, while consuming very similar energy to perform the data collection process. It is noted that even both algorithms are characterized as single-agent reinforcement learning algorithms aiming at determining the UAV's optimal path plan, SARSA is an on-policy RL algorithm versus Q-BRAVE, which is an off-policy RL algorithm. This fundamental difference, SARSA tends to be more risk-averse because it updates its value function based on the actual actions taken, including the exploratory actions that can lead to penalties, i.e., high energy consumption or low data collection. Thus, SARSA converges to inefficient paths in terms of both visiting a smaller number of AoIs (Fig. 9a), collecting a smaller amount of data (Fig. 8b), and resulting in a substantially lower efficiency (Fig. 9b) compared to the Q-BRAVE, even though it enforces the UAV to consume a similar amount of energy compared to the Q-BRAVE algorithm (Fig. 8a).

8.3. Realistic Public Safety Scenario enabled by the BRAVE Framework

In this section, a realistic scenario of applying the BRAVE framework is presented. Specifically, the Boston Marathon Bombing public safety event is considered and the Q-BRAVE model is applied, given its superiority compared to the other models in terms of efficiency, as presented in the previous section. Focusing on the Boston Marathon bombing on April 15, 2013, six AoIs were involved: 1) Central Square, 2) Massachusetts Institute of Technology (MIT) campus, 3) 3rd Street, Cambridge, 4) Memorial Drive, 5) The



AoI ID	No. of visits	Data (Mbits)
1	4	27
2	3	19.8
3	2	2.3
4	3	10
5	3	2.4
6	1	2.1

Figure 10: Q-BRAVE model applied in the Boston Marathon Bombing.

intersection of Brighton Avenue and Commonwealth Avenue, and 6) Watertown. The suspects' residence was located in AoI 3, however, the suspects were on the move in a hijacked SUV, making AoIs 1 and 2 more critical. The suspects were seen in AoI 1 around 10 PM on April 18, prompting an increased law enforcement presence in the area. Just 30 minutes later, a police officer was shot at AoI 2, escalating its importance. Another 30 minutes after that, the owner of the hijacked car was released in AoI 4, further raising the urgency of this area. As a result of these events, AoIs 1, 2, and 4 were considered more critical, and thus, a higher volume of data collection from these locations was needed, as presented in Fig. 10. Although AoI 5 was along the suspects' escape route toward Watertown, less data was gathered from this area. The final suspect was captured in AoI 6, which served as the UAV's final destination in this case study. The embedded table in Fig. 10 demonstrates the UAV's path planning derived from the Q-BRAVE model, which fully aligns with the need to collect and process data, to support the Boston Marathon Bombing public safety event and the rescue mission of the Emergency Control Center.

8.4. BRAVE-MARL and Comparative Evaluation

In this section, our analysis is focused on the multi UAV scenario where the BRAVE-MARL algorithm is applied. Specifically, we consider 3 UAVs and the total number of 10 AoIs that have available data to upload to the UAVs. The BRAVE-MARL algorithm is compared to the following scenarios: (i) Independent Learning (IL): each UAV independently learns its

Q-table in order to decide its optimal path, (ii-iii) the previous scenario with double (Double Extended Training – DET) and triple (Triple Extended Training – TET) sets of episodes for training the learning algorithm; (iv) BRAVE-GREEDY, and (v) Dijkstra, where each UAV determines sequentially its optimal path to collect data from the AoIs. To the best of our knowledge, there are no existing approaches in the current state-of-the-art dealing with the RL-based multi-agent path planning in a swarm of UAVs, where the UAVs cooperate among each other, and this is the fundamental contribution of our research work. Focusing on the comparison of the BRAVE framework to existing RL-based path planning approaches, the IL, DET, and TET algorithms are considered for the comparison. On the other hand, focusing on comparing the BRAVE framework to non-RL multi-agent path planning approaches, the Dijkstra's algorithm is adopted, where the cost of each link among two AoIs is the equal-weight linear combination of the distance cost and the normalized inverse of the amount of collected data from the destination-AoI. Also, the comparison of the BRAVE framework to non-RL single-agent approaches is demonstrated through the comparison to the BRAVE-GREEDY algorithm. Fig. 11a – 11e present the UAVs' total consumed energy, amount of collected data, number of visited AoIs, efficiency, and execution time of the algorithms for all the comparative scenarios, respectively. It is noted that a Monte Carlo analysis is followed for 500 executions of the experiments.

The results demonstrate that the BRAVE-MARL model outperforms the comparative scenarios by collecting a greater volume of data compared to the IL and DET scenarios (Fig. 11b) and visiting more AoIs compared to all the comparative scenarios (Fig. 11c), albeit with higher energy consumption (Fig. 11a). However, it delivers significantly enhanced efficiency (Fig. 11d) while drastically reducing the algorithm's execution time (Fig. 11e). In contrast, the Independent Learning approach delivers the poorest performance in terms of data collection efficiency. Even with extended training (double and triple episodes), the improvements in efficiency come at the prohibitive cost of significantly increased execution time, especially as the number of episodes increases. Focusing on the BRAVE-GREEDY algorithm, while simpler with respect to its design, it achieves low efficiency given that it falls short compared to BRAVE-MARL in terms of the number of visited AoIs and the overall data collection volume due to the fact that the UAVs do not cooperate among each other, leading to overlaps and/or inefficiencies in the path planning process. For similar reasons, the Dijkstra's algorithm results in very low amount of collected data, by consuming also a small amount of energy due to the fact that it visits very few AoIs, resulting in a very low efficiency compared to the proposed BRAVE-MARL framework. These results quantify the superiority of the BRAVE-MARL algorithm, which leverages the UAVs' cooperation for optimal path planning, achieving the highest efficiency in data collection during public safety events.

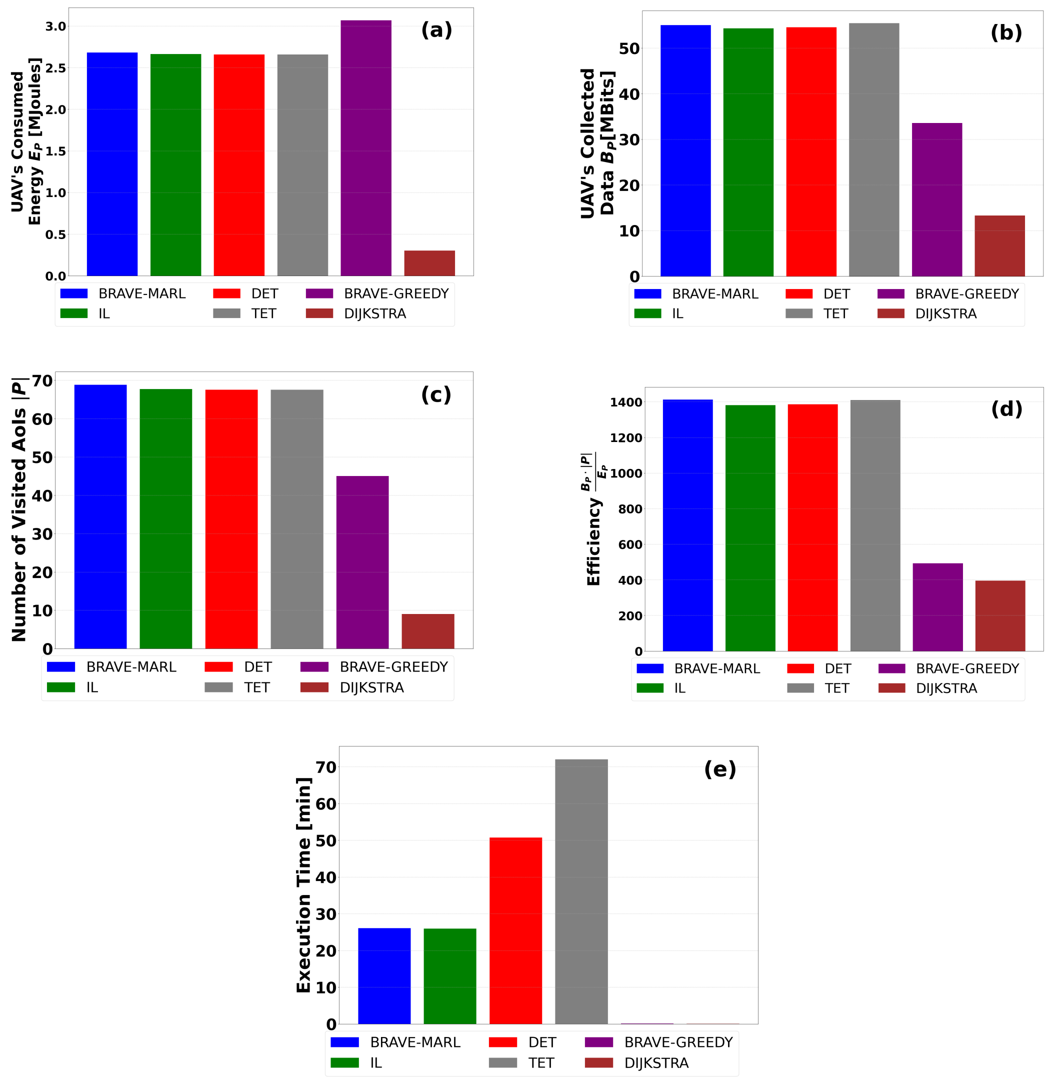


Figure 11: UAVs' total consumed energy, amount of collected data, number of visited AoIs, efficiency, and execution time of the algorithms for all the comparative scenarios.

The BRAVE framework jointly addresses the problem of optimal users' data offloading and UAV path planning in disaster-stricken areas. The data offloading process focuses on optimizing users' utility by balancing satisfaction and costs related to latency and energy consumption. On the other hand, the UAVs' path planning prioritizes the spatial and temporal efficiency in order to ensure that the UAVs visit the AoIs effectively while considering their resource constraints and the dynamic environmental factors. By treating these processes separately, we can design specialized algorithms that focus on the unique requirements of each level without imposing undue complexity on a single monolithic framework. Also, this approach makes the overall design more modular and the overall system becomes more scalable and adaptable to varying operational scenarios. Considering the outcome of the users' optimal data offloading, the UAVs decide the future AoIs to be visited by accounting for the data availability in each AoI. The current design employs advanced game-theoretic principles, including the use of submodular games for data offloading, to ensure convergence to a stable Pure Nash Equilibrium. Focusing on the UAVs' path planning, the reinforcement learning techniques explore a broad solution space during the exploration phase to mitigate the risk of local optima.

9. Conclusions

In conclusion, this paper introduces the BRAVE framework as a comprehensive solution for leveraging UAVs as mobile MEC servers in public safety environments and addressing critical challenges in energy consumption, path planning, and data offloading. By accurately modeling the UAVs' energy consumption and incorporating the users' QoS requirements, the BRAVE framework ensures the efficient support for disaster-stricken areas. The BRAVE framework's two-level decision-making mechanism integrates a submodular game for the optimal users' data offloading and a reinforcement learning-based approach for the UAVs' path optimization. Extending to a collaborative multi-UAV setting, the BRAVE-MARL mechanism demonstrates the benefits of cooperative reinforcement learning in terms of maximizing the data collection and processing support while meeting the UAVs' energy constraints. Extensive simulation-based evaluations validate the framework's adaptability and effectiveness and offer practical insights for the Emergency Control Centers to tailor the solution to diverse public safety scenarios.

References

[1] A. Beishenalieva, S.-J. Yoo, Uav path planning for data gathering in wireless sensor networks: Spatial and temporal substate-based q-learning, IEEE Internet of Things Journal 11 (6) (2024) 9572–9586. doi:10.1109/JIOT.2023.3323921.

[2] Y. Wang, W. Chen, T. H. Luan, Z. Su, Q. Xu, R. Li, N. Chen, Task offloading for post-disaster rescue in unmanned aerial vehicles networks, IEEE/ACM Transactions on Networking 30 (4) (2022) 1525–1539. doi:10.1109/TNET.2022.3140796.

[3] D. Yang, J. Wang, F. Wu, L. Xiao, Y. Xu, T. Zhang, Energy efficient transmission strategy for mobile edge computing network in uav-based patrol inspection system, IEEE Transactions on Mobile Computing 23 (5) (2024) 5984–5998. doi:10.1109/TMC.2023.3315477.

[4] N. Zhao, Z. Ye, Y. Pei, Y.-C. Liang, D. Niyato, Multi-agent deep reinforcement learning for task offloading in uav-assisted mobile edge computing, IEEE Transactions on Wireless Communications 21 (9) (2022) 6949–6960. doi:10.1109/TWC.2022.3153316.

[5] A. M. Said, M. Marot, C. Boucetta, H. Afifi, H. Moungla, G. Roujanski, Reinforcement learning vs rule-based dynamic movement strategies in uav assisted networks, Vehicular Communications 48 (2024) 100788.

[6] A. M. Rahmani, A. Haider, S. Alsubai, A. Alqahtani, A. Alanazi, M. Hosseinzadeh, A novel energy-efficient and cost-effective task offloading approach for uav-enabled mec with leo enhancement in internet of remote things networks, Simulation Modelling Practice and Theory 137 (2024) 103018.

[7] H. A. Alharbi, B. A. Yosuf, M. Aldossary, J. Almutairi, J. M. H. Elmirghani, Energy efficient uav-based service offloading over cloud-fog architectures, IEEE Access 10 (2022) 89598–89613. doi:10.1109/ACCESS.2022.3201112.

[8] Q. Tang, L. Liu, C. Jin, J. Wang, Z. Liao, Y. Luo, An uav-assisted mobile edge computing offloading strategy for minimizing energy consumption, Computer Networks 207 (2022) 108857.

[9] B. Ma, H. Kuang, S. Liu, C. Li, Uav assisted cellular network traffic offloading: Joint swarm, 3d deployment, and user allocation optimization based on a data-aware method, Computer Networks 231 (2023) 109812.

[10] H. Huang, Z.-Y. Chai, B.-S. Sun, H.-S. Kang, Y.-J. Zhao, Multiobjective deep reinforcement learning for computation offloading and trajectory control in uav-base-station-assisted mec, IEEE Internet of Things Journal 11 (19) (2024) 31805–31821. doi:10.1109/JIOT.2024.3420884.

[11] Y. Spyridis, T. Lagkas, P. Sarigiannidis, J. Zhang, Modelling and simulation of a new cooperative algorithm for uav swarm coordination in mobile rf target tracking, Simulation Modelling Practice and Theory 107 (2021) 102232.

[12] X. Chen, Y. Bi, G. Han, D. Zhang, M. Liu, H. Shi, H. Zhao, F. Li, Distributed computation offloading and trajectory optimization in multi-uav-enabled edge computing, IEEE Internet of Things Journal 9 (20) (2022) 20096–20110. doi:10.1109/JIOT.2022.3175050.

[13] X. Zhu, L. Wang, Y. Li, S. Song, S. Ma, F. Yang, L. Zhai, Path planning of multi-uavs based on deep q-network for energy-efficient data collection in uavs-assisted iot, Vehicular Communications 36 (2022) 100491.

[14] W. Lee, T. Kim, Multiagent reinforcement learning in controlling offloading ratio and trajectory for multi-uav mobile-edge computing, IEEE Internet of Things Journal 11 (2) (2024) 3417–3429. doi:10.1109/JIOT.2023.3296774.

[15] B. Li, W. Liu, W. Xie, X. Li, Energy-efficient task offloading and trajectory planning in uav-enabled mobile edge computing networks, Computer Networks 234 (2023) 109940.

[16] S. Li, F. Wu, S. Luo, Z. Fan, J. Chen, S. Fu, Dynamic online trajectory planning for a uav-enabled data collection system, IEEE Transactions on Vehicular Technology 71 (12) (2022) 13332–13343. doi:10.1109/TVT.2022.3200458.

[17] C. Peng, X. Huang, Y. Wu, J. Kang, Constrained multi-objective optimization for uav-enabled mobile edge computing: Offloading optimization and path planning, IEEE Wireless Communications Letters 11 (4) (2022) 861–865. doi:10.1109/LWC.2022.3149007.

[18] S. Akter, D. Van Anh Duong, D.-Y. Kim, S. Yoon, Task offloading and resource allocation in uav-aided emergency response operations via soft actor critic, IEEE Access 12 (2024) 69258–69275. doi:10.1109/ACCESS.2024.3401115.

[19] Z. Shah, U. Javed, M. Naeem, S. Zeadally, W. Ejaz, Mobile edge computing (mec)-enabled uav placement and computation efficiency maximization in disaster scenario, IEEE Transactions on Vehicular Technology 72 (10) (2023) 13406–13416. doi:10.1109/TVT.2023.3274107.

[20] S. Ghosh, P. Kuila, Efficient offloading in disaster-affected areas using unmanned aerial vehicle-assisted mobile edge computing: A gravitational search algorithm-based approach, International Journal of Disaster Risk Reduction 97 (2023) 104067.

[21] T. Wang, X. Fu, A. Guerrieri, Joint resource scheduling and flight path planning of uav-assisted iots in response to emergencies, Computer Networks 253 (2024) 110731.

[22] X. Liu, Z.-Y. Chai, Y.-L. Li, Y.-Y. Cheng, Y. Zeng, Multi-objective deep reinforcement learning for computation offloading in uav-assisted multi-access edge computing, Information Sciences 642 (2023) 119154.

[23] T. Ju, L. Li, S. Liu, Y. Zhang, A multi-uav assisted task offloading and path optimization for mobile edge computing via muti-agent deep reinforcement learning, Journal of Network and Computer Applications (2024) 103919.

[24] H. V. Abeywickrama, B. A. Jayawickrama, Y. He, E. Dutkiewicz, Comprehensive energy consumption model for unmanned aerial vehicles, based on empirical studies of battery performance, IEEE Access 6 (2018) 58383–58394. doi:10.1109/ACCESS.2018.2875040.

[25] Y. Wang, M. Sheng, X. Wang, L. Wang, J. Li, Mobile-edge computing: Partial computation offloading using dynamic voltage scaling, IEEE Transactions on Communications 64 (10) (2016) 4268–4282. doi:10.1109/TCOMM.2016.2599530.

[26] T. Wang, C. You, Distributed user association and computation offloading in uav-assisted mobile edge computing systems, IEEE Access 12 (2024) 63548–63567. doi:10.1109/ACCESS.2024.3396471.

[27] X. Vives, Complementarities and games: New developments, Journal of Economic Literature 43 (2) (2005) 437–479.

[28] Z. Lyu, M. Z. Islam, A. J. Yu, A scalable and adaptable supervised learning approach for solving the traveling salesman problems, IEEE Transactions on Intelligent Transportation Systems 25 (11) (2024) 17092–17104. doi:10.1109/TITS.2024.3410691.

[29] P. A. Apostolopoulos, M. Torres, E. E. Tsiropoulou, Satisfaction-aware data offloading in surveillance systems, in: Proceedings of the 14th workshop on challenged networks, 2019, pp. 21–26.

[30] R. S. Sutton, Reinforcement learning: An introduction, A Bradford Book (2018).