Data Aggregation and Dissemination in Emerging Communication Networks

by

© Ahmed A. Al-habob

A dissertation submitted to the School of Graduate Studies

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Faculty of Engineering and Applied Science

Memorial University of Newfoundland

May 2022

St. John's, Newfoundland

Abstract

The emerging communication networks generate a huge amount of data that need to be aggregated/disseminated, processed, and responded to in a very short time. Major challenges associate with the need to handle such tremendous amount of data, including high energy consumption, larger delay, and constrained computation capabilities. Consequently, more efficient frameworks could be exploited for data aggregation, dissemination, and processing. This work aims to design energy-efficient and age-optimum frameworks for data aggregation/dissemination. Moreover, reliable and low latency offloading frameworks for sequential and parallel mobile edge computing (MEC) offloading are also developed. A device-role assignment framework is designed to optimize the role of each device in the network and enable in-network data processing. More sophisticated scenarios with mobile data aggregator/disseminator are explored as well. Mobile data aggregator(s)/disseminator(s) for terrestrial and underwater scenarios are considered. Different metrics are studied including the overall energy consumption in the data aggregation/ dissemination systems, age-of-information (AoI) in the data aggregation systems, and the latency and offloading error in the MEC systems. A novel metric referred to as the correlation-aware AoI is also proposed to captures both the freshness and diversity in the aggregated data. Computationally efficient solution approaches are developed to find solutions for the proposed frameworks, including genetic algorithms, ant colony optimization, conflict graphs, and deep reinforcement learning agents. To show the effectiveness of the developed solution approaches, their performance is compared to baseline approaches. Extensive simulations show that the proposed solution approaches provide performance close to the optimal solutions, which are obtained through exhaustive search or computationally intensive methods.

Acknowledgments

I would like to express my deepest appreciation to my supervisor, Prof. Octavia A. Dobre for her encouragement, trust, and valuable guidance. I am really grateful to her, not only for supervising me and contributing valuable suggestions to this work, but most importantly for training me to be an independent researcher, with the ability to identify research problems and to develop their solution frameworks. I feel honored to work under the supervision of Prof. Dobre; the PhD journey with her is one of the most amazing things in my life. I would like to acknowledge the financial support provided by my supervisor, Memorial University's Research Chair, the Faculty of Engineering and Applied Science, the School of Graduate Studies, and the Natural Science and Engineering Research Council of Canada.

My appreciation also goes to my colleagues in our research group, for the friendly academic atmosphere, the useful discussions and their encouragement. I am not going to mention names not to forget anyone, thank you All!

The deepest word of acknowledgment is reserved to my parents for their unconditional support. I would like to pay my humble respect and the deepest thanks from my heart to them. I thank them infinitely for their continuous encouragement, prayers, support, and extra care.

The final warmest thanks from my heart go to my lovely wife Nahed, the twin of my soul and the love of my life. Finally, special thanks to my sons Yahya and Lutf, I love you! Your smile always makes me stronger.

Co-Authorship Statement

I, Ahmed A. Al-habob, hold a principle author status for all the manuscript chapters (Chapter 3 - 6) in this thesis. However, each manuscript is co-authored by my supervisor and co-researchers, whose contributions have facilitated the development of this work as described below.

- Paper 1 in Chapter 3 - Section 3.2: Ahmed A. Al-habob, Octavia A. Dobre, and H. Vincent Poor, "Role Assignment for Spatially-Correlated Data Aggregation Using Multi-Sink Internet of Underwater Things,"*IEEE Trans. Green Commun. Netw.*, vol. 5, no. 3, pp. 1570-1579, Apr. 2021.

  I was the primary author, with authors 2 and 3 contributing to the idea, its formulation and development, and refinement of the presentation.

- Paper 2 in Chapter 3 - Section 3.2: Ahmed A. Al-habob and Octavia A. Dobre, "Role Assignment for Energy-Efficient Data Gathering Using Internet of Underwater Things,"in *Proc. IEEE Int. Conf. Commun.*, Dublin, Ireland, Jun. 2020, pp. 1–6.

  I was the primary author, with author 2 contributing to the idea, its formulation and development, and refinement of the presentation.

- Paper 3 in Chapter 3 - Section 3.3: Ahmed A. Al-habob, Octavia A. Dobre, and and H. Vincent Poor, "Energy-Efficient Spatially-Correlated Data Aggregation Using Unmanned Aerial Vehicles," in *Proc. IEEE Int. Symp. on Personal, Indoor and Mobile Radio Commun.*, London, UK, 2020, pp. 1–6.

  I was the primary author, with authors 2 and 3 contributing to the idea, its formulation and development, and refinement of the presentation.

- Paper 4 in Chapter 4 - Section 4.2: Ahmed A. Al-habob, Octavia A. Dobre, and H. Vincent Poor, "Age-Optimal Information Gathering in Linear Underwater Networks: A Deep Reinforcement Learning Approach,"*IEEE Trans. Veh. Technol.*, Early Access, Oct. 2021.

  I was the primary author, with authors 2 and 3 contributing to the idea, its formulation and development, and refinement of the presentation.

- Paper 5 in Chapter 4 - Section 4.3: Ahmed A. Al-habob, Octavia A. Dobre, and H. Vincent Poor, "Age- and Correlation-Aware Information Gathering,"*IEEE Wireless Commun. Lett.*, Early Access, Nov. 2021.

  I was the primary author, with authors 2 and 3 contributing to the idea, its formulation and development, and refinement of the presentation.

- Paper 6 in Chapter 5 - Section 5.2.1: Ahmed A. Al-habob, Octavia A. Dobre, Sami Muhaidat, and H. Vincent Poor, "Energy-Efficient Data Dissemination Using a UAV: An Ant Colony Approach,"*IEEE Wireless Commun. Lett.*, vol. 10, no. 1, pp. 16-20, Jan. 2021.

  I was the primary author, with authors 2 - 4 contributing to the idea, its formulation and development, and refinement of the presentation.

- Paper 7 in Chapter 5 - Section 5.3: Ahmed A. Al-habob, Octavia A. Dobre, Sami Muhaidat, and H. Vincent Poor, "Energy-Efficient Information Placement and Delivery Using UAVs,"*Submitted to IEEE Internet Things J.* Dec. 2021.

  I was the primary author, with authors 2 - 4 contributing to the idea, its formulation and development, and refinement of the presentation.

- Paper 8 in Chapter 6 - Section 6.2: Ahmed A. Al-habob, Ahmed Ibrahim, Octavia A. Dobre, and Ana García Armada, "Collision-free Sequential Task Offloading for Mobile Edge Computing,"*IEEE Commu. Lett.*vol. 24, no. 1, pp. 71-75, Jan. 2020.

  I was the primary author, with authors 2 - 4 contributing to the idea, its formulation and development, and refinement of the presentation.

- Paper 9 in Chapter 6 - Section 6.3: Ahmed A. Al-habob, Octavia A. Dobre, Ana García Armada, and Sami Muhaidat "Task Scheduling for Mobile Edge Computing Using Genetic Algorithm and Conflict Graphs,"*IEEE Trans. Veh. Technol.*, vol. 69, no. 8, pp. 8805-8819, Aug. 2020.

  I was the primary author, with authors 2 - 4 contributing to the idea, its formulation and development, and refinement of the presentation.

- Paper 10 in Chapter 6 - Section 6.3: Ahmed A. Al-habob, Octavia A. Dobre, and Ana García Armada, "Sequential Task Scheduling for Mobile Edge Computing Using Genetic Algorithm,"in *Proc. IEEE Global Commun. Conf.*, Waikoloa, HI, USA, Dec. 2019.

  I was the primary author, with authors 2 and 3 contributing to the idea, its formulation and development, and refinement of the presentation.

Table of Contents

List of Tables

List of Figures

List of Abbreviations and Acronyms

5G	Fifth-generation
6G	Sixth-generation
ACO	Ant colony optimization
AoI	Age-of-information
AUV	Autonomous underwater vehicle
CAAoI	Correlation-aware age-of-information
CPU	Central processing unit
DDPG	Deep deterministic policy gradient
DNN	Deep neural network
DRL	Deep reinforcement learning
GA	Genetic algorithm
IoT	Internet-of-things
IoUT	Internet-of-underwater-things
IRS	Intelligent reflecting surface
LoS	Line-of-sight
MCC	Mobile cloud computing
MCGA	Multi-chromosome genetic algorithm
MCS	Modulation and coding scheme
MEC	Mobile edge computing
ML	Machine learning

mmWave	Millimeter wave
NLoS	Non line-of-sight
NN	Nearest neighbor
PER	Packet error rate
RL	Reinforcement learning
SNR	Signal-to-noise-ratio
TSP	Traveling salesman problem
UAV	Unmanned aerial vehicle
UE	User equipment
WSN	Wireless sensor network

Chapter 1

Introduction

1.1 Background and Motivation

The emerging fifth-generation (5G) and the envisioned sixth-generation (6G) communication networks require full-fledged frameworks for connected things with stringent and diverse requirements in terms of energy efficiency, information freshness, latency, and reliability. Smart objects, like mobile phones, vehicles, wearable devices, and sensors, are expected to be connected and share information to each other in the future communication networks [1]. This associates with a wide range of applications that require a timely information [2] and computation-intensive tasks [3]. This paradigm gave birth to a new communication ecosystem, namely Internet-of-Things (IoT) which utilizes the ubiquity of sensor-equipped devices to aggregation/dissemination information at low cost. Due to the massive number of IoT devices and large volume of aggregated/disseminated information, the traditional techniques of wireless sensor networks such as relaying and routing the data across the nodes to the sink node are inefficient. The IoT devices are intelligent and capable

of sensing, interpreting and processing acquired data, and reacting to the environment due to the powerful tracking technologies and embedded sensors and modems in these devices. Unlike its terrestrial counterpart, the underwater communication environment is harsh; in this context, the Internet-of-Underwater-Things (IoUT) is considered, which refers to a set of interconnected underwater devices that provide low-cost and energy-efficient network deployment [4].

Efficient data aggregation and dissemination represent primary goals of the IoT/IoUT networks. Data aggregation refers to collecting data from various devices and integrates them to minimize the traffic load in the network, which depends on the correlation among the data. Data dissemination refers to distributing data to various devices, which depends on the diversity among the devices' requirements. Traditional metrics such as delay and throughput can not fully characterize the information freshness. A performance metric, namely age-of-information (AoI), was introduced to characterize the freshness of the information from the destination device perspective [5]. AoI is defined as the time elapsed since the most recently received updated information at the destination was generated at the source device [5]. Moreover, running computation-intensive applications consumes a significant portion of the devices battery power. Mobile-edge computing (MEC) is considered as a promising solution that avoids communication bottlenecks and provides cloud-like computing at the network edge [6, 7]. The practical scenarios of MEC systems comprise: (1) data aggregation, which includes task offloading to the MEC servers; (2) task computation at the MEC server; and (3) data dissemination, which includes distributing the computed results the the corresponding devices.

Motivated by what is mentioned above, this work aims to design frameworks for (1) Energy-efficient data aggregation; (2) Energy-efficient data dissemination; (3) Age-optimum

data gathering; and (4) Reliable and low latency MEC offloading. In these frameworks, different metrics are considered including the overall energy consumption in the data aggregation/dissemination systems, AoI in the data aggregation systems, and the latency and the offloading error in the MEC systems. Some of the proposed frameworks are applicable for aerial, terrestrial, and underwater communication environments. Consequently, in some scenarios I simulate aerial environments [8–10], terrestrial environments [11–13], and underwater communication environments [14–16]. Each framework yields a combinatorial or mixed integer optimization problem. Solving these optimization problems using a brute-force complete enumeration procedure or traditional exact solution methods can obtain solutions for relatively small search space; the complexity of such solution methods increases exponentially as the search space increases.

Computational intelligence, which refers to a set of nature-inspired solution approaches, such as genetic algorithms (GAs) [17] and ant colony optimization (ACO) [10], has been adopted as computationally efficient approaches for solving complex optimization problems. In GAs, a population of candidate solutions (referred to as individuals) undergoes a set of mutation operations and evolves towards the optimized solution. ACO is a probabilistic approach that converges towards the optimized solution based on two parameters, i.e., *attractiveness* which represents a priori information that captures the structure of a promising solution and *pheromone* which is a posteriori information that captures the structure of previously obtained good solutions; it is progressively updated by the ants to bias future ants toward high quality solutions. Recently, significant progress has been made to solve optimization problems by combining advances in deep learning with reinforcement learning (RL), resulting in deep reinforcement learning (DRL) approaches, in which deep neural network function approximators estimate the action-value function. However, while

DRL model solves problems with high-dimensional observation spaces, it cannot handle optimization problem with continuous decision variables. The actor-critic DRL can solve optimization problems with continuous action spaces [18].

1.2 Thesis Objective

In this thesis, I have identified and investigated the following research points:

1. Develop a device-role assignment framework, in which not only the optimum set of active devices is selected, but also the role of each active device is optimally assigned to maximize the total gathered information with minimal energy consumption [14, 15].

2. Develop a framework that optimizes both the active device selection and the itinerary of a mobile data gathering center for energy-efficient data aggregation [8].

3. Develop a framework to maintain the information freshness about a set of physical processes at a mobile data gathering center that gathers data from a set of linearly-deployed devices [16].

4. Introduce a novel metric referred to as correlation-aware AoI (CAAoI) that not only captures the information freshness at the receiver, but also reflects the diversity in the gathered information. With this new metric, an information gathering system is studied, in which a mobile data gathering center gathers information about a set of physical processes; each process can be measured by one or more devices [9].

5. Develop a framework to optimize the itinerary of a data gathering center for energy-

efficient data dissemination to spatially-distributed devices. A two-tiered data dissemination framework is considered, in which a subset of devices receive data from the data gathering center and forward them to other devices [10].

6. Develop an energy-efficient data placement and delivery framework, in which a fleet of mobile data gathering centers delivers a library of data files to a set of devices. The framework optimizes both the file placement and itinerary of the data gathering centers

7. Propose a collision-free sequential task offloading scheme to multiple MEC servers, in which a a delay-sensitive and computationally-intensive task is allocated to a set of MEC servers to minimize the latency and offloading failure probability [11].

8. Design a scheduling approach for inter-dependent sub-tasks offloading in MEC system, in which a delay-sensitive and computationally-intensive task is offloaded to multiple MEC servers. The task consists of a set of sub-tasks, and the general dependency among sub-tasks is considered in the system model. Optimization problems are formulated to jointly minimize the latency and offloading failure probability in both parallel and sequential offloading schemes [12, 13].

1.3 Thesis Outline

In the remainder of this dissertation, each research point mentioned above is discussed as follows. Chapter 2 introduces the concepts of spatial correlation and AoI, unmanned aerial vehicle (UAV) communication and power dissipation models, and the acoustic channel model. Chapter 3 introduces the proposed energy-efficient frameworks for data aggre-

gation. Chapter 4 discusses the proposed age-optimum data aggregation framework and introduces a novel metric to capture both freshness and diversity in the aggregated information. Chapter 5 introduces the proposed energy-efficient frameworks for data dissemination. Chapter 6 presents the proposed frameworks for task offloading in MEC systems. Finally, the thesis is concluded in Chapter 7.

Chapter 2

Preliminaries

This chapter introduces an iterative approach to quantify the joint entropy of a set of spatially distributed devices. This approach is utilized in Section 3.2, Section 3.3, and Section 4.3. The definition of the instantaneous AoI metric is provided along with the analysis of the time-average AoI. This analysis is utilized in Section 4.2 and Section 4.3. The power consumption and communication models of a UAV-enabled network are discussed. These models are employed in Section 3.3, Section 4.3, Section 5.2.1, and Section 5.3. Finally, the acoustic communication channel model is introduced, which is used in Section 3.2 and Section 4.2.

2.1 Spatial Correlation

Naturally, data measured by devices located within close proximity of an event is expected to be spatially correlated. The extent of such spatial correlation depends on the measured event and the distances among the devices. In order to characterize the spatial correlation, the joint entropy of the devices is considered to measure the total uncorrelated information.

Based on the empirical study in [19], the joint entropy of a set of N devices can be obtained using a constructive iterative approach, as follows. Assuming that each device generates $\mathcal{H}_1 = L$ (bits) of raw data, the entropy of the first device is L. The joint entropy of the first and second devices can be expressed as $\mathcal{H}_2 = L + \left[1 - \frac{1}{(\bar{d}_2/\rho+1)}\right] L$, where $\bar{d}_2$ is the distance between the first and second devices and ρ is a constant parameter that represents the spatial correlation extent in the data and its numerical value depends on the application. The joint entropy of the first three devices can be expressed as

$$\mathcal{H}_3 = L + \left[1 - \frac{1}{(\bar{d}_2/\rho + 1)}\right] L + \left[1 - \frac{1}{(\bar{d}_3/\rho + 1)}\right] L, \tag{2.1}$$

where $\bar{d}_2$ and $\bar{d}_3$ are obtained as shown in Fig. 2.1.

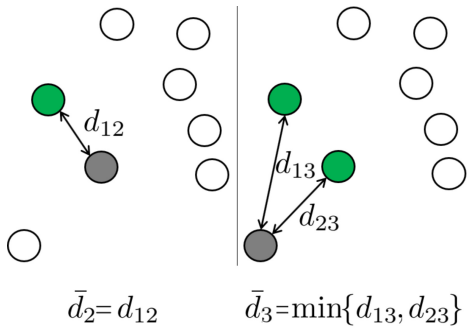

Figure 2.1: Illustration of obtaining $\bar{d}_2$ and $\bar{d}_3$.

Consequently, the amount of uncorrelated data that can be gathered by N devices can be expressed as

$$H = L + L \sum_{i=1}^{N} \left[1 - \frac{1}{\bar{d}_i/\rho + 1}\right], \tag{2.2}$$

where $\bar{d}_i$ is the minimum distance between the device n_i and all devices $n_k \; \forall k = 1, 2, \ldots, i-1$.

2.2 Age of Information

To assess the freshness of the received updates at the destination, Kaul *et al.* [5] define the instantaneous AoI of the physical process p_k at time instant t as

$$\Delta_k(t) = t - u^k(t), \tag{2.3}$$

where $\Delta_k(0) = 0$ and $u^k(t)$ is the time instant at which the last update about p_k was generated (also referred to as the "timestamp" of the last update). For a time interval T, the time-average AoI can be expressed as

$$\mathbb{E}_T\big[\Delta_k(t)\big] = \frac{1}{T}\int_0^T \Delta_k(t)\, dt. \tag{2.4}$$

The time average AoI $\mathbb{E}_T\big[\Delta_k(t)\big]$ equals the area under the curve of $\Delta_k(t)$ divided by T. In case no update of p_k is received, the area under the curve of $\Delta_k(t)$ will be a triangle (appears in blue color dashed-line in Fig. 2.2) and $\mathbb{E}_T\big[\Delta_k(t)\big]$ reaches its maximum value of $\frac{T}{2}$. Each update of p_k received at time instant $\tau_\iota^{(k)}$ reduces $\mathbb{E}_T\big[\Delta_k(t)\big]$ and this reduction equals the area of a parallelogram; the lengths of adjacent sides of the parallelogram are $\sqrt{2}(\tau_{\iota+1}^{(k)} - \tau_\iota^{(k)})$ and $(\tau_\iota^{(k)} - \ell_\iota^k)$ and its small angle is 45°. Consequently, the area the parallelogram resulted from the ι-th update is $\sqrt{2}(\tau_{\iota+1}^{(k)} - \tau_\iota^{(k)})(\tau_\iota^{(k)} - \ell_\iota^k)\sin(45^\circ) = (\tau_{\iota+1}^{(k)} - \tau_\iota^{(k)})(\tau_\iota^{(k)} - \ell_\iota^k)$. Consequently, the time-average AoI $\mathbb{E}\left[\Delta_k(t)\right]$ can be expressed as

$$\mathbb{E}_T\big[\Delta_k(t)\big] = \frac{T}{2} - \frac{1}{T}\sum_{\iota=1}^{U_k(T)} \left(\tau_{\iota+1}^{(k)} - \tau_\iota^{(k)}\right)\left(\tau_\iota^{(k)} - \ell_\iota^k\right), \tag{2.5}$$

where $U_k(T)$ is the total number of updates received during the time interval T.

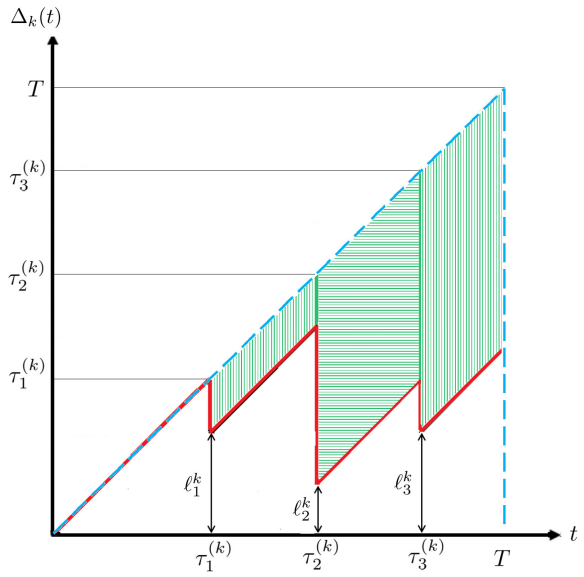

Figure 2.2: AoI of process p_k with $U_k(T) = 3$ updates.

2.3 UAV Power Dissipation Model

Power is consumed by the UAV to realize three primary functions: communication, hovering, and traveling. The energy consumed by the UAV at location $\dot{\psi}_j = \{\dot{x}_j, \dot{y}_j, \dot{z}_j\}$ to receive L bits from device n_i is $\frac{P_r L}{R_{ij}}$, where P_r denotes the power consumed by the receiver circuitry of the UAV and R_{ij} is the data rate between device n_i and the UAV at location $\dot{\psi}_j$, which can be obtained using (2.8). The hover power is the power consumed by the UAV while it hovers at a fixed position, which can be expressed as [20]

$$P_{\text{hov}} = \sqrt{\frac{(Mg)^3}{2\pi r^2 p \vartheta}}, \tag{2.6}$$

where M is the mass of the UAV, r is the propellers' radius, p is the number of propellers, g is the acceleration of gravity on earth, and ϑ is the density of air.

The traveling power is the power consumed by the UAV moving from a position to

another and can be written as

$$P_{\text{mov}} = \frac{v}{v_{\text{max}}} \left(P_{\text{max}} - P_{\text{stop}} \right) + P_{\text{stop}}, \tag{2.7}$$

where v and v_{max} are the traveling and maximum speed of the UAV, respectively [20, 21]. P_{max} is the UAV's consumed power when it moves at full speed, and P_{stop} is the UAV's consumed power when it is idle (i.e., $v = 0$).

2.4 UAV Communication Model

Assuming an additive white Gaussian noise channel, the average data rate between a device n_i and the UAV located at point $\dot{\psi}_j$ is

$$R_{ij} = B \log_2 \left(1 + \frac{P_T}{\varphi_{ij} N_0} \right), \tag{2.8}$$

where B is the bandwidth of the channel, P_T is the transmit power of the device, N_0 is the power of the noise, and φ_{ij} is the average channel path-loss. Following the works in [21], [22], and [23] a probabilistic ground-to-air communication model is considered, in which the average path-loss between a device n_i and the UAV located at point $\dot{\psi}_j$ is expressed as

$$\varphi_{ij} = \Pr_{ij}(\text{LoS})\, \varphi_{ij}(\text{LoS}) + \left[1 - \Pr_{ij}(\text{LoS})\right] \varphi_{ij}(\text{NLoS}), \tag{2.9}$$

where $\Pr_{ij}(\text{LoS})$ represents the probability of a line-of-sight (LoS) communication between n_i and the UAV located at aggregation point $\dot{\psi}_j$. This probability can be expressed as

$$\Pr_{ij}(\text{LoS}) = \frac{1}{1 + \alpha \exp(-\beta \left[\theta_{ij} - \alpha\right])}, \tag{2.10}$$

where θ_{ij} is the elevation angle of the UAV at the aggregation point $\dot{\psi}_j$ with regard to the device n_i located at $\psi_i = \{x_i, y_i, 0\}$ which is expressed as $\theta_{ij} = \frac{180}{\pi} \tan^{-1}(\frac{\dot{z}_j}{r_{ij}})$, with $r_{ij} = \sqrt{(x_i - \dot{x}_j)^2 + (y_i - \dot{y}_j)^2}$, while α and β are parameters depending on the carrier frequency and environment type such as dense urban, urban, or rural. Finally, $\varphi_{ij}\,(\text{LoS})$ and $\varphi_{ij}\,(\text{NLoS})$ are the path losses for LoS and non line-of-sight (NLoS) connections, respectively, and can be expressed as

$$\varphi_{ij}\,(\text{LoS}) = 10\epsilon \log_{10}\left(\frac{4\pi f_c}{c}\|\psi_i - \dot{\psi}_j\|_2\right) + \zeta_{\text{LoS}}, \tag{2.11}$$

$$\varphi_{ij}\,(\text{NLoS}) = 10\epsilon \log_{10}\left(\frac{4\pi f_c}{c}\|\psi_i - \dot{\psi}_j\|_2\right) + \zeta_{\text{NLoS}}, \tag{2.12}$$

where f_c is the carrier frequency, ϵ is the path loss exponent, c is the speed of light, and ζ_{LoS} and ζ_{NLoS} represent the mean additional losses for LoS and NLoS, respectively.

2.5 Acoustic Channel Model

For a distance ℓ and a frequency f, the attenuation factor $\Lambda(\ell, f)$ of an underwater acoustic channel is given by [24]:

$$10 \log \Lambda(\ell, f) = \kappa 10 \log \ell + \ell 10 \log \varrho\,(f)\,, \tag{2.13}$$

where $\kappa = 1.5$ for practical spreading, and $\varrho\,(f)$ is the absorption coefficient which can be obtained using the Thorp's formula as follows [25]:

$$10 \log \varrho\,(f) = \frac{0.11 f^2}{1 + f^2} + \frac{44 f^2}{4100 + f^2} + \frac{2.75 f^2}{10^4} + 0.003. \tag{2.14}$$

The ambient noise in underwater acoustic channel is affected by four components: turbulence Ω_t, shipping Ω_ϖ, waves Ω_w, and thermal noise Ω_{th}. The following formulae give

the power spectral density of these noise components in dB re μPa per Hz as a function of frequency in kHz

$$\begin{aligned} 10\log\Omega_t(f) =& 17 - 30\log f \\ 10\log\Omega_{\varpi}(f) =& 40 + 20(\varpi - 0.5) + 26\log f - 60\log(f + 0.03) \\ 10\log\Omega_w(f) =& 50 + 7.5w^{\frac{1}{2}} + 20\log f - 40\log(f + 0.4) \\ 10\log\Omega_{\text{th}}(f) =& -15 + 20\log f, \end{aligned} \tag{2.15}$$

with $0 \leq \varpi \leq 1$ as the shipping activity factor and w as the wind speed at the surface in m/s. Consequently, the noise power spectral density of the ambient noise can be expressed as

$$\Omega(f) = \Omega_t(f) + \Omega_{\varpi}(f) + \Omega_w(f) + \Omega_{\text{th}}(f). \tag{2.16}$$

The usable bandwidth of the channel is defined as $B(\ell) = b\ell^{-\beta}$, where the parameters b and β depend on the target signal-to-noise-ratio (SNR) [24]. The acoustic transmit power P_t^{aco} can be converted into electrical transmit power P_t using the following empirical relation $P_t^{aco} = \zeta P_t 10^{17.2}$, where ζ is the conversion efficiency and $10^{-17.2}$ is a conversion factor [26]. Consequently, the SNR at a receiver distance ℓ from the transmitter can be expressed as

$$\gamma(\ell) = \frac{\zeta P_t 10^{17.2}}{B(\ell)\,\Omega\left(f_0(\ell)\right)\Lambda\left(\ell, f_0(\ell)\right)}, \tag{2.17}$$

where $f_0(\ell)$ is the optimal carrier frequency at a given distance ℓ [24].

The latency of transmitting I (bits) over distance ℓ (m) in the acoustic channel can be calculated as [27]

$$L(I, \ell) = \frac{I}{\varphi B(\ell)} + \frac{\ell}{c_s}, \tag{2.18}$$

where φ is the modulation bandwidth efficiency in bps/Hz and c_s is the underwater acoustic propagation speed in m/s, which depends on the water depth, temperature, and salinity.

The acoustic power $P_t(\ell)$ can be converted into electrical power using the following empirical relation

$$P_t^{el}(\ell) = P_t(\ell) \times 10^{-17.2}/\zeta. \tag{2.19}$$

Let P_r denote the power consumed by the receiver's circuitry, the total energy consumed to transmit one bit between two devices over a single hop of length ℓ can be calculated as [27]

$$E^t(\ell) = \frac{P_t^{el}(\ell) + P_r}{\varphi B(\ell)}, \tag{2.20}$$

where φ is the modulation bandwidth efficiency in bps/Hz.

Chapter 3

Energy-Efficient Data Aggregation

3.1 Background and Motivation

Data aggregation is one of the primary goals of IoT/IoUT networks. Two main data aggregation policies can be found in the literature, namely mobile agent-based and client-server-based. In the former, one/several mobile device/s travel and visit all (or a group of) the devices to collect their sensed data. In the latter, devices deliver data to one (multiple) sink device(s) in a multi-hop manner. The latter can be implemented in cluster, aggregation tree, or centralized form. In the cluster aggregation form, all devices are grouped into various clusters. Each cluster consists of a header device, which is known as a cluster-head. The cluster-head receives data from its members and sends the resulting data to the sink. In the aggregation tree form, all devices are organized to construct a hierarchical tree, in which each device sends its sensed data to a parent device for the aggregation process. Accordingly, the data aggregation process is performed from the leaf devices to the sink. In the centralized aggregation form, all devices send their data to a most powerful one through

the shortest route. The header aggregates the data, and sends the results to the sink [28].

Naturally, data measured by devices located within close proximity of an event is expected to be correlated. Transmission of redundant correlated data is not only unnecessary but also costly, which degrades the network efficiency. The effect of such correlation is more severe in the underwater environment which is characterized by limited bandwidth, and communications costs in this environment are far greater than sensing and computational costs. In-network processing is a promising solution, in which raw data is processed locally in intermediate devices to remove correlation and fused uncorrelated data is transmitted to the data gathering center, which is more efficient than sending the raw data [29]. However, local processing device selection should be optimized to reduce data traffic, increase the processing efficiency, and satisfy the network topology constraints. This chapter introduces energy-efficient data aggregation frameworks from a set of devices. Two frameworks are introduced, namely: (1) Multi-sink data aggregation, in which a set of data aggregating stations with fixed-locations are aggregating the data; (2) A mobile data aggregation center gathers the data from the devices.

3.1.1 Related Work

Recently, increased attention has been paid to developing efficient data aggregation algorithms to minimize the use of energy and communication resources. Several studies have proposed different algorithms and schemes to aggregate data in terrestrial communication networks (e.g., [30–37]) and underwater networks (e.g., [38–40]). In [30], a delay-aware and energy-efficient data aggregation method was proposed. The authors minimized the data transmission delay by providing energy-efficient routing paths and intelligently se-

lecting a direct-forwarding scheme for delay-sensitive data transmission to minimize end-to-end delay; a wait-forwarding scheme was exploited for delay-tolerant data transmission to reduce total energy consumption. In [31], the system efficiency for both minimal total energy consumption and min-max per-device energy consumption was considered. The authors developed throughput-efficient scheduling of transmissions in various data aggregation scenarios and shortest-path routes from devices to the sink were planned. In [32], a cluster-based data aggregation protocol based on the devices' computing capabilities was presented. A three-layer cluster-based mechanism for energy-efficient data aggregation was proposed in [33]. In this mechanism, each device transmits its residual energy and location coordination information to the sink for selecting the upper cluster-heads using fuzzy c-means clustering. A power control-based protocol for IoUT was explored in [34]. This protocol considers the link quality, neighborhood density, distance, and energy consumption to select the suitable transmission power level at each device. In [35], a multi-phase data aggregation algorithm was investigated to optimally select the set of active devices to balance the energy load and maximize the data utility. A multiple mobile agents scenario with multi-hop hierarchical clustering-based data aggregation was proposed in [36]. The objective is to minimize the dissipated energy by incorporating clustering schemes and data aggregation. A repositioning framework was proposed in [37] to relocate the devices from their random initial locations to optimized positions such that the energy consumption is reduced and the coverage area increased.

In the underwater networks, some studies (e.g., [38–40], and references therein) have proposed marine data aggregation algorithms and schemes. In [38], autonomous underwater vehicles (AUVs) and multi-hop transmission data aggregation approaches were proposed. The objective is to balance the load and energy consumption of the devices and to

reduce the transmission delay. A stratification-based data collection scheme for underwater networks was investigated in [39]. In this scheme, a set of devices is divided into two groups: the first group employs a multi-hop forwarding approach for data collection; in the second group, a neighbor density clustering-based AUV data aggregation approach is applied. In [40], a sender-receiver role-based scheduling scheme was proposed to provide computation-utilization and high energy-efficiency for smart underwater objects.

This chapter introduces two energy-efficient data aggregation frameworks, the first framework assigns a role for each device in the network to maximize the total uncorrelated data and minimize the energy consumption. The second framework minimizes the energy consumption in a UAV-enabled data aggregation scenario.

3.2 Role Assignment for Multi-Sink Data Aggregation

The following question is addressed: *For a set of spatially distributed devices in an area of interest, what is the role of each device in maximize the total gathered information with minimal energy consumption?* To answer this question, a device-role assignment framework is proposed, in which not only the optimum set of active devices is selected, but also the role of each active device is optimally assigned. The objective is to maximize the aggregated information and minimize the total energy consumption.

3.2.1 System Model

A multi-sink data gathering scenario is considered, in which a set $\mathcal{N} = \{n_i\}_{i=1}^{N}$ of N devices randomly scattered in an area of interest to gather and send data to a set $\mathcal{M} = \{m_j\}_{j=1}^{M}$ of M data aggregation stations (sinks). The devices and data gathering stations

are located at $\boldsymbol{\psi} = [\psi_i]_{N\times 3}$ and $\dot{\boldsymbol{\psi}} = [\dot{\psi}_j]_{M\times 3}$, respectively, where $\psi_i = \{x_i, y_i, z_i\}$ and $\dot{\psi}_j = \{\dot{x}_j, \dot{y}_j, \dot{z}_j\}$ denote the geographical coordinates of device n_i and data gathering station m_j, respectively. Each device n_i has a communication range ϑ_i. The inter-device communication link matrix $\mathbf{C} = [c_{ik}]_{N\times N}$ is defined such that

$$c_{ik} = \begin{cases} 1, & \text{if } \vartheta_i \geq \ell_{ik} \\ 0, & \text{otherwise,} \end{cases} \tag{3.1}$$

where ℓ_{ik} is the distance between device n_i and device n_k. The communication links between the devices and data gathering stations are represented using the matrix $\dot{\mathbf{C}} = \left[\dot{c}_{ij}\right]_{N\times M}$ such that

$$\dot{c}_{ij} = \begin{cases} 1, & \text{if } \vartheta_i \geq \dot{\ell}_{ij} \\ 0, & \text{otherwise,} \end{cases} \tag{3.2}$$

where $\dot{\ell}_{ij}$ is the distance between device n_i and data gathering stations m_j. At each decision-making time interval, a network management center assigns one of the following roles to each device:

- Sensor type 1: which transmits its data to a data gathering station;
- Sensor type 2: which transmits its data to an aggregator device;
- An aggregator type 1: which aggregates its data and data from other devices and transmits the result to a data gathering station;
- An aggregator type 2: which aggregates its data and data from other devices and transmits the result to an aggregator device;
- Inactive device: which switches to sleep mode.

Let us define a device-role assignment decision $\boldsymbol{\eta} = [\eta_{il}]_{N\times 5}$ such that $\eta_{ir} = 1$ if $r = 1$ and device n_i acts as a sensor of type 1; $r = 2$ and device n_i acts as a sensor of type 2; $r = 3$ and device n_i acts as a data aggregator of type 1; $r = 4$ and device n_i acts as a data aggregator of type 2; $r = 5$ and device n_i acts as an inactive device; and $\eta_{ir} = 0$ otherwise. It is clear that $\boldsymbol{\eta}$ assigns a role to each device; however, it fails to associate each sensor of type 2 or aggregator of type 2 with one of the available aggregator devices. Consequently, $\boldsymbol{\mu} = [\mu_{ik}]_{N\times N}$ is defined such that $\mu_{ik} = 1$ if $n_i \in \mathcal{N}$ is a sensor of type 2 or an aggregator of type 2 and decides to send the data to aggregator $n_k \in \mathcal{N}$, and $\mu_{ik} = 0$ otherwise. Furthermore, each sensor of type 1 and aggregator of type 1 should be associated with one of the data gathering stations. Consequently, $\dot{\boldsymbol{\mu}} = \left[\dot{\mu}_{ij}\right]_{N\times M}$ is defined such that $\dot{\mu}_{ij} = 1$ if $n_i \in \mathcal{N}$ is a sensor of type 1 or an aggregator of type 1 and decides to send the data to gathering station $m_j \in \mathcal{M}$, and $\dot{\mu}_{ij} = 0$ otherwise.

For a given $\boldsymbol{\eta}$ and $\boldsymbol{\mu}$, data of each sensor of type 2 and aggregator of type 2 reaches one of the gathering stations via a sequence (one or more) of aggregator devices. Consequently, to represent the data path from a sensor of type 2 or aggregator of type 2 to the gathering station, $\boldsymbol{\mathcal{T}}(\boldsymbol{\eta}, \boldsymbol{\mu}) = [t_{ik}]_{N\times N}$ is defined such that

$$t_{ik} = \begin{cases} 1, & \text{if } n_i \text{ contributes to forwarding data from } n_k, \\ 0, & \text{otherwise.} \end{cases} \tag{3.3}$$

A sequence of aggregator devices should contains an aggregator device type 1 to guarantee that the data of all devices associated with this sequence reaches a gathering station and to prevent aggregation loop, this can be guaranteed if $\prod_{k=1}^{N}(1 - \eta_{k3}t_{ki}) = 0$ for each sensor of type 2 and aggregator type 2.

For a given role assignment decision $\boldsymbol{\eta}$, similar to (2.2), the joint entropy (in bits) of the

active devices (devices with $\eta_{i5} = 0$) can be expressed as

$$\mathcal{H}(\boldsymbol{\eta}) = \varpi \left(L + L \sum_{i=1}^{N} (1 - \eta_{i5}) \left[1 - \frac{1}{d_i(\boldsymbol{\eta})/\rho + 1} \right] \right), \tag{3.4}$$

where $\varpi = 1 - \prod_{i=1}^{N} \eta_{i5}$ and $d_i(\boldsymbol{\eta})$ is the minimum distance between the device n_i and all devices n_k with $\eta_{k5} = 0 \; \forall k = 1, 2, \ldots, i-1$. The amount of data that is transmitted by a device n_i to another device or to one of the data gathering station can be expressed as

$$\mathcal{H}_i(\boldsymbol{\eta}, \boldsymbol{\mu}) = (1 - \eta_{i5}) \left(L + L \sum_{k=1}^{N} t_{ik} \left[1 - \frac{1}{\ell_{ik}/\rho + 1} \right] \right). \tag{3.5}$$

The energy is consumed by devices to realize three primary functions: data acquisition, data processing (aggregation) and data transmission. E_i^a is the energy consumed by device n_i to sense and obtain L bits of raw data. For a given $\boldsymbol{\eta}$, $\boldsymbol{\mu}$, and $\dot{\boldsymbol{\mu}}$, the energy consumption of device n_i can be expressed using (3.6).

$$\begin{aligned}
\mathcal{E}_i(\boldsymbol{\eta}, \boldsymbol{\mu}, \dot{\boldsymbol{\mu}}) = {} & \eta_{i1} \left(E_i^a + \sum_{j=1}^{M} \dot{\mu}_{ij} E^t\left(\dot{\ell}_{ij}\right) L \right) + \eta_{i2} \left(E_i^a + \sum_{k=1}^{N} \mu_{ik} E^t(\ell_{ik}) L \right) + \eta_{i5} E_i^s \\
& + \eta_{i3} \left(E_i^a + E_i^p \left(L + \sum_{k=1}^{N} \mu_{ki} \mathcal{H}_k(\boldsymbol{\eta}, \boldsymbol{\mu}) \right) + \sum_{j=1}^{M} \dot{\mu}_{ij} E^t\left(\dot{\ell}_{ij}\right) \mathcal{H}_i(\boldsymbol{\eta}, \boldsymbol{\mu}) \right) \\
& + \eta_{i4} \left(E_i^a + E_i^p \left(L + \sum_{k=1}^{N} \mu_{ki} \mathcal{H}_k(\boldsymbol{\eta}, \boldsymbol{\mu}) \right) + \sum_{k=1}^{N} \mu_{ik} E^t(\ell_{ik}) \mathcal{H}_i(\boldsymbol{\eta}, \boldsymbol{\mu}) \right).
\end{aligned} \tag{3.6}$$

In this equation, $E_i^p = \delta \varepsilon_i$ is the energy consumed by device n_i to process (aggregate) one bit with δ depending on the computational complexity of the monitoring task and ε_i as the energy value consumed by the central processing unit (CPU) of n_i to perform one computation cycle. $E^t(\ell)$ is the energy consumed to transmit one bit between two devices over a single hop of length ℓ. $E^t(\ell)$ can be obtained using (2.20) for the acoustic underwater

scenario or using first-order radio model (similar to Eq. (1) of [41] and references therein) for the terrestrial scenario. E_i^s is the energy consumed by device n_i in the sleep mode.

3.2.2 Problem Formulation

The role of each device depends on its connectivity to the gathering stations, connectivity to other devices, distance to other devices, and preference of the decision-maker to maximize the gathered information or minimize the consumed energy. Consequently, the active devices set should be intelligently selected and a suitable role should be assigned to each active device. The total consumed energy is $\mathcal{E}\left(\boldsymbol{\eta}, \boldsymbol{\mu}, \dot{\boldsymbol{\mu}}\right) = \sum_{i=1}^{N} \mathcal{E}_i\left(\boldsymbol{\eta}, \boldsymbol{\mu}, \dot{\boldsymbol{\mu}}\right)$, where $\mathcal{E}_i\left(\boldsymbol{\eta}, \boldsymbol{\mu}, \dot{\boldsymbol{\mu}}\right)$ is the energy consumed by device n_i which can be calculated using (3.6). The total gathered information can be obtained using (3.18). Our objective is to maximize the gathered information $\mathcal{H}\left(\boldsymbol{\eta}\right)$ and minimize the total consumed energy $\mathcal{E}\left(\boldsymbol{\eta}, \boldsymbol{\mu}, \dot{\boldsymbol{\mu}}\right)$. Keeping in mind the trade-off between $\mathcal{H}\left(\boldsymbol{\eta}\right)$ and $\mathcal{E}\left(\boldsymbol{\eta}, \boldsymbol{\mu}, \dot{\boldsymbol{\mu}}\right)$, and the fact that they have different orders of magnitude and ranges, they should be transformed such that they have similar ranges [42]. Consequently, a multi-objective weighted sum objective function is defined as follows

$$O\left(\boldsymbol{\eta}, \boldsymbol{\mu}, \dot{\boldsymbol{\mu}}\right) = \lambda \left[\frac{\mathcal{H}\left(\boldsymbol{\eta}\right) - H_0}{H_{\text{max}} - H_0}\right] + \left(1 - \lambda\right) \left[1 - \frac{\mathcal{E}\left(\boldsymbol{\eta}, \boldsymbol{\mu}, \dot{\boldsymbol{\mu}}\right) - E_0}{E_{\text{max}} - E_0}\right], \tag{3.7}$$

where $0 \leq \lambda \leq 1$ is a relative weight to give preference to maximize the information or minimize the energy, H_{max} is the maximum amount of information that can be generated by $\mathcal{N}$, E_{max} is the maximum energy consumption which can be calculated when all the devices transmit their data to the stations directly, H_0 is the minimum amount of information which is zero (bits), and E_0 is the minimum value of the energy consumption (the energy consumption when all devices are switched to the sleep mode), which is expressed

as $E_0 = \sum_{i=1}^{N} E_i^s$. The optimization problem is formulated as shown in (3.8). Constraint (3.8b) guarantees that a role is assigned to each device. Constraint (3.8c) guarantees that each sensor of type 2 and aggregator of type 2 offloads its data to one aggregator device and there is a communication link between them. Constraint (3.8d) guarantees that each sensor of type 1 and aggregator of type 1 offloads its data to one aggregation station and there is a communication link between them. Constraint (3.8e) guarantees that each aggregator aggregates data from at least one device. Constraint (3.8f) guarantees that the data of each sensor of type 2 or aggregator type 2 reaches a gathering station without aggregation loops.

$$\text{P1} \max_{\boldsymbol{\eta},\boldsymbol{\mu},\dot{\boldsymbol{\mu}}} \; O\left(\boldsymbol{\eta},\boldsymbol{\mu},\dot{\boldsymbol{\mu}}\right), \tag{3.8a}$$

$$\text{s.t.} \quad \sum_{r=1}^{5} \eta_{ir} = 1, \forall\, n_i \in \mathcal{N}, \tag{3.8b}$$

$$\sum_{k=1}^{N} \mu_{ik} c_{ik} = \eta_{i2} + \eta_{i4}, \forall\, n_i \in \mathcal{N}, \tag{3.8c}$$

$$\sum_{j=1}^{M} \dot{\mu}_{ij} \dot{c}_{ij} = \eta_{i1} + \eta_{i3}, \forall\, n_i \in \mathcal{N}, \tag{3.8d}$$

$$\sum_{i=1}^{N} \mu_{ik} \geq \eta_{k3} + \eta_{k4}, \forall\, n_k \in \mathcal{N}, \tag{3.8e}$$

$$(\eta_{k2} + \eta_{k4}) \prod_{i=1}^{N} (1 - \eta_{i3} t_{ik}) = 0, \forall\, n_k \in \mathcal{N}, \tag{3.8f}$$

$$\eta_{ir} \in \{0,1\}, \forall\, n_i \in \mathcal{N}\ \&\ 1 \leq r \leq 5, \tag{3.8g}$$

$$\mu_{ik}, \dot{\mu}_{ij} \in \{0,1\}, \forall\, n_i, n_k \in \mathcal{N}, m_j \in \mathcal{M}. \tag{3.8h}$$

It is worth noting that $\boldsymbol{\eta}$, $\boldsymbol{\mu}$, and $\dot{\boldsymbol{\mu}}$ are coupled variables and the search space of the optimization problem in (5.20) is $2^{(5+N+M)N}$. Consequently, the optimum solution complexity is prohibitive for a reasonable number of devices and stations. The next section

presents a more efficient method to solve the problem using ACO approach.

3.2.3 Proposed Solution Approach

In the proposed ACO algorithm, a colony of A ants collaborate to solve (3.8). Each ant $a \in A$ travels through a two-stage tour, in which it assigns a role for each device in the first stage and associates the sensors and aggregators with one of the gathering stations or with aggregators in the second stage.

In the first stage, the probability that ant a assigns role r to device n_i (i.e., the probability to set $\eta_{ir}^{(a)} = 1$) is obtained as

$$\bar{\pi}_{ir}^{(a)} = \frac{\left(\bar{\tau}_{ir}\right)^{\alpha}\left(\bar{\varrho}_{ir}\right)^{\beta}}{\sum_{r=1}^{5}\left(\bar{\tau}_{ir}\right)^{\alpha}\left(\bar{\varrho}_{ir}\right)^{\beta}}, \tag{3.9}$$

where $\bar{\tau}_{ir}$ is the trail pheromone and $\bar{\varrho}_{ir}$ is the attractiveness of assigning role r to device n_i.[1] The latter is set to be

$$\bar{\varrho}_{ir} = \begin{cases} 1 - \prod_{j=1}^{M}(1 - \dot{c}_{ij}), & \text{if } r = 1, 3, \\ 1 - \prod_{k=1}^{N}(1 - c_{ik}), & \text{if } r = 2, 4, \\ 1, & \text{if } r = 5, \end{cases} \tag{3.10}$$

to provide similar attractiveness to assigning a role device n_i. However, if the device n_i does not have a communication link with any gathering station (i.e., $\prod_{j=1}^{M}(1 - \dot{c}_{ij}) = 1$), the attractiveness $\bar{\varrho}_{ir} = 0$ for $r = 1$ and $r = 3$. Consequently, ant a does not assign a sensor type 1 or aggregator type 1 role to device n_i. Similarly, if the device n_i does not have a communication link with any other device (i.e., $\prod_{k=1}^{N}(1 - c_{ik}) = 1$), the attractiveness $\bar{\varrho}_{ir} = 0$ for $r = 2$ and $r = 4$. Consequently, ant a does not assign a sensor type 2 or

[1] α and β control the influence of the pheromone and attractiveness, respectively.

aggregator type 2 role to device n_i. It is worth noting that $\sum_{k=1}^{N} c_{ik}(\eta_{k3}^{(a)} + \eta_{k4}^{(a)}) = 0$ indicates that device n_i does not have a communication link to any aggregator device; in such a case, if n_i was assigned a role of sensor type 2 (or aggregator type 2), it will be re-assigned a role of sensor type 1 (or aggregator type 1) if it has a communication link with at least one aggregation station, respectively; otherwise, it switches to the sleep mode.

In the second stage, ant a associates the active devices with one of the gathering stations or another device. The attractiveness of associating device n_i with device n_k is set to be

$$\bar{\bar{\varrho}}_{ik} = c_{ik}\frac{D}{\ell_{ik}}, \tag{3.11}$$

which suggests more attractiveness to associating device n_i with the nearest device with $c_{ik} = 1$, and D is a constant [43]. The inter-device association decision $\boldsymbol{\mu}^{(a)}$ is formed by ant a such that the probability to set $\mu_{ik}^{(a)} = 1$ is

$$\bar{\bar{\pi}}_{ik}^{(a)} = \frac{\left(\eta_{i2}^{(a)} + \eta_{i4}^{(a)}\right)\left(\eta_{k3}^{(a)} + \eta_{k4}^{(a)}\right)\left(\bar{\bar{\tau}}_{ik}\right)^{\alpha}\left(\bar{\bar{\varrho}}_{ik}\right)^{\beta}}{\sum_{l=1}^{N}\left(\eta_{l3}^{(a)} + \eta_{l4}^{(a)}\right)\left(\bar{\bar{\tau}}_{il}\right)^{\alpha}\left(\bar{\bar{\varrho}}_{il}\right)^{\beta}}, \tag{3.12}$$

where $\bar{\bar{\tau}}_{ik}$ is the trail pheromone.[1] Once $\boldsymbol{\mu}^{(a)}$ is obtained, ant a obtains $\boldsymbol{\mathcal{T}}(\boldsymbol{\eta}^{(a)}, \boldsymbol{\mu}^{(a)})$; in-case a group of type 2 aggregators form an aggregation loop, any of these aggregators is re-assigned the aggregation type 1 role and if none has a communication link with a gathering station, all of these devices and devices connected to them are switched to the sleep mode. If ant a assigns the role of aggregator type 1 (or type 2) to device n_i and no device is associated with n_i, n_i is re-assigned the sensor type 1 (or type 2) role. The attractiveness of associating device n_i with station m_j is set to be

$$\dot{\bar{\bar{\varrho}}}_{ij} = \dot{c}_{ij}\frac{D}{\dot{\ell}_{ij}}, \tag{3.13}$$

which suggests more attractiveness to associating device n_i with the nearest station with $\dot{c}_{ij} = 1$. The device-station association decision $\dot{\boldsymbol{\mu}}^{(a)}$ is formed by ant a such that the

probability to set $\dot{\mu}_{ij}^{(a)} = 1$ is

$$\dddot{\pi}_{ij}^{(a)} = \frac{\left(\eta_{i1}^{(a)} + \eta_{i3}^{(a)}\right) (\dddot{\tau}_{ij})^{\alpha} (\dddot{\varrho}_{ij})^{\beta}}{\sum_{q=1}^{M} (\dddot{\tau}_{iq})^{\alpha} (\dddot{\varrho}_{iq})^{\beta}}, \tag{3.14}$$

where $\dddot{\tau}_{ij}$ is the trail pheromone.[1]

It is worth noting that based on the above described procedure, $\boldsymbol{\eta}^{(a)}$, $\boldsymbol{\mu}^{(a)}$, and $\dot{\boldsymbol{\mu}}^{(a)}$ represent a feasible solution to (3.8). The quality of this solution is reflected in its deposit pheromone $\Delta\tau^{(a)}$, which is set to be

$$\Delta\tau^{(a)} = O(\boldsymbol{\eta}^{(a)}, \boldsymbol{\mu}^{(a)}, \dot{\boldsymbol{\mu}}^{(a)})Q, \tag{3.15}$$

where Q is a constant.

In the proposed ACO, a global updating rule is adopted in which the globally best and second-best ants (i.e., ants which obtain the highest and second-highest $O(\boldsymbol{\eta}^{(a)}, \boldsymbol{\mu}^{(a)}, \dot{\boldsymbol{\mu}}^{(a)})$ $\forall a \in A$) are allowed to deposit their pheromone [44]. The pheromone in the first and second stages are updated as follows:

$$\bar{\tau}_{ir} \leftarrow (1-\sigma)\, \bar{\tau}_{ir} + \eta_{ir}^{(a)} \Delta\tau^{(a)}, \tag{3.16a}$$

$$\bar{\bar{\tau}}_{ik} \leftarrow (1-\sigma)\, \bar{\bar{\tau}}_{ik} + \mu_{ik}^{(a)} \Delta\tau^{(a)}, \tag{3.16b}$$

$$\dddot{\tau}_{ij} \leftarrow (1-\sigma)\, \dddot{\tau}_{ij} + \dot{\mu}_{ij}^{(a)} \Delta\tau^{(a)}, \tag{3.16c}$$

where σ is the pheromone evaporation coefficient. The algorithm repeats I colonies and returns the best solution found so far. The ACO algorithm is presented as Algorithm 8.

An ant a performs $\mathcal{O}(5NM)$ and $\mathcal{O}(N^2M)$ operations in the first and second stages, respectively. Evaluating the objective function requires $\mathcal{O}(N^2 + N^2) = \mathcal{O}(N^2)$ operations. Finally, deposing the pheromone requires $\mathcal{O}(5N + N^2 + NM) = \mathcal{O}(N^2 + NM)$ operations. Consequently, the computational complexity of the proposed ACO algorithm

Algorithm 1 ACO algorithm for IoUT role assignment.

Input: N, M, $\boldsymbol{C}$, $\dot{\boldsymbol{C}}$, α, β, A, and I;

Initialize $\bar{\tau}_{ir}$, $\bar{\bar{\tau}}_{ik}$, and $\dot{\bar{\tau}}_{ij}$, $\forall 1 \leq i, k \leq N$, $1 \leq j \leq M$, $1 \leq r \leq 5$;

$O \leftarrow -\infty$;

for Iteration $= 1$ to I do

  $O_1 \leftarrow -\infty$; $O_2 \leftarrow -\infty$;

  for $a = 1$ to A do

  Obtain $\boldsymbol{\eta}^{(a)}$, $\boldsymbol{\mu}^{(a)}$, and $\dot{\boldsymbol{\mu}}^{(a)}$ using (5.22), (3.12), and (3.14), respectively;

  Evaluate $O(\boldsymbol{\eta}^{(a)}, \boldsymbol{\mu}^{(a)}, \dot{\boldsymbol{\mu}}^{(a)})$ using (3.7);

   if $O < O(\boldsymbol{\eta}^{(a)}, \boldsymbol{\mu}^{(a)}, \dot{\boldsymbol{\mu}}^{(a)})$

    $\boldsymbol{\eta}^{*} \leftarrow \boldsymbol{\eta}^{(a)}$; $\boldsymbol{\mu}^{*} \leftarrow \boldsymbol{\mu}^{(a)}$; $\dot{\boldsymbol{\mu}}^{*} \leftarrow \dot{\boldsymbol{\mu}}^{(a)}$;

   end if

   if $O_1 < O(\boldsymbol{\eta}^{(a)}, \boldsymbol{\mu}^{(a)}, \dot{\boldsymbol{\mu}}^{(a)})$

    $\boldsymbol{\eta}^{(a_1)} \leftarrow \boldsymbol{T}^{(a)}$; $\boldsymbol{\mu}^{(a_1)} \leftarrow \boldsymbol{\mu}^{(a)}$; $\dot{\boldsymbol{\mu}}^{(a_1)} \leftarrow \dot{\boldsymbol{\mu}}^{(a)}$;

     else if $O_2 < O(\boldsymbol{\eta}^{(a)}, \boldsymbol{\mu}^{(a)}, \dot{\boldsymbol{\mu}}^{(a)})$

      $\boldsymbol{\eta}^{(a_2)} \leftarrow \boldsymbol{T}^{(a)}$; $\boldsymbol{\mu}^{(a_2)} \leftarrow \boldsymbol{\mu}^{(a)}$; $\dot{\boldsymbol{\mu}}^{(a_2)} \leftarrow \dot{\boldsymbol{\mu}}^{(a)}$;

   end if

  end for

   Deposit pheromone of a_1 and a_2 using (5.25);

end for

Return $\boldsymbol{\eta}^{*}$, $\boldsymbol{\mu}^{*}$, and $\dot{\boldsymbol{\mu}}^{*}$.

is $\mathcal{O}([N^2M + N^2]AI + [N^2 + NM]I) = \mathcal{O}(N^2MAI)$. This computational complexity is experienced by the network management center, and thus, it does not represent a severe problem due to the high processing power and energy abilities of this center. On the other hand, this computational complexity comes at a cost of reducing the energy consumption of the devices, which is constrained by limited battery life-time. It is worth mentioning that the computational complexity of the proposed ACO algorithm is significantly lower than the complexity of exhaustive search, which is $\mathcal{O}(N^2 2^{(5+N+M)N})$.

3.2.4 Simulation Results

To study and evaluate the effects of the main parameters on the performance of the proposed role assignment framework, an underwater area of interest is considered. The performance of the proposed framework is compared with a baseline approach, in which each device transmits its data to the nearest gathering station and switches to the sleep mode if no gathering station lies within its transmission range. In obtaining these results, a binary phase-shift-keying system is considered [27] and the usable bandwidth of the acoustic communication channel is adapted according to the model in Section 2.5. The default simulation parameters summarized in Table 3.1 are considered in the results, unless otherwise stated.

Figure 3.1 illustrates the objective function and the number of active devices of the proposed framework versus the relative weight λ. This figure illustrates both ACO solution and optimal solution, obtained through exhaustive search. It is seen that the proposed ACO algorithm achieves near-optimum performance and as λ increases, the proposed framework activates more devices to increase the gathered data.

Table 3.1: Simulation parameters of the multi-sink data aggregation framework.

Parameter	Value	Parameter	Value
UAoI dimensions	1×1 km^2	Water depth	300 m
L	2048 bits [27]	φ	0.75 [27]
ρ	10^2	δ	1900 cycles per bit
χ	10 dB [27]	ξ	-10 dB [27]
$\hat{\gamma}$	10 dB [27]	E_i^s	10 nJ [45]
E_i^a	50 μJ [45]	ε_i	10 nJ per bit [45]
ζ	0.25 [27]	N	20 devices
ϑ_i	400 m	M	3 stations
α	1 [44]	A	100
β	1 [44]	I	10^3

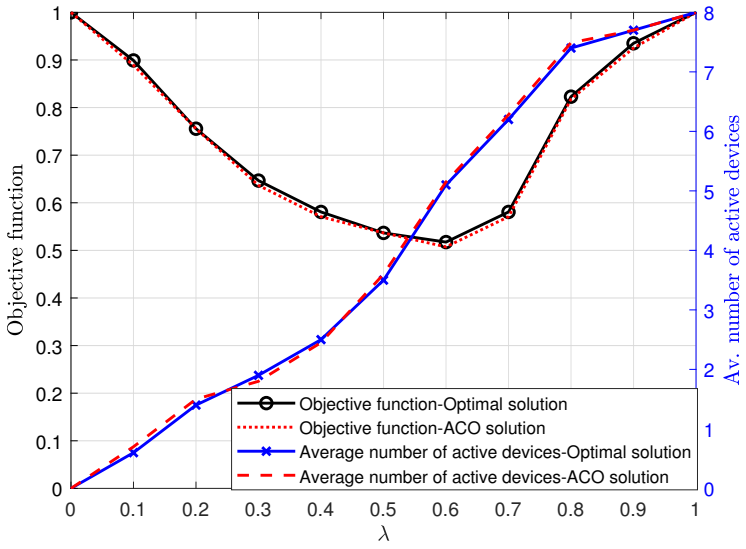

Figure 3.1: The objective function in (3.7) and active devices versus the relative weight λ.

Figure 3.2 shows the total gathered information and energy expenditure of both proposed framework and baseline approach with respect to the degree of correlation ρ. It is

noticed that for low values of ρ, the proposed framework gathers more information with less energy consumption. As ρ increases, the gathered information of both approaches decreases; this is due to the fact that increasing ρ reduces the joint entropy of the available devices. However, it is worth noticing that the proposed framework preserves more energy as ρ increases, whereas the devices in the baseline approach keep sending raw data without exploring the correlation among data. Consequently, the energy consumption of the baseline approach is not a function of ρ.

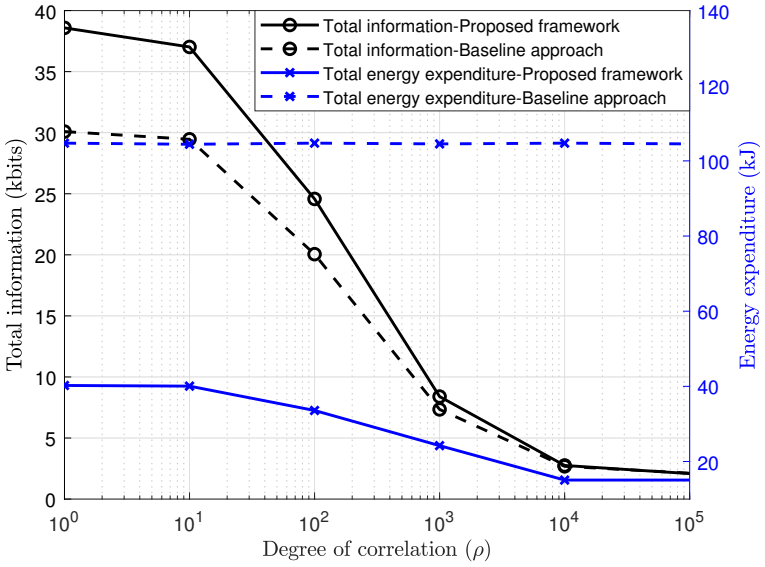

Figure 3.2: Total gathered information and energy consumption versus the degree of correlation ρ.

Figure 3.3 shows the total gathered information and energy consumption of the proposed framework and the baseline approach for different numbers of available devices N. The gathered information of both proposed framework and baseline approach increases as N increases. On the other hand, it is noticed that the energy consumption of the proposed framework is less and flattens as N increases. This is due to the fact that as N increases, the

correlation among the data of the devices increases; however, the baseline approach aggregates this correlated data. On the other hand, increasing N gives the proposed framework diversity to activate a subset of devices and to optimally assign a role to each active device.

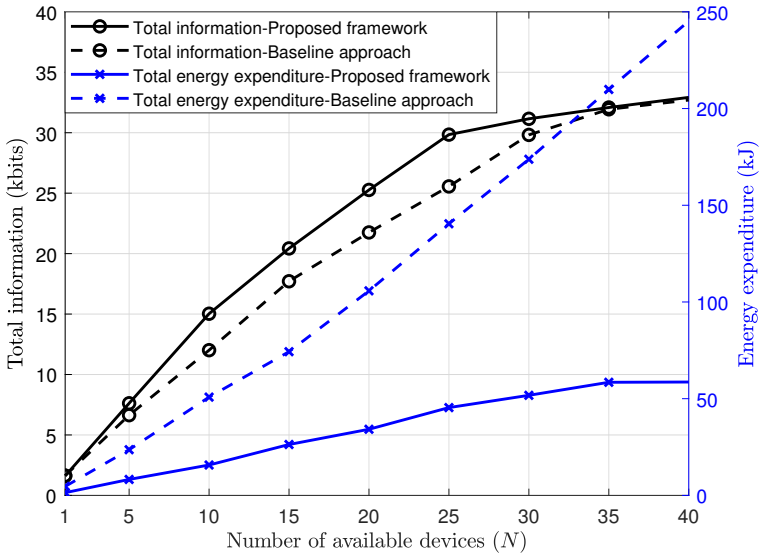

Figure 3.3: Total gathered information and energy consumption versus the number of available devices N.

3.3 Energy-Efficient Data Aggregation Using a UAV

In the IoT data gathering scenario, UAVs can play the role of mobile aggregator nodes that dynamically hover towards the IoT devices, gather their data, and return to a data aggregation center which is beyond the communication range of the IoT devices. If the UAV hovers close to all devices, higher data rates can be achieved, and in turn, this reduces the com-

munication time; however, it leads to higher energy consumption due to the long traveling tour. On the other hand, stopping the UAV at far positions may reduce the traveling energy but the data rates degrade, and thus, the hovering time to gather data at such far trajectories increases. Moreover, gathering correlated data is cost-expensive, and it involves the UAV hovering and gathering repeated data. Consequently, both the active devices and the UAV's path should be optimized to ensure that the required uncorrelated data has been gathered with minimum energy consumption. In this work, unlike existing studies, a framework that takes into account the spatial correlation among IoT devices' data is proposed. Taking into account the trade-off between the amount of aggregated data and the energy consumption, the proposed framework minimizes the energy expenditure of the UAV and devices and guarantees that an adequate amount of uncorrelated data is aggregated.

3.3.1 System Model

The considered system model consists of a set $\mathcal{N} = \{n_i\}_{i=1}^{N}$ of N devices placed in an area of interest. A UAV is responsible for aggregating data from the devices. The maximum number of stop positions (data aggregation points or positions) where the UAV may stop to aggregate data from devices equals N. These aggregation points are represented by $\tilde{\boldsymbol{\psi}} = [\tilde{\psi}_j]_{N\times 3}$, with $\tilde{\psi}_j = \{\tilde{x}_j, \tilde{y}_j, \tilde{z}_j\}$ as the 3D geographical coordinates. At each aggregation point, the UAV aggregates data from a subset of devices using a time division multiple access scheme. Each device obtains L bits of raw data and communicates with the UAV at only one aggregation point.

Let us define the device activation decision $\hat{\boldsymbol{\eta}} = \left[\hat{\eta}_{ij}\right]_{N\times N}$ such that

$$\hat{\eta}_{ij} = \begin{cases} 1, & \text{if } n_i \text{ sends its data to the UAV at } \psi_j, \\ 0, & \text{otherwise.} \end{cases} \tag{3.17}$$

It is worth mentioning that, for a given $\hat{\boldsymbol{\eta}}$, the UAV travels and stops only at each aggregation point with $\sum_{i=1}^{N} \hat{\eta}_{ij} \geq 1$. Initially, the UAV is placed at a docking station located at $\psi_0 = \{x_0, y_0, z_0\}$ to which it has to return after completing the data aggregation. For a given $\hat{\boldsymbol{\eta}}$, the number of aggregation points is $K \leq N$ with $\sum_{i=1}^{N} \hat{\eta}_{ij} \geq 1$. Consequently, the UAV's data aggregation trip $\bar{\boldsymbol{\psi}}\left(\hat{\boldsymbol{\eta}}, \tilde{\boldsymbol{\psi}}\right)$ is defined as a traveling path that starts at ψ_0, passes through all the K aggregation points, and returns to the docking station, i.e., $\bar{\boldsymbol{\psi}}\left(\hat{\boldsymbol{\eta}}, \tilde{\boldsymbol{\psi}}\right) = \left[\bar{\psi}_j\right]_{(K+2)\times 3}$, where $\bar{\psi}_1 = \bar{\psi}_{K+2} = \psi_0$ and $\bar{\psi}_2, \bar{\psi}_3, \ldots, \bar{\psi}_{K+1}$ is an ordered set of all aggregation points with $\sum_{i=1}^{N} \hat{\eta}_{ij} \geq 1$. For a given device activation decision $\boldsymbol{\eta}$, similar to (2.2), the joint entropy of active devices can be expressed as

$$\mathcal{H}\left(\hat{\boldsymbol{\eta}}\right) = \tau\left(\hat{\boldsymbol{\eta}}\right)\left(L + L\sum_{i=1}^{N}\sum_{j=1}^{N}\hat{\eta}_{ij}\left[1 - \frac{1}{d_i\left(\hat{\boldsymbol{\eta}}\right)/\rho_s + 1}\right]\right), \tag{3.18}$$

where $\tau\left(\hat{\boldsymbol{\eta}}\right) = 1 - \prod_{i=1}^{N}\prod_{j=1}^{N}\left[1 - \hat{\eta}_{ij}\right]$ and $d_i\left(\hat{\boldsymbol{\eta}}\right)$ is the minimum distance between the device n_i and all devices n_k with $\eta_{kj} = 1$ for $\forall j$ and $\forall k = 1, 2, \ldots, i-1$.

3.3.2 Problem Formulation

Activating a large number of devices leads to a larger traveling tour for the UAV, and consequently, to an increase in the total consumed energy. On the other hand, a fewer number of participating devices decreases the aggregated information. Consequently, the number of active devices should be intelligently selected and the UAV should stop at suitable aggregation points such that an adequate amount of information is aggregated and the total

consumed energy is decreased. For a given $\hat{\boldsymbol{\eta}}$, the total consumed energy by device n_i can be calculated as

$$\mathcal{E}_i\left(\hat{\boldsymbol{\eta}}, \bar{\boldsymbol{\psi}}\right) = \sum_{j=1}^{N} \hat{\eta}_{ij} \left(E_i^a + \frac{P_{T_i} L}{R_{ij}} \right) + \left(1 - \sum_{j=1}^{N} \hat{\eta}_{ij} \right) E_i^s, \tag{3.19}$$

where P_{T_i} is the transmit power of device n_i and R_{ij} is the data rate and can be obtained using (2.8). The energy dissipated by the UAV during the data aggregation trip can be calculated as

$$\mathcal{E}_{\text{UAV}}\left(\hat{\boldsymbol{\eta}}, \bar{\boldsymbol{\psi}}\right) = \underbrace{\sum_{j=1}^{K+1} \left(P_{\text{hov}} + P_{\text{mov}}\right) \frac{\|\bar{\psi}_{j+1} - \bar{\psi}_j\|_2}{v}}_{\text{Path energy dissipation}} + \underbrace{\sum_{i=1}^{N} \sum_{j=1}^{N} \hat{\eta}_{ij} \frac{\left(P_{\text{hov}} + P_r\right) L}{R_{ij}}}_{\text{Data aggregation energy dissipation}}, \tag{3.20}$$

where P_{hov} and P_{mov} represent the hovering and the moving power consumption of the UAV, which can be obtained using (2.6) and (2.7), respectively. The objective function is expressed as the sum of the energy dissipated by the UAV and devices as follows:

$$\mathcal{E}\left(\hat{\boldsymbol{\eta}}, \bar{\boldsymbol{\psi}}\right) = \lambda \mathcal{E}_{\text{UAV}}\left(\hat{\boldsymbol{\eta}}, \bar{\boldsymbol{\psi}}\right) + (1 - \lambda) \sum_{i=1}^{N} \mathcal{E}_i\left(\hat{\boldsymbol{\eta}}, \bar{\boldsymbol{\psi}}\right). \tag{3.21}$$

where, $0 \leq \lambda \leq 1$ is the relative weight.

Let I (in bits) be the minimum amount of information (uncorrelated data) required to be aggregated[2]; the optimization problem can be formulated as follows:

$$\mathbf{P1} \min_{\hat{\boldsymbol{\eta}}, \bar{\boldsymbol{\psi}}} \ \mathcal{E}\left(\hat{\boldsymbol{\eta}}, \bar{\boldsymbol{\psi}}\right), \tag{3.22a}$$

$$\text{s.t.} \quad \sum_{j=1}^{N} \hat{\eta}_{ij} \leq 1, \ \forall\, 1 \leq i \leq N, \tag{3.22b}$$

$$\mathcal{H}\left(\hat{\boldsymbol{\eta}}\right) \geq I, \tag{3.22c}$$

$$[\sum_{i=1}^{N} \hat{\eta}_{ij}]^{+} \left(h_{\min} \leq \tilde{z}_j \leq h_{\max}\right), \ \forall\, 1 \leq j \leq N, \tag{3.22d}$$

$$\hat{\eta}_{ij} \in \{0, 1\}, \ \forall\, 1 \leq i, j \leq N. \tag{3.22e}$$

[2] I should be upper bounded by the joint entropy of the device set $\mathcal{N}$ (i.e., $0 \leq \Gamma = \frac{I}{H} \leq 1$).

Constraint (3.29b) guarantees that each device communicates with the UAV in at most one data aggregation point. Constraint (3.22c) guarantees that the total aggregated information (uncorrelated data) satisfies the requirement. Constraint (3.22d) guarantees that the UAV hovers within the allowable altitude, where $[\nu]^+ \triangleq \min\{\nu, 1\}$.

3.3.3 Proposed Solution Approach

A greedy algorithm is developed to solve (3.22). First, the algorithm aims to select an optimum set of devices that can provide an adequate amount of information, and to the extent possible, are located in close proximity to the docking station. Secondly, the algorithm determines the aggregation points and the contributing devices associated with each of these points. Finally, the optimized itinerary of the UAV between the aggregation points to complete the aggregation tour is determined.

3.3.3.1 Device Activation Optimization Sub-Problem and the Proposed Genetic Algorithm (GA)

Let us define a device activation indicator $\hat{\boldsymbol{\mu}} = [\hat{\mu}_i]_{N\times 1}$ such that

$$\hat{\mu}_i = \begin{cases} 1, & \text{if } n_i \text{ sends the data to the UAV,} \\ 0, & \text{otherwise.} \end{cases} \tag{3.23}$$

For a given $\hat{\boldsymbol{\mu}}$, similar to (2.2), the joint entropy of the active devices can be written as

$$\mathcal{H}\left(\hat{\boldsymbol{\mu}}\right) = \tau\left(\hat{\boldsymbol{\mu}}\right)\left(L + L\sum_{i=1}^{N}\hat{\mu}_i\left[1 - \frac{1}{d_i\left(\hat{\boldsymbol{\mu}}\right)/\rho + 1}\right]\right), \tag{3.24}$$

where $\tau\left(\hat{\boldsymbol{\mu}}\right) = 1 - \prod_{i=1}^{N}\left[1 - \hat{\mu}_i\right]$ and $d_i\left(\hat{\boldsymbol{\mu}}\right)$ is the minimum distance between device n_i and all devices n_k with $\hat{\mu}_k = 1\ \forall k = 1, 2, \ldots, i-1$. Therefore, with d_{i0} as the distance

between the docking station and device n_i, the contributing devices selection sub-problem can be formulated as

$$\mathbf{P2} \min_{\hat{\boldsymbol{\mu}}} \sum_{i=1}^{N} \mu_i d_{i0}, \tag{3.25a}$$

$$\text{s.t. } \mathcal{H}\left(\hat{\boldsymbol{\mu}}\right) \geq I, \tag{3.25b}$$

$$\hat{\mu}_i \in \{0, 1\}, \; \forall \; 1 \leq i \leq N. \tag{3.25c}$$

The search space for (3.25) is 2^N, which makes finding the optimum solution prohibitive for a reasonable number of devices. Consequently, a GA is devised to solve (3.25). In the proposed GA, a device activation indicator $\hat{\boldsymbol{\mu}}$ is a *chromosome*, which consists of a set of N binary *genes*. The raw fitness value of a chromosome $\hat{\boldsymbol{\mu}}^{(k)}$ is expressed as

$$\chi^{(k)} = \frac{1}{\sum_{i=1}^{N} \hat{\mu}_i^{(k)} d_{i0}}. \tag{3.26}$$

Each chromosome $\hat{\boldsymbol{\mu}}^{(k)}$ is associated with a feasibility indicator which is obtained as follows:

$$\iota^{(k)} = \begin{cases} 1, & \text{if } \mathcal{H}(\hat{\boldsymbol{\mu}}^{(k)}) \geq I, \\ 0, & \text{otherwise.} \end{cases} \tag{3.27}$$

Consequently, the fitness value of the chromosome $\hat{\boldsymbol{\mu}}^{(k)}$ is expressed as

$$\mathcal{X}^{(k)} = \iota^{(k)} \chi^{(k)}. \tag{3.28}$$

The chromosome with the highest $\mathcal{X}^{(k)} > 0$ satisfies (3.25b) and (3.25c) and minimizes (3.25a). The adopted GA to solve (3.25) consists of the following steps:

1. Generate an initial population as a set of Ξ randomly constructed chromosomes.

2. Calculate the fitness of each chromosome using (3.28).

3. Choose the parent chromosomes. Two chromosomes are selected and the chromosome with the highest raw fitness value is chosen as the first parent. The process is repeated to choose the second parent.

4. Obtain a child by performing *crossover* and *mutation* operations on the parent chromosomes. The crossover operation is performed by selecting a crossover point $1 \leq m \leq N$ randomly such that the first m genes of the child chromosome are inherited from the first parent while the remaining $N - m$ genes are inherited from the second parent. The *mutation* operation is performed by randomly exchanging two genes in the children. After generating the child chromosome, its fitness value is evaluated using (3.28).

5. Replace the chromosome with the lowest raw fitness value in the population by the child chromosome.

6. Repeat steps 3 to 5 until $\bar{M}$ offspring have been obtained without improving the best solution found so far or a maximum number of offspring $\bar{\bar{M}}$ ($\bar{\bar{M}} > \bar{M}$) has been obtained.

7. Return $\hat{\boldsymbol{\mu}}^*$, which is the chromosome with the highest $\mathcal{X}^{(k)}$.

3.3.3.2 UAV's Itinerary Optimization

For a given $\hat{\boldsymbol{\mu}}^* = [\hat{\mu}_i^*]_{N\times 1}$ as a solution of (3.25), the optimization problem (3.22) can be written as

$$\mathbf{P3} \min_{\hat{\boldsymbol{\eta}}, \bar{\boldsymbol{\psi}}} \ \mathcal{E}\left(\hat{\boldsymbol{\eta}}, \bar{\boldsymbol{\psi}}\right), \tag{3.29a}$$

$$\text{s.t.} \quad \sum_{j=1}^{N} \hat{\eta}_{ij} = \hat{\mu}_i^*, \ \forall\, 1 \leq i \leq N, \tag{3.29b}$$

$$\left[\sum_{i=1}^{N} \hat{\eta}_{ij}\right]^{+} \left(h_{\min} \leq \tilde{z}_j \leq h_{\max}\right), \ \forall\, 1 \leq j \leq N, \tag{3.29c}$$

$$\hat{\eta}_{ij} \in \{0, 1\}, \ \forall\, 1 \leq i, j \leq N. \tag{3.29d}$$

To solve (3.29), the optimal number of aggregation points K should be determined, the location of each aggregation point should be optimally determined, the optimum set of devices with $\hat{\mu}_i^* = 1$ to contact the UAV, and finally, the optimum route that the UAV needs to follow to visit the K aggregation points. Algorithm 2 is designed to find the solution of (3.29), in which the main steps are implemented as follows. First, the algorithm obtains the optimum set of active devices by solving (3.25) using the GA. The algorithm iterates and varies the number of aggregation points from 1 to N^*, where $N^* = \sum_{i=1}^{N} \hat{\mu}_i^*$ is the number of active devices. At each iteration, the algorithm performs K-means clustering such that each active device is assigned to the UAV at the closest aggregation point. The K-means clustering converges and obtains K centroids, with each centroid representing $\tilde{x}_j$ and $\tilde{y}_j$ coordinates of the data aggregation point $\tilde{\psi}_j$.[3] The optimum altitude of the aggregation point $\tilde{z}_j$ depends on the distance between its centroid and the farthest active device that transmits the data to the UAV at $\tilde{\psi}_j$, and can be expressed as $\tilde{z}_j = \hat{r}_{ij} \tan(\theta_{\text{opt}})$, where $\hat{r}_{ij} = \max_{1 \leq i \leq N} \{\hat{\eta}_{ij} r_{ij}\}$, with $r_{ij} = \sqrt{(x_i - \tilde{x}_j)^2 + (y_i - \tilde{y}_j)^2}$ and $\theta_{\text{opt}} = 75.52°, 54.62°, 42.44°$, and

[3]The instructions in lines 9 to 14 of Algorithm 1 iterate only a few iterations before clusters converge and the condition in line 9 is dissatisfied.

20.34° for the high-rise urban, dense urban, urban, and suburban environments, respectively [23]. Consequently, the altitude of each aggregation point is obtained as follows:

$$\tilde{z}_j = \begin{cases} h_{\max}, & \text{if } \hat{r}_{ij} \tan(\theta_{\text{opt}}) > h_{\max}, \\ h_{\min}, & \text{if } \hat{r}_{ij} \tan(\theta_{\text{opt}}) < h_{\min}, \\ \hat{r}_{ij} \tan(\theta_{\text{opt}}), & \text{otherwise.} \end{cases} \tag{3.30}$$

Once the K aggregation points are obtained, the following question raises: *what is the shortest possible route that starts at ψ_0 and visits each aggregation point exactly once and returns to ψ_0?* This question has the same structure as a well known problem in the combinatorial optimization called traveling salesman problem (TSP). Here, the nearest neighbor (NN) scheme is adopted which solves the TSP efficiently. In the adopted NN scheme, the UAV's tour starts at ψ_0, and repeatedly chooses the next aggregation point as the unvisited aggregation point closest to the current one. Once all aggregation points have been chosen, it closes the tour by returning to ψ_0. In other words, for a given set of K aggregation points, the data aggregation tour $\bar{\boldsymbol{\psi}}\left(\hat{\boldsymbol{\eta}}, \tilde{\boldsymbol{\psi}}\right)$ is obtained as

$$\bar{\boldsymbol{\psi}}\left(\hat{\boldsymbol{\eta}}, \boldsymbol{\psi}\right) = \{\psi_0, \tilde{\psi}_{\hat{j}_1}, \tilde{\psi}_{\hat{j}_2}, \ldots, \tilde{\psi}_{\hat{j}_K}, \psi_0\}, \tag{3.31}$$

where $\tilde{\psi}_{\hat{j}_1}$ is the closest point to ψ_0, $\tilde{\psi}_{\hat{j}_2}$ is the closest point to $\tilde{\psi}_{\hat{j}_1}$, and so on. The algorithm examines all the possible values of K and returns $\hat{\boldsymbol{\eta}}^*$ and $\tilde{\boldsymbol{\psi}}^*$ as a solution for (3.22).

Solving (3.25) using the proposed GA requires $\mathcal{O}(N[\bar{\bar{M}}+\Xi])$ operations. Consequently, the computational complexity of solving (3.22) using the proposed greedy algorithm is $\mathcal{O}(N[\bar{\bar{M}} + \Xi] + N^3)$. It is worth mentioning that this is reduced when compared with the search space of (3.22), which is $2^{N^2} N!$.

Algorithm 2 A greedy algorithm for data aggregation using a UAV.

Input: I, ψ_0, $\boldsymbol{\psi}$, N;

Obtain $\hat{\boldsymbol{\mu}}^*$ by solving (3.25); set $N^* \leftarrow \sum_{i=1}^{N} \hat{\mu}_i^*$;

$\mathcal{E}^* \leftarrow \infty$;

for $K = 1$ to N^* **do**

$\hat{\boldsymbol{\eta}} \leftarrow [0]_{N \times N}$;

Selected a device with $\hat{\mu}_i^* = 1 \; \forall i = 1, 2, \ldots N^*$;

$\eta_{i\hat{j}} \leftarrow 1$ if $\hat{\mu}_i^* = 1$ with $\hat{j} = \arg \min_{1 \le j \le K} \{r_{ij}\}$;

$\{\dot{x}_j, \dot{y}_j\} = \{x_0, y_0\} \; \forall j = 1, 2, \ldots K$;

While $\exists \sqrt{\left(\dot{x}_j - \tilde{x}_j\right)^2 + \left(\dot{y}_j - \tilde{y}_j\right)^2} > \delta \; \forall j = 1, 2, \ldots K$ **do**

$\{\dot{x}_j, \dot{y}_j\} \leftarrow \{x_j, y_j\}$;

$\tilde{x}_j = \frac{\sum_{i=1}^{N} \hat{\eta}_{ij} x_i}{\sum_{i=1}^{N} \hat{\eta}_{ij}}, \;\; \tilde{y}_j = \frac{\sum_{i=1}^{N} \hat{\eta}_{ij} y_i}{\sum_{i=1}^{N} \hat{\eta}_{ij}} \;\; \forall j = 1, 2, \ldots K$;

$\hat{\boldsymbol{\eta}} \leftarrow [0]_{N \times N}$;

$\hat{\eta}_{i\hat{j}} \leftarrow 1$ if $\hat{\mu}_i^* = 1$ with $\hat{j} = \arg \min_{1 \le j \le K} \{r_{ij}\}$;

end While

Obtain $\tilde{z}_j$ using (3.30) $\forall j = 1, 2, \ldots K$;

$\tilde{\psi}_j = \{\tilde{x}_j, \tilde{y}_j, \tilde{z}_j\} \; \forall j = 1, 2, \ldots K$;

Obtain $\bar{\boldsymbol{\psi}} \left(\hat{\boldsymbol{\eta}}, \tilde{\boldsymbol{\psi}}\right)$ using (3.31);

Evaluate $\mathcal{E}\left(\hat{\boldsymbol{\eta}}, \bar{\boldsymbol{\psi}}\right)$ using (3.21);

if $\mathcal{E}^* > \mathcal{E}\left(\hat{\boldsymbol{\eta}}, \bar{\boldsymbol{\psi}}\right)$

$\mathcal{E}^* \leftarrow \mathcal{E}\left(\hat{\boldsymbol{\eta}}, \bar{\boldsymbol{\psi}}\right)$; $\hat{\boldsymbol{\eta}}^* \leftarrow \hat{\boldsymbol{\eta}}$; $\bar{\boldsymbol{\psi}}^* \leftarrow \bar{\boldsymbol{\psi}}$;

end if

end for

Return $\mathcal{E}^*$, $\hat{\boldsymbol{\eta}}^*$, and $\bar{\boldsymbol{\psi}}^*$.

3.3.4 Simulation Results

The energy expenditure of the proposed algorithm is compared with the following baseline approaches: (1) Baseline approach 1 in which all the available devices are active and the UAV stops above each device to aggregate data; (2) Baseline approach 2 in which the optimum set of devices are active and the UAV stops above each active device to aggregate data; (3) Baseline approach 3 in which all the available devices are active and the UAV stops at the optimum aggregation points. In obtaining these results, the N devices are placed at random in an area-of-interest of dimensions 1×1 km^2. The default simulation parameters summarized in Table 3.2 [20, 21] are considered in the results, unless otherwise stated.

Table 3.2: Simulation parameters of the data aggregation using a UAV framework.

Parameter	**Value**	**Parameter**	**Value**	**Parameter**	**Value**
P_T	21 dBm	B	20 kHz	f_c	2 GHz
L	2048 bytes	N_0	-174 dBm/Hz	ϵ	2
α	10	β	0.03	P_r	0.0126 W
ζ_{NLoS}	20 dB	ζ_{LoS}	0 dB	P_{stop}	0 W
P_{max}	6 W	v	15 m/s	v_{max}	15 m/s
M	0.5 kg	r	20 cm	p	4
Ξ	$10N$	$\bar{M}$	100	$\bar{\bar{M}}$	10^4

Figure 3.4 shows the energy expenditure of the proposed algorithm and the baseline approaches for different numbers of available devices N. It is noticed that the energy expenditure of the proposed algorithm is less and flattens as N increases. This is due to the fact that as N increases, the correlation among the data of the devices increases; however, baseline approaches 1 and 3 aggregate this correlated data and baseline approach 3 has a

longer UAV tour in comparison with the proposed algorithm.

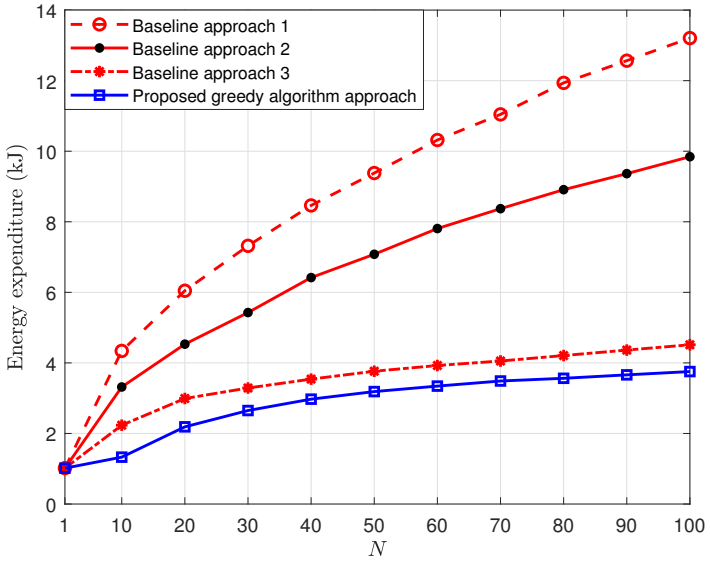

Figure 3.4: Energy expenditure versus the number of available devices N.

Figure 3.5 illustrates the performance of the baseline approaches and proposed algorithm versus the ratio $\Gamma = \frac{I}{H}$. It is seen that for small values of Γ, the proposed algorithm preserves more energy. As Γ (i.e., I) increases, the energy expenditure of the proposed algorithm increases; however, it is upper-bounded by the energy expenditure of baseline approach 3. This is not the case for baseline approach 2, whose energy expenditure increases linearly with Γ. The energy expenditure of baseline approaches 1 and 3 is not a function of Γ.

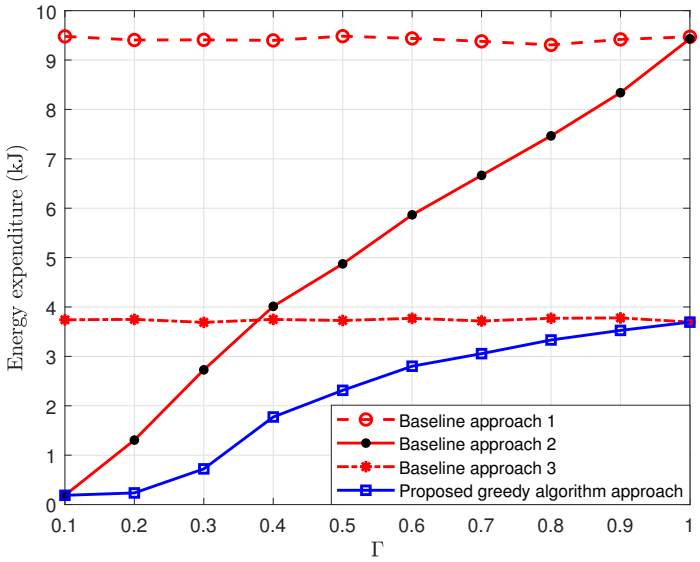

Figure 3.5: Energy expenditure versus the ratio $\Gamma = \frac{I}{H}$ with $N = 50$ and $\rho_s = 10^3$.

3.4 Concluding Remarks

In this chapter, two energy-efficient data aggregation frameworks have been proposed. The first framework is a device-role assignment framework, in which an optimization problem has been formulated with the objective of maximizing the gathered information with minimal energy by activating a subset of the devices and assigning a role to each active device. An ACO solution approach has been developed to solve the optimization problem. Simulation results have illustrated that the proposed framework gathers more information with less energy when compared with a baseline approach, under different scenarios and the ACO algorithm achieves near-optimal performance.

The second framework considered spatially-correlated data aggregation from IoT devices using a UAV. In this framework, an optimization problem has been formulated to minimize the energy expenditure of the UAV and devices and to guarantee that an ade-

quate amount of uncorrelated data is aggregated. A greedy algorithm has been developed to optimize the contributing devices and the itinerary followed by the UAV in order to ensure energy-efficient data aggregation. Simulation results have revealed that the proposed algorithm reduced the energy expenditure when compared with three baseline approaches.

Chapter 4

Age-Aware Data Gathering

4.1 Background and Motivation

A wide range of applications (such as sensor networking, IoT, intelligent transportation lines monitoring, and environment monitoring) requires a timely information about the state of processes of interest. In order to have efficient decision making and control commands in these applications, the data gathering centers need to maintain freshness of the measurements of different devices. Accordingly, the key enabler for such applications is the freshness of the information at the data gathering centers [2].

Traditional metrics such as delay and throughput can not fully characterize the information freshness. A performance metric, namely AoI, was introduced to characterize the freshness of the information from the destination device perspective. AoI is defined as the time elapsed since the most recently received update information at the destination was generated at the source device [5]. It is worth mentioning that under hardware limitations, minimizing AoI is not guaranteed by maximizing other parameters such as the through-

put or the contention window size [46]. Some techniques, such as power-domain non-orthogonal multiple access, can be utilized to minimize the AoI and balance the consumed power of the devices [47].

Wireless sensor networks (WSNs) and IoT constitute rapidly growing surveillance and monitoring technologies, in which a set of tiny yet intelligent devices that monitor processes of interest, transmit and forward data to a designated sink or data gathering center. A fundamental design issues of WSN and IoT is the placement topology where the nodes are located to effectively handle the application requirements and operations. In this context, the linear network typology represents an interesting sub-class of WSNs and IoT typologies, in which the devices are placed in a one-dimensional linear topology. Linear networks play a crucial role in many monitoring occasions such as city drainage pipelines monitoring, highways and railways security, power lines inspection, international border surveillance, and oil and gas pipelines monitoring systems. The linear network topology imposes additional communication challenges, such as the fact that the devices' communication range cannot cover the entire linear network, and due to the distribution of the devices along the network, the intuitive transmission mode is the multi-hop mode. In a multi-hop transmission mode, the source device transmits the updates to the data gathering centre via a sequence of devices [48]. This transmission mode leads to imbalance energy consumption and high latency [14, 15]. Consequently, maintaining information freshness in linear networks is a challenging issue which becomes more severe in communication environment with relatively high transmission delay such as underwater communication environment.

Underwater linear structures, such as underwater pipelines, have played a crucial role to enable offshore exploration industry by providing a sustainable transportation of essen-

tial resources such as oil, natural gas, and water. Underwater pipelines normally laid on the seabed and in such harsh environments require regular inspection and monitoring. The decision maker should be aware of the status of the pipeline and some physical processes around it, such as temperature and sea currents vibrations, to guarantee the pipeline's continuing and fluent integrity. Traditional data gathering methods in underwater linear networks such as divers or towfish need a highly trained personnel, and as the length of the network extends for long distance, these methods become expensive and time consuming.

AUVs are utilized as a cost and time efficient method to collect data from the devices in underwater linear networks. In this case, the AUV represents a mobile data collection center that gathers the data, and if needed, reports any incidents to a surface center. Moreover, AUVs are embedded with large storage and computing capabilities, which enables the AUVs to play the role of organizing and managing the underwater networks; this minimizes the human interaction in the monitoring operation. Optimizing the AUV's traveling path and the locations at which it communicates with the devices improves the network throughput, balances the energy consumption, and enables the AUV to maintain freshness of the information about the monitored processes.

4.1.1 Related Work

Mobile data gathering centers are an emerging technology that can be harnessed for underwater [49–54] and terrestrial [55–60] networks. In [49], a deep reinforcement learning framework for AUV motion planning was proposed. In this framework, the sensors information is utilized to enable the AUV to reach multiple target points in a sequence while avoiding obstacles. A cooperative trajectory estimation algorithm for a fleet of AUVs which

minimizes the mission time by accurately localizing the payload data was proposed in [50]. A multi-modal data gathering from a set of sensors by employing a mobile AUV was proposed in [51]. The authors studied a multi-modal hybrid transmission using both acoustic and optical communications where the appropriate mode is chosen according to the quality of transmissions that take place over time. A task-based underwater pipeline inspection system, in which an AUV utilizes a set of underwater sensors to localize and inspect an underwater pipeline was presented in [52]. In [53], an AUV path planning algorithm was proposed to enable data gathering with maximum value of the received information. The problem was formulated as a integer linear programming optimization problem, and the proposed algorithm was compared with three baseline solutions. In [54], a linear underwater sensor network was deployed to monitor underwater pipelines and upload the data to an AUV. The authors studied different AUV movement strategies to improve the average data packet delay and delivery ratio. To the best of the authors' knowledge, the information freshness has not been studied in underwater networks scenario.

In [55], a UAV was considered as a mobile data gathering center in a terrestrial linear network. The authors' objective was to optimize the UAV speed to minimize the total flight time while gathering the data of a linearly placed devices. In [56], a UAV-assisted data gathering network was considered, in which the UAV hovers above ground devices to collect status updates about their observed processes. The objective was minimizing the normalized weighted sum of AoI by jointly optimizing the UAV's path and scheduling the status updates. In [57], a UAV was considered as a mobile relay to minimize the average peak AoI for a source–destination pair. Only one source-destination pair was considered, in which the source device sends its measurements to the destination device via the UAV. The problem was formulated to jointly optimize the UAV's flight trajectory in a finite set

of points and energy allocations for data transmissions. In [58–60], deep reinforcement learning (DRL) networks with different settings were proposed to find an optimal AUV trajectory and/or scheduling policy in order to minimize the AoI of ground devices. However, the authors of these works considered that both the time instants and UAV trajectory are discretized. Moreover, in such discretized scenario, the AoI is described by a discrete function which increases by one at each discrete time instate if no update is received from the corresponding device. This discretization introduces approximation errors to the trajectory planning as well as to obtained solutions and limits their implementation in real-world scenarios.

This chapter introduces a framework to minimize the normalized weighted sum AoI of a set of physical processes at a mobile data gathering center that gathers data from a set of linearly-deployed devices. Each device senses one or more processes. A novel metric referred to as correlation-aware AoI (CAAoI) is also introduced to capture both freshness and diversity in the gathered information. The CAAoI of an information gathering system is studied, in which a UAV gathers information about a set of physical processes; each process can be measured by one or more ground devices.

4.2 Age-Optimal Information Gathering in Linear Networks

4.2.1 System Model

In this work, data gathering in a linear network scenario is considered, in which a mobile data gathering center is given a mission of a finite horizon of time T seconds to monitor a

set $\mathcal{P} = \{p_k\}_{k=1}^{P}$ of P physical processes by gathering information from a set $\mathcal{N} = \{n_i\}_{i=1}^{N}$ of N devices. The devices are located on a line, which is represented by $\hat{\boldsymbol{\psi}} = \{\hat{\psi}_i\}_{i=1}^{N}$ with $\hat{\psi}_i = \{\hat{x}_i, 0, \hat{d}\}$ as the 3D geographical coordinates of device n_i. Each device is able to sense one or more processes. To represent the devices ability to observe the processes, $\boldsymbol{\chi} = [\chi_{ik}]_{N\times P}$ is defined such that

$$\chi_{ik} = \begin{cases} 1, & \text{if } n_i \text{ is able to observe } p_k, \\ 0, & \text{otherwise.} \end{cases} \tag{4.1}$$

The mobile data gathering center needs to maintain freshness of its information status about the processes $\mathcal{P}$ as it moves along a linear path that starts at initial point ψ_0 and passes through a set of M data gathering points $\boldsymbol{\psi} = \{\psi_j\}_{j=1}^{M}$, with $\psi_j = \{x_j, 0, d\}$. The decision variable $\boldsymbol{\mathcal{T}} = \{\mathcal{T}_j\}_{j=1}^{M}$ is defined such that $\mathcal{T}_j$ is the time allocated for the mobile data gathering center to stop at ψ_j. For a given time allocation decision $\boldsymbol{\mathcal{T}}$, the device-center association decision variable $\boldsymbol{\eta} = \left[\eta_{ijk}\right]_{N\times M\times P}$ is defined such that $\eta_{ijk} = 1$ if n_i sends data of process p_k to the mobile data gathering center at data gathering ψ_j, and $\eta_{ijk} = 0$ otherwise. For a given decision variable $\boldsymbol{\eta}$, the devices' transmission ordering decision $\boldsymbol{\mu} = \left[\mu_q\right]_{Q\times 3}$ with $Q = \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{P}\eta_{ijk}$ is defined such that each row contains 3 indices representing the index of gathering point, device, and process; the transmission ordering follows the order of the rows in $\boldsymbol{\mu}$.

The traveling time of the mobile data gathering center between data gathering points is represented by the decision variable $\bar{\boldsymbol{\mathcal{T}}} = \{\bar{\mathcal{T}}_j\}_{j=1}^{M}$, with $\bar{\mathcal{T}}_j$ as the traveling time from ψ_{j-1} to ψ_j. Consequently, the location of data gathering points $\boldsymbol{\psi}$ can be inferred from the decision variable $\bar{\boldsymbol{\mathcal{T}}}$.

4.2.2 AUV Kinematics Model

Since the data gathering center travels in a straight path, its kinematic equations reduce to the kinematic equations for linear motion. The maximum straight line velocity of the data gathering center is $V_{\max}$, and its constant acceleration and deceleration are $\delta > 0$ and $\delta' < 0$, respectively. Consequently, the data gathering center accelerates from constancy to $V_{\max}$ in $t_0 = V_{\max}/\delta$ (seconds) and over the distance $d_0 = V_{\max}^2/2\delta$ (m). Similarly, it decelerates from $V_{\max}$ to constancy in $t_0' = V_{\max}/-\delta'$ (seconds) and over the distance $d_0' = V_{\max}^2/-2\delta'$ (m).

For a given $\bar{\mathcal{T}}_j$ as the allocated time to the data gathering center to travel from point ψ_{j-1} to ψ_j in the observation line, if $\bar{\mathcal{T}}_j \geq t_0 + t_0'$, then the data gathering center accelerates until it reaches $V_{\max}$, moves with $V_{\max}$ for $\bar{\mathcal{T}}_j - t_0 - t_0'$ seconds, and decelerates until constancy. If $\bar{\mathcal{T}}_j < t_0 + t_0'$, then the data gathering center accelerates in the first portion of $\bar{\mathcal{T}}_j$ and decelerates in the second portion. Consequently, the location of the data gathering point ψ_j can be obtained as $\psi_j = \{x_{j-1} + \Delta_j, 0, d\}$, where Δ_j is the traveling distance of the data gathering center in $\bar{\mathcal{T}}_j$ seconds and can be expressed as

$$\Delta_j = \begin{cases} \frac{V_{\max}^2}{2\delta} + V_{\max}(\bar{\mathcal{T}}_j - t_0 - t_0') + \frac{V_{\max}^2}{-2\delta'}, \text{ if } \bar{\mathcal{T}}_j \geq t_0 + t_0', \\ \\ \frac{1}{2}\delta\left(\frac{-\delta'\bar{\mathcal{T}}_j}{-\delta'+\delta}\right)^2 + \frac{1}{2}(-\delta')\left(\frac{\delta\bar{\mathcal{T}}_j}{-\delta'+\delta}\right)^2, \quad \text{if } \bar{\mathcal{T}}_j \leq t_0 + t_0'. \end{cases} \tag{4.2}$$

It is worth mentioning that the location of the initial point ψ_0 is known and for a given $\bar{\mathcal{T}}_j$, the above described iterative procedure is able to determine the locations of the data gathering points $\boldsymbol{\psi}$.

4.2.3 Problem Formulation

The objective is to maintain freshness of processes' information at the mobile data gathering center. The objective function is defined as a normalized weighted sum of time-average AoI as follows

$$O\left(\boldsymbol{\mathcal{T}}, \bar{\boldsymbol{\mathcal{T}}}, \boldsymbol{\eta}, \boldsymbol{\mu}\right) = \frac{1}{0.5T} \sum_{k=1}^{P} \lambda_k \mathbb{E}_T\big[\Delta_k(t)\big] \,, \tag{4.3}$$

where $\mathbb{E}_T\big[\Delta_k(t)\big]$ is the time-average AoI and can be obtain using (2.5) and λ_k is the importance weight of process p_k, such that $\sum_{k=1}^{P} \lambda_k = 1$. Hence, the optimization problem is formulated as

$$\mathbf{P1} \min_{\boldsymbol{\mathcal{T}}, \bar{\boldsymbol{\mathcal{T}}}, \boldsymbol{\eta}, \boldsymbol{\mu}} \; O\left(\boldsymbol{\mathcal{T}}, \bar{\boldsymbol{\mathcal{T}}}, \boldsymbol{\eta}, \boldsymbol{\mu}\right), \tag{4.4a}$$

$$\text{s.t.} \quad \mathcal{T}_j \geq \sum_{i=1}^{N} \sum_{k=1}^{P} \eta_{ijk} L\left(I_k, \ell_{ij}\right), \forall\, j \leq M, \tag{4.4b}$$

$$\sum_{j=1}^{M} \mathcal{T}_j + \bar{\mathcal{T}}_j \leq T, \tag{4.4c}$$

$$\eta_{ijk} = \chi_{ik} \eta_{ijk}, \forall\, i \leq N, j \leq M, k \leq P, \tag{4.4d}$$

$$\gamma(\ell_{ij}) \geq \left[\eta_{ijk}\right]^{+} \gamma_{\text{th}}, \forall\, i \leq N, j \leq M, k \leq P, \tag{4.4e}$$

$$\mathcal{T}_j, \bar{\mathcal{T}}_j \geq 0, \eta_{ijk} \in \{0, 1\}, \forall\, i \leq N, j \leq M, k \leq P. \tag{4.4f}$$

Constraint (4.4b) guarantees that the assigned time for the AUV at each gathering point is enough to gather the data scheduled at that point. Constraint (4.4c) guarantees that the entire mission is performed within the time constraint. Constraint (4.4d) guarantees that each device is able to sense the assigned processes. Constraint (4.4e) guarantees that the SNR at the AUV is greater than its sensitivity γ_{th}, where $[\nu]^{+} \triangleq \min\{\nu, 1\}$. It is worth noting that the optimization problem (4.4) is a non-convex mixed integer problem that constraints over coupled variables, and thus, it cannot be solved by the standard optimization techniques.

This motivates the proposed DRL based solution introduced in the next section.

Implementation Issues: The proposed framework can be applied in the terrestrial linear networks as well, e.g., the system model in [55], in which an UAV gathers data from a set of ground devices deployed in a linear model. In such a case, the kinematics model of the UAV can be deduced based on its acceleration-deceleration and maximum speed. The latency of transmitting the information from a device to the UAV at a given point can be obtained using a probabilistic ground-to-air communication channel model.

4.2.4 Proposed Solution Approach

The first part of this section aims to give a brief overview of, and key notation for, RL, DRL, and actor-critic DRL. The second part presents the proposed DRL-based solution method.

4.2.5 Theoretical Preliminaries

RL refers to the process of learning of an agent by interacting with an environment in discrete decision epochs. At each learning epoch l, the agent executes an action $\boldsymbol{a}_l \in \mathcal{A}$, observes state $\boldsymbol{s}_l \in \mathcal{S}$, receives a reward r_l, and transits to the next state $\boldsymbol{s}_{l+1} \in \mathcal{S}$, where $\mathcal{A}$ is the set of all possible actions and $\mathcal{S}$ is the set of all possible states. The agent's actions are governed by a policy $\pi(\boldsymbol{a} \mid \boldsymbol{s})$, which represents the probability of taking action $\boldsymbol{a}$ at state $\boldsymbol{s}$. The goal of the learning process is to improve the policy to maximize the discounted reward over the learning time horizon. A key metric of RL is the Q-function $Q_\pi(\boldsymbol{s}, \boldsymbol{a})$ which is the expected return by taking action $\boldsymbol{a}$ at state $\boldsymbol{s}$ by following policy π. The learning process based on Q-function is known as the Q-learning, in which the optimum Q-function can be

obtained by solving the Bellman equation using iterative solution approaches. It has been shown that Q-learning converges to the optimal solution if each state-action pair is visited in sufficient learning iterations [61]. The Q-learning method, also known as table-based learning, requires a table to store each state-action pair and the Q value of the pairing, such that the Q value is updated only when that state-action pair is actually experienced during the training. Consequently, this method becomes impractical in two common scenarios: (1) The number of states/actions is very large; (2) The state space and/or action space are/is continuous.

In DRL, the action-value function is approximated by deep neural network (DNN) such that the Q-function becomes $Q(\boldsymbol{s}, \boldsymbol{a} \mid \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is the weight vector of the DNN [62]. DRL has three main advantages: (1) There is no need to evaluate each state-action pair and store all these pairs along with the updated; (2) A pioneer idea of using double deep Q-learning namely actor-critic method is able to deal with system environments with continuous state/action spaces [18]. In the actor-critic method, the actor network $\pi(\boldsymbol{s}_l \mid \boldsymbol{\theta^{\pi}})$ is learned by updating the weight $\boldsymbol{\theta^{\pi}}$ to obtain the best action at a given state $\boldsymbol{s}_l$; while the critic network $Q(\boldsymbol{s}_l, \boldsymbol{a}_l \mid \boldsymbol{\theta^{Q}})$ is an ordinary DRL network that approximates the Q-value of a state-action pair. The critic network is trained by updating its weight $\boldsymbol{\theta^{Q}}$ to minimize the following loss function [18]:

$$L(\boldsymbol{\theta^{Q}}) = \mathbb{E}\left[\Big(y_l - Q(\boldsymbol{s}_l, \boldsymbol{a}_l \mid \boldsymbol{\theta^{Q}})\Big)^2\right], \tag{4.5}$$

where y_l is a target value that can be obtained as follows

$$y_l = r(\boldsymbol{s}_l, \boldsymbol{a}_l) + \phi Q\Big(\boldsymbol{s}_{l+1}, \pi\big(\boldsymbol{s}_{l+1} \mid \boldsymbol{\theta^{\pi}}\big) \mid \boldsymbol{\theta^{Q}}\Big), \tag{4.6}$$

with $0 \leq \phi \leq 1$ as a discount factor [18]. The actor network is updated using a deep

deterministic policy gradient (DDPG) [18] by maximizing the following function:

$$J(\boldsymbol{\theta^{\pi}}) = \mathbb{E}\left[Q(\boldsymbol{s}, \boldsymbol{a} \mid \boldsymbol{\theta^{Q}})\big|_{\boldsymbol{s}=\boldsymbol{s}_l, a=\pi(\boldsymbol{s}_l|\boldsymbol{\theta^{\pi}})}\right], \tag{4.7}$$

such that the weight $\boldsymbol{\theta^{\pi}}$ is updated using the gradient of (4.7):

$$\nabla J(\boldsymbol{\theta^{\pi}}) = \nabla_{\boldsymbol{a}} Q(\boldsymbol{s}, \boldsymbol{a} \mid \boldsymbol{\theta^{Q}})\big|_{\boldsymbol{s}=\boldsymbol{s}_l, a=\pi(\boldsymbol{s}_l)} \cdot \nabla_{\boldsymbol{\theta^{\pi}}} \pi(\boldsymbol{s} \mid \boldsymbol{\theta^{\pi}})\big|_{\boldsymbol{s}=\boldsymbol{s}_l}. \tag{4.8}$$

Finally, it has been noted that the direct application of the above descried actor-critic training procedure directly leads to oscillations and uneven divergence. A simple yet effective method to stabilize the actor-critic training procedure was proposed in [62]. This learning stabilization method consists of two techniques: (1) Experience replay buffer **B**, which is replay memory that stores transition experiences $(\boldsymbol{s}_l, \boldsymbol{a}_l, r_l, \boldsymbol{s}_{l+1})$ such that the training is performed based on a mini-batch of randomly-selected samples from the replay memory to minimize correlations between the training samples; (2) Separate target networks, which are copies of the actor and critic networks; the weight of these target networks can be updated using the Polyak averaging method as follows

$$\boldsymbol{\theta^{\pi'}} = (1-\sigma)\boldsymbol{\theta^{\pi'}} + \sigma\boldsymbol{\theta^{\pi}};\;\; \boldsymbol{\theta^{Q'}} = (1-\sigma)\boldsymbol{\theta^{Q'}} + \sigma\boldsymbol{\theta^{Q}}, \tag{4.9}$$

with $1 \leq \sigma \leq 0$. This makes the training of the target networks stable and robust, and hence, improves the convergence.

4.2.6 Proposed DRL-Based Solution Method

In the proposed DRL solution method, an agent is designed to find the optimum travelling and stopping time allocation decisions $\bar{\boldsymbol{\mathcal{T}}}$ and $\boldsymbol{\mathcal{T}}$, respectively. Based on $\bar{\boldsymbol{\mathcal{T}}}$, the locations of the stopping points can be obtained using (4.2). Consequently, devices that are able to

communicate with the AUV at each stopping point can be determined by testing constraint (5.7d). Based on $\boldsymbol{\mathcal{T}}$, the device-process can be scheduled to minimize the objective function in (4.3). For a given $\hat{\boldsymbol{\psi}}$, $\boldsymbol{\psi}$, and γ_{th}, the device point communication matrix $\boldsymbol{C} = \left[c_{ij}\right]_{N\times M}$ can be obtained such that

$$c_{ij} = \begin{cases} 1, & \text{if } \gamma(\ell_{ij}) \geq \gamma_{\text{th}}, \\ 0, & \text{otherwise.} \end{cases} \tag{4.10}$$

In the proposed DRL, the state, action, and environment are defined as follows

- State: The state at the l-th training step $\boldsymbol{s}_l = [s_l]_{1\times 2M}$ consists of two parts: The first part represents the current locations of the AUV stopping points, referred to as "the data gathering points". Since all the points are located on a line, the first part of the state can be represented as $\boldsymbol{s}_l = \{x_1, x_2, \ldots, x_M\}$, where x_j is the x-coordinate of data gathering ψ_j. The second part of the state represents the number of devices that can communicate with the AUV at each stopping point. Consequently, the $(M+j)$-th element of the state equals $\sum_{i=1}^{N} C_{ij}$.

- Action: The action of the agent is defined as $\boldsymbol{a}_l = \{a_1, a_2, a_3, a_4, \ldots, a_{2M}\}$, where a_1 represents the portion of the total time T, which is allocated for the AUV to move from the initial point ψ_0 to ψ_1. Then a_2 represents the portion of the total time T, which is allocated to the AUV to stop at ψ_1. Similarly, a_3 represents the portion of the total time T, which is allocated to the AUV to move from point ψ_1 to ψ_2; while a_4 represents the portion of the total time T, which is allocated to the AUV to stop at ψ_2 and so on.

- Environment: Our environment is descried in Algorithm 3.

The training steps of the proposed DRL algorithm are described in Algorithm 4, which starts by initializing all neural networks as well as the replay buffer **B** (lines 1-1). The training iterates over E episodes; in each episode, the environment is initialize by distributing the devices over the observation line.

Algorithm 3 System environment at the l-th learning iteration.

1: Receive action $\boldsymbol{a}_l$;
2: Obtain $\boldsymbol{\mathcal{T}}$,$\bar{\boldsymbol{\mathcal{T}}}$:$\mathcal{T}_j = \frac{{a_2}_j{}^T}{\sum_{l=1}^{2M} a_l}$, $\bar{\mathcal{T}}_j = \frac{{a_{2(j-1)+1}}^T}{\sum_{l=1}^{2M} a_l}$, $\forall 1 \leq j \leq M$;
3: Obtain data gathering points $\boldsymbol{\psi}$ using $\bar{\boldsymbol{\mathcal{T}}}$ and (4.2);
4: Obtain device-point communication matrix $\boldsymbol{C}$ using (4.10);
5: $\boldsymbol{\eta} = \mathbf{0}$ and $\boldsymbol{\mu} \leftarrow \emptyset$;
6: **For** $j = 1$ **to** M **do**
7: **If** $\sum_{i=1}^{N} \sum_{k=1}^{P} c_{ij}\chi_{ik}\lambda_k \neq 0$;
8: temp $= 0$
9: **While** temp $< \mathcal{T}_j$
10: $k^* = \max_{1 \leq k \leq P} \{c_{ij}\chi_{ik}\lambda_k \Gamma_k(t)\}, \forall n_i \in \mathcal{N}$;
11: $i^* = \max_{1 \leq i \leq N} \{\frac{c_{ij}\chi_{ik^*}}{L(I_{k^*}, \ell_{ij})}\}$;
12: **If** $L\left(I_{k^*}, \ell_{i^*j}\right) \leq (\mathcal{T}_j - \text{temp})$
13: $\eta_{i^*jk^*} = 1$;
14: Embed $[j, i^*, k^*]$ into $\boldsymbol{\mu}$;
15: temp = temp $+ L\left(I_{k^*}, \ell_{ij}\right)$;
16: **Else**
17: temp = $\mathcal{T}_j$;
18: **End If**
19: **End while**
20: **End If**
21: **End for**
22: Evaluate the reward $r_l = 1/O\left(\boldsymbol{\mathcal{T}}, \bar{\boldsymbol{\mathcal{T}}}, \boldsymbol{\eta}, \boldsymbol{\mu}\right)$;
23: $\boldsymbol{s}_{l+1} = [x_1, x_2, \ldots, x_M, \sum_{i=1}^{N} c_{i1}, \sum_{i=1}^{N} c_{i2}, \ldots, \sum_{i=1}^{N} c_{iM}]$;

For each episode, Algorithm 1 performs L learning iterations; in each iteration it obtains the action using the actor network (line 7), applies the action to the environment,

and receives the immediate reward and the next state using Algorithm 1 (line 8). After storing the transition tuple $(\boldsymbol{s}_l, \boldsymbol{a}_l, r_l, \boldsymbol{s}_{l+1})$ in the replay memory **B** (line 9), a randomly selected samples of $\hat{B}$ transition tuples are utilized to learn the critic and actor networks (lines 10-12). The target networks are updated using (4.9) (line 13).

Algorithm 4 DRL algorithm for AoI data gathering.

1: Initialize the weights of the actor θ^{π} and critic θ^{Q} networks;
2: Initialize target networks $\theta^{\pi'}$ and $\theta^{Q'}$;
3: Initialize replay buffer **B**;
4: **For** $episode = 1$ **to** E **do**
5: Initialize the environment and receive an initial state $\boldsymbol{s}_1$;
6: **For** $l = 1$ **to** L **do**
7: Select the time allocation action $\boldsymbol{a}_l = \pi(\boldsymbol{s}_l \mid \boldsymbol{\theta^{\pi}})$;
8: Execute action $\boldsymbol{a}_l$ using Algorithm 1 and receive the reward r_l and next state $\boldsymbol{s}_{l+1}$;
9: Store transition $(\boldsymbol{s}_l, \boldsymbol{a}_l, r_l, \boldsymbol{s}_{l+1})$ in **B**;
10: Sample a random mini-batch of $\hat{B}$ transitions from **B**;
11: Update the weight of the critic network by minimizing the loss $L(\boldsymbol{\theta^{Q}}) = \frac{1}{\hat{B}} \sum_{\hat{l}}^{\hat{B}} (y_{\hat{l}} - Q(\boldsymbol{s}_{\hat{l}}, \boldsymbol{a}_{\hat{l}} \mid \boldsymbol{\theta^{Q}}))^2$;
12: Update the weight of the actor network using the sampled policy gradient in (4.8);
13: Update the weights of target networks using (4.9);
14: **End for**
15: **End for**

4.2.7 Simulation Results

The proposed age-optimal information gathering framework is evaluated in an underwater linear network. In the considered underwater network, the mobile data gathering center is mounted on an AUV . The performance of the proposed solution approach is compared with the following approaches:

- *A baseline approach* in which the AUV stops for each device and gathers its data.

- *A K-means clustering approach*[1] in which the AUV stops at $M = K$ data gathering points and each device communicates with the AUV at the closest aggregation point. In this approach, the adopted value of M is obtained by varying K from 1 to N and selecting the value that minimizes the objective function.

The default simulation parameters summarized in Table 4.1 are considered in the results, unless otherwise stated.

Table 4.1: Simulation parameters of the information gathering in linear networks framework.

Parameter	Value	Parameter	Value
Pipeline length	2 km	ϖ	0.5 [24]
Pipeline depth	300 m	w	5 m/s [24]
AUV depth	100 m	χ/ξ	10 dB/-10 dB [27]
$V_{\max}$	$0.8\ m/s$ [63]	c	1500 m/s [24, 27]
δ/δ'	$0.2/-0.2\ m/s^2$ [63]	ζ	0.25 [27]
N	20 devices	γ_{th}	10 dB [24]
M	5 points	φ	1 bps/Hz [27]
T	30 mins	Discount factor ϕ	0.99
P	4 processes	Replay buffer size	10^4
I_k	2048 bits [27]	mini-batch	32
λ_k	$\frac{1}{P}$	E/L	10^4episodes/200 steps

Figure 4.1 shows the normalized weighted sum AoI of the proposed framework, baseline approach, and K-means clustering approach versus the pipeline length. It is clear that the performance of the baseline approach and K-means clustering approach follow a semi-convex behaviour. Such a behaviour is due to the fact that, for a constant observa-

[1]The K-means clustering algorithm converges to obtain K centroids, with each centroid representing the location of a data gathering in the travelling path of the AUV.

tion time and number of devices, the inter-device distance increases as the pipeline's length increases. For short pipeline lengths, the AUV collects the data from the close proximity devices and stays idle the remaining time. For long pipeline lengths, the AUV collects the data from the sparsely located devices and spends most of the observation time travelling to receive data from as many devices as possible. This is not the case for the proposed framework which provides less normalized weighted sum AoI over the considered range of the pipeline length. This is a result of optimizing both the location and stopping time of the data gathering points, which gives the proposed framework a robustness versus changing the pipeline length.

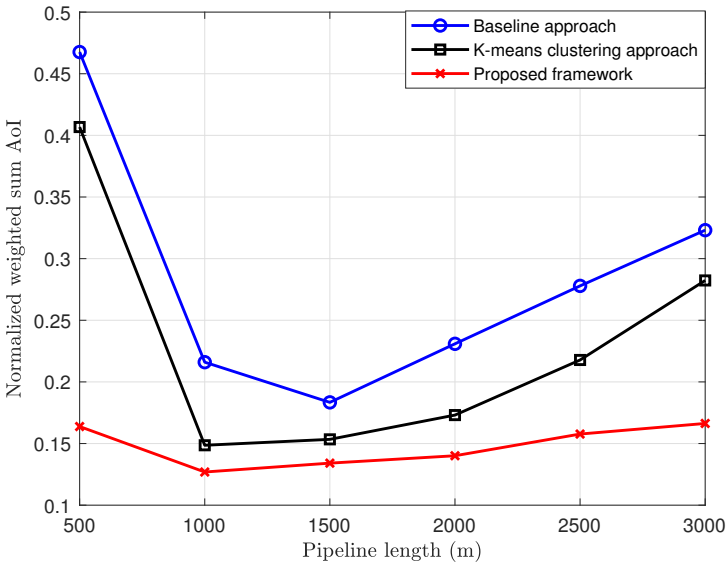

Figure 4.1: Normalized weighted sum AoI versus the pipeline length.

Figure 4.2 illustrates the performance of the three solution approaches versus the observation time T. It can be seen that the proposed framework outperforms the other two

approaches. As the observation time T changes, the proposed framework optimizes the travelling and stopping times which leads to reduced normalized weighted sum AoI. It is noticed that for a small observation time, the K-means clustering approach achieves relatively less normalized weighted sum AoI in comparison with the baseline approach. This can be attributed to the fact that the AUV in the baseline approach cannot receive adequate updates of all the processes. However, for large values of the observation time, the AUV collects the data from the devices and stays idle the remaining time in both the baseline and K-means clustering approaches.

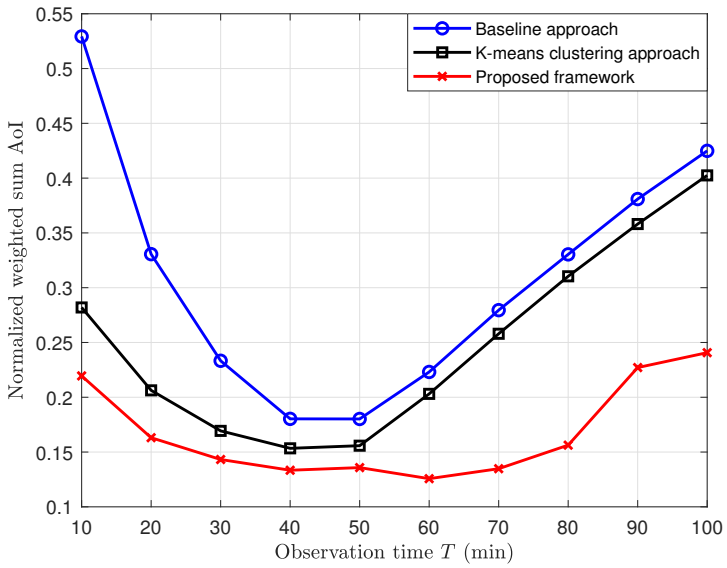

Figure 4.2: Normalized weighted sum AoI versus the observation time.

4.3 Age- and Correlation-Aware Information Gathering

Measurements from spatially proximal devices or successive measurements from the same device are correlated. the following question is addressed: *Can we usefully characterize both the freshness and diversity of the received information in information gathering systems?* To answer this question, a novel metric is introduced that not only captures the information freshness at the receiver, but also reflects the diversity in the gathered information.

To assess the freshness of the received updates at the destination, Kaul *et al.* [5] define the instantaneous AoI of the physical process p_k at time instant t as $\Delta_k(t) = t - u^k(t)$, where $\Delta_k(0) = 0$ and $u^k(t)$ is the time instant at which the last update about p_k was generated (also referred to as the "timestamp" of the last update). An update with a timestamp u_j^k reaches the destination at time instant τ_j^k, such that $\tau_j^k = u_j^k + \ell_j^k$, with ℓ_j^k as the latency of transmitting the j-th update from the source to the destination. Fig. 4.3 illustrates the AoI of p_k with a total number of updates $U^k(T) = 3$ over a time duration T. Note

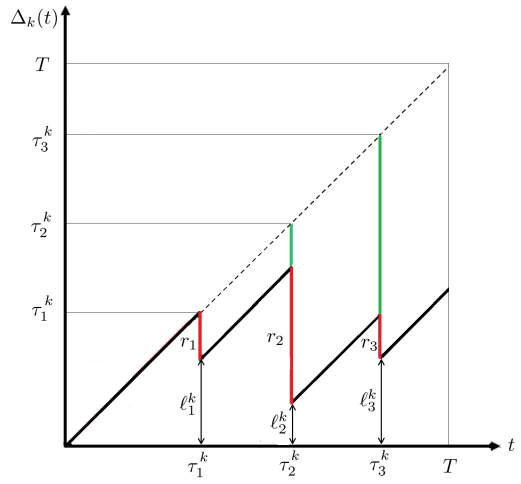

Figure 4.3: AoI of the physical process p_k with $U^k(T) = 3$ updates.

that the first update reduces the AoI by $r_1 \triangleq \tau_1^k - \ell_1^k = u_1^k$, the second update reduces the AoI by $r_2 \triangleq \tau_2^k - \ell_2^k - r_1 = u_2^k - u_1^k$, while the third update reduces the AoI by $r_3 \triangleq \tau_3^k - \ell_3^k - r_1 - r_2 = u_3^k - u_2^k$. Consequently, the instantaneous AoI of the physical process p_k can be expressed as

$$\Delta_k(t) = t - \sum_{j=1}^{U^k(t)} (u_j^k - u_{j-1}^k), \tag{4.11}$$

where $u_0^k = 0$ and $U_k(t)$ is the number of updates received before time instant t. Measurements from spatially proximal devices or successive measurements from the same device are correlated. To capture this, the instantaneous CAAoI of a physical process p_k at time instant t is defined as follows:

$$\Gamma_k(t) = t - \sum_{j=1}^{U^k(t)} \alpha_j^k (u_j^k - u_{j-1}^k), \tag{4.12}$$

where $0 \leq \alpha_j^k \leq 1$ reflects the novelty of the j-th update with respect of the previous $(j-1)$ updates of the physical process p_k, such that $\alpha_1^k = 1$ and α_j^k is defined as

$$\alpha_j^k = 1 - \frac{\xi_s + \xi_t}{\xi_s \dot{d}_j / \rho_s^k + \xi_t \dot{t}_j / \rho_t^k + \xi_s \xi_t + 1}, \tag{4.13}$$

with ρ_s^k and ρ_t^k as constant parameters that represent the spatial and temporal correlation extent in the physical process p_k, respectively, $\dot{d}_j$ as the minimum distance between the device that sends the j-th update and all the devices that have sent the $(j-1)$ updates of the physical process p_k, and $\dot{t}_j$ as the time difference between the current update and the last update about p_k from the same device.[2] Further, ξ_s and ξ_t are introduced to give the decision maker the ability to consider either spatial correlation ($\xi_s = 1, \xi_t = 0$), temporal correlation ($\xi_s = 0, \xi_t = 1$), or both spatial and temporal correlation ($\xi_s = \xi_t = 1$).

[2]The initial value of $\dot{t}_j$ is set to be $(1 - \xi_s)\xi_s^{-1}$.

4.3.1 System Model

In this work, a UAV-enabled data gathering scenario is considered, in which a UAV is given the mission of monitoring a set $\mathcal{P} = \{p_k\}_{k=1}^{P}$ of P physical processes by gathering information from a set $\mathcal{N} = \{n_i\}_{i=1}^{N}$ of N devices. Each device is able to sense one physical process; to represent the devices ability to observe the processes, $\boldsymbol{\chi} = [\chi_{ik}]_{N\times P}$ is defined such that

$$\chi_{ik} = \begin{cases} 1, & \text{if } n_i \text{ observes } p_k, \\ 0, & \text{otherwise.} \end{cases} \tag{4.14}$$

The UAV has a finite battery capacity and can only operate for a finite time interval. Consequently, the total observation time is T seconds. Initially, the UAV is placed at a docking station ψ_0 and the updates from devices are scheduled to keep the UAV updated about the status of the physical processes $\mathcal{P}$ during the observation time T. To gather information from device n_i, the UAV hovers at $\psi_i = \{x_i, y_i, h\}$, where (x_i, y_i) represents the coordinates of device n_i and h is the UAV's altitude.

A scheduling policy is represented by $(\hat{\boldsymbol{\eta}}, \hat{\boldsymbol{\mu}}, \boldsymbol{u})$, such that $\hat{\boldsymbol{\eta}} = [\hat{\eta}_i]_{1\times N}$ with $\hat{\eta}_i$ as the number of updates from device n_i, $\hat{\boldsymbol{\mu}} = [\hat{\mu}_\iota]_{1\times F}$ with $\hat{\mu}_\iota$ as the index of the device that transmits the ι-th update and $F = \sum_{i=1}^{N} \hat{\eta}_i$ as the total number of updates received at the UAV about $\mathcal{P}$ from $\mathcal{N}$, and $\boldsymbol{u} = [u_\iota]_{1\times F}$ with u_ι as the timestamp of the ι-th update. The time required for moving the UAV from the device that sends the $(\iota-1)$-th update to the one that sends the ι-th update is $\bar{\Upsilon}_\iota = \left\|\psi_{\hat{\mu}_{\iota-1}} - \psi_{\hat{\mu}_\iota}\right\|/v$, where v is the speed of the UAV and $\|\cdot\|$ is the second norm. The required time for the UAV to receive an update from device n_i is $\Upsilon_i = \frac{\sum_{k=1}^{P} \chi_{ik} I_k}{R_i}$, where I_k (in bits) is the payload size of an update of p_k. Device n_i has a finite capacity battery of $E_i^{\max}$ and each time it is scheduled to transmit an update to

the UAV, its battery level shrinks by $P_i \Upsilon_i$.

4.3.2 Problem Formulation

The objective is to find a scheduling that minimizes the time-average CAAoI of the processes of interest. The time-average CAAoI of the physical process p_k over time interval T can be expressed as

$$\begin{aligned} \langle \Gamma_k(t) \rangle_T &\triangleq \frac{1}{T} \int_0^T \Gamma_k(t)\, dt \\ &= \frac{T}{2} - \frac{1}{T} \sum_{j=1}^{U^k(T)} \alpha_j^k (u_j^k - u_{j-1}^k)(T - \tau_j^k), \end{aligned} \tag{4.15}$$

where $U^k(T)$ is the total number of updates about the physical process p_k during the observation interval T. It is worth mentioning that the maximum value of $\langle \Gamma_k(t) \rangle_T$ is $T/2$ and $\sum_{k=1}^{P} U^k(T) = F$. The objective function is a normalized weighted sum of time-average CAAoI of the processes of interest and the optimization problem is formulated as

$$\textbf{P1} \min_{\hat{\boldsymbol{\eta}}, \hat{\boldsymbol{\mu}}, \boldsymbol{u}} \quad O\left(\hat{\boldsymbol{\mu}}, \boldsymbol{u}\right) = \frac{1}{0.5T} \sum_{k=1}^{P} \lambda_k \langle \Gamma_k(t) \rangle_T, \tag{4.16a}$$

$$\text{s.t.} \quad \sum_{\iota=1}^{F} (\Upsilon_\iota + \bar{\Upsilon}_\iota) \leq T, \tag{4.16b}$$

$$P_i \Upsilon_i \hat{\eta}_i \leq E_i^{\max}, \forall n_i \in \mathcal{N}, \tag{4.16c}$$

$$\hat{\eta}_i \in \mathbb{Z}^+, \forall n_i \in \mathcal{N}, \tag{4.16d}$$

$$\hat{\mu}_\iota \in \{1, 2, \cdots, N\}, \tag{4.16e}$$

where λ_k is an importance weight for the physical process p_k, such that $\sum_{k=1}^{P} \lambda_k = 1$ and $\mathbb{Z}^+$ represents the set of non-negative integers. Constraint (4.16b) guarantees that the UAV has enough time to travel and receive all the scheduled updates. Constraint (4.16c) guarantees that each device is able to transmit all its scheduled updates.

4.3.3 Proposed Solution Approaches

4.3.3.1 Proposed Ant Colony Optimization Algorithm

The proposed ACO algorithm is presented as Algorithm 5, in which a colony of A ants collaborate to solve (4.16). The tour of each ant $a \in A$ starts by setting the time indication $t^{(a)} = 0$ and $\boldsymbol{s}^{(a)} = \mathbf{0}_{N\times N}$, and the updates from the devices are embedded in the scheduling policy $(\hat{\boldsymbol{\eta}}^{(a)}, \hat{\boldsymbol{\mu}}^{(a)}, \boldsymbol{u}^{(a)})$ until there is no time to receive more updates or none of the devices has sufficient transmission energy. The probability of scheduling the update from device n_l after the current device n_i is

$$\pi_{il}^{(a)} = \frac{\varepsilon_{il}^{(a)} \epsilon_{l}^{(a)} \left(\varrho_{il}^{(a)}\right)^{\gamma_1} \left(\delta_{il}\right)^{\gamma_2}}{\sum\limits_{\substack{n=1 \\ n\neq l}}^{N} \varepsilon_{nl}^{(a)} \epsilon_{n}^{(a)} \left(\varrho_{il}^{(a)}\right)^{\gamma_1} \left(\delta_{il}\right)^{\gamma_2}}. \tag{4.17}$$

The parameters in (4.17) are as follows:

- $\varrho_{il}^{(a)}$ is the attractiveness of scheduling the update from device n_l after the current device n_i; it is set to be[3]

$$\varrho_{il}^{(a)} = \frac{D \sum_{k=1}^{P} \chi_{lk} \lambda_k}{(1 + \hat{\eta}_l^{(a)}) \ell_{il}}, \tag{4.18}$$

  where $\ell_{il} = \|\psi_i - \psi_l\|$ is the distance between n_i and n_l, and D is a constant [43]. ϱ_{il} in (4.18) suggests more attractiveness to devices that monitor physical processes with higher λ_k, are in close proximity to n_i to minimize the UAV's traveling distances, and have less already scheduled updates.

- δ_{in} is the trail pheromone.

[3]For the conventional AoI, $\varrho_{il}^{(a)} = \frac{D \sum_{k=1}^{P} \chi_{lk} \lambda_k}{\ell_{il}}$.

- $\varepsilon_{il}^{(a)}$ indicates that there is enough time to receive the update from n_l; it is set to be

$$\varepsilon_{il}^{(a)} = \begin{cases} 1, & \text{if } t^{(a)} + \Upsilon_l + \ell_{il}/v \leq T, \\ 0, & \text{otherwise.} \end{cases} \tag{4.19}$$

- $\epsilon_{l}^{(a)}$ indicates that n_l has enough transmission energy; it is set to be

$$\epsilon_{l}^{(a)} = \begin{cases} 1, & \text{if } P_l \Upsilon_l \hat{\eta}_l^{(a)} \leq E_l^{\max}, \\ 0, & \text{otherwise.} \end{cases} \tag{4.20}$$

- γ_1 and γ_2 control the influence of the attractiveness and pheromone, respectively.

Once a device n_{l^*} is selected according to (4.17), the device index l^* is embedded to $\hat{\boldsymbol{\mu}}^{(a)}$, $t^{(a)} = t^{(a)} + \Upsilon_{l^*} + \ell_{il^*}/v$, the timestamp of the update is embedded to $\boldsymbol{u}^{(a)}$, and $F^{(a)}$ and the corresponding elements $\hat{\eta}_{l^*}^{(a)}$ and $s_{il^*}^{(a)} \in \boldsymbol{s}^{(a)}$ increase by one. The quality of a solution constructed by ant a is reflected in the deposited pheromone $\varpi^{(a)}$, which is set to be

$$\varpi^{(a)} \triangleq \frac{1}{O\left(\hat{\boldsymbol{\mu}}^{(a)}, \boldsymbol{u}^{(a)}\right)} = \frac{0.5T}{\sum_{k=1}^{P} \lambda_k \langle \Gamma_k^{(a)}(t) \rangle_T}. \tag{4.21}$$

A global updating rule is considered in the proposed ACO, in which the algorithm repeats I colonies and in each colony only two ants with highest and second-highest deposit pheromone according to (4.21) are allowed to deposit their pheromone [44]. The trail pheromone is updated as follows:

$$\delta_{il} \leftarrow (1 - \sigma)\, \delta_{il} + s_{il}^{(a)} \varpi^{(a)}, \tag{4.22}$$

where σ is the pheromone evaporation coefficient.

Algorithm 5 ACO algorithm for CAAoI information gathering.

Input: $\mathcal{N}$, T, $E_i^{\max}$, λ_k, ρ_s, ρ_t, A, and I;

Initialize δ_{il}; Calculate ℓ_{il} and Υ_i;

$O \leftarrow \infty$;

For Iteration $= 1$ to I **do**

  $O_1 \leftarrow \infty$; $O_2 \leftarrow \infty$;

  For $a = 1$ to A **do**

  $\hat{\boldsymbol{\eta}}^{(a)} = \mathbf{0}_{1\times N}$; $\hat{\boldsymbol{\mu}}^{(a)} \leftarrow \emptyset$; $\boldsymbol{u}^{(a)} \leftarrow \emptyset$; $t^{(a)} = 0$; $\boldsymbol{s}^{(a)} = \mathbf{0}_{N\times N}$;

  Set $i = 0$; Evaluate $\varepsilon_{il}^{(a)}$ and $\epsilon_l^{(a)}$ $\forall 1 \leq l \leq N$

  While $\prod_{l=1}^{N}(1-\varepsilon_{il}^{(a)}) + \prod_{l=1}^{N}(1-\epsilon_l^{(a)}) = 0$ **do**

    Select a device n_{l^*} using (4.17); $t^{(a)} = t^{(a)} + \Upsilon_{l^*} + \ell_{il^*}/v$;

    $s_{il^*}^{(a)} = s_{il^*}^{(a)} + 1$; ; $i = l^*$;

    $\hat{\eta}_i^{(a)} = \hat{\eta}_i^{(a)} + 1$; $\hat{\boldsymbol{\mu}}^{(a)} = [\hat{\boldsymbol{\mu}}^{(a)}\ i]$; $F^{(a)} = F^{(a)} + 1$;

    Re-evaluate $\epsilon_i^{(a)}$ and $\varepsilon_{il}^{(a)}$ $1 \leq l \leq N$;

  End While

  Evaluate $O(\hat{\boldsymbol{\mu}}^{(a)}, \boldsymbol{u}^{(a)})$ using (4.15) and (4.3);

  If $O > O(\hat{\boldsymbol{\eta}}^{(a)}, \hat{\boldsymbol{\mu}}^{(a)}, \dot{\hat{\boldsymbol{\mu}}}^{(a)})$

  $\hat{\boldsymbol{\eta}}^* \leftarrow \hat{\boldsymbol{\eta}}^{(a)}$; $\hat{\boldsymbol{\mu}}^* \leftarrow \hat{\boldsymbol{\mu}}^{(a)}$; $\boldsymbol{u}^* \leftarrow \boldsymbol{u}^{(a)}$;

  End if

  If $O_1 > O(\hat{\boldsymbol{\mu}}^{(a)}, \boldsymbol{u}^{(a)})$

  $\hat{\boldsymbol{\eta}}^{(a_1)} \leftarrow \hat{\boldsymbol{\eta}}^{(a)}$; $\hat{\boldsymbol{\mu}}^{(a_1)} \leftarrow \hat{\boldsymbol{\mu}}^{(a)}$; $\boldsymbol{u}^{(a_1)} \leftarrow \boldsymbol{u}^{(a)}$; $\boldsymbol{s}^{(a_1)} \leftarrow \boldsymbol{s}^{(a)}$;

  Else if $O_2 > O(\hat{\boldsymbol{\mu}}^{(a)}, \boldsymbol{u}^{(a)})$

  $\hat{\boldsymbol{\eta}}^{(a_2)} \leftarrow \hat{\boldsymbol{\eta}}^{(a)}$; $\hat{\boldsymbol{\mu}}^{(a_2)} \leftarrow \hat{\boldsymbol{\mu}}^{(a)}$; $\boldsymbol{u}^{(a_2)} \leftarrow \boldsymbol{u}^{(a)}$; $\boldsymbol{s}^{(a_2)} \leftarrow \boldsymbol{s}^{(a)}$;

  End if

End for

Deposit pheromone of a_1 and a_2 using (4.22);

End for

Return $\hat{\boldsymbol{\eta}}^*$, $\hat{\boldsymbol{\mu}}^*$, and $\boldsymbol{u}^*$.

The total number of updates that can be gathered by the UAV over a time duration T is upper bounded by

$$\tilde{F} = \left\lceil \frac{T}{\min\limits_{\substack{1\leq i,l\leq N \\ i\neq l}} \{\ell_{il}/v\} + \min\limits_{1\leq i\leq N} \{\Upsilon_i\}} \right\rceil. \tag{4.23}$$

The denominator in (4.23) is the minimum required time to travel between two devices plus the minimum required time to receive an update from a device. The search space of the optimization problem (4.16) is $\mathcal{O}(N(N-1)^{\tilde{F}-1})$. An ant a performs $\mathcal{O}(N^2\tilde{F})$ operations to construct a solution and $\mathcal{O}(N\tilde{F}^2\log(\tilde{F}))$ operations to evaluate the objective function. Consequently, the computational complexity of the ACO algorithm is $\mathcal{O}(IA[N^2\tilde{F} + N\tilde{F}^2\log(\tilde{F})] + IN^2) = \mathcal{O}(IA[N^2\tilde{F} + N\tilde{F}^2\log(\tilde{F})])$; this is remarkably lower than the complexity of the exhaustive search approach, which is $\mathcal{O}(N^2(N-1)^{\tilde{F}-1}\tilde{F}^2\log(\tilde{F}))$.

4.3.3.2 Proposed DRL Solution Method

The optimization problem in (4.16) can be solved using Q-learning, in which an agent learns to maximize the Q-function $Q_\pi(\boldsymbol{s}, \boldsymbol{a})$ —the expected return— by taking action $\boldsymbol{a}$ in state $\boldsymbol{s}$ while following policy π. The state, action, and reward are defined as follows:

- **State:** The state at the l-th step reflects three components: (1) the distance between the UAV at the current state and all the devices; (2) the energy level at each device; and (3) the time instant t_l, which is the required time to move from the initial state up to the l-th state i.e., $t_l = \sum_{q=1}^{l}(\Upsilon_q + \bar{\Upsilon}_q)$. Consequently, the l-th state is represented as $\boldsymbol{s}_l = [d_{l1}, d_{l2}, \cdots, d_{lN}, E_1(l), E_2(l), \cdots, E_N(l), t_l]$, with $d_{li} = \|\psi_l - \psi_i\|$ as the distance between the UAV at state l and device n_i. Note that the initial state is $\boldsymbol{s}_0 = [d_{01}, d_{02}, \cdots, d_{0N}, E_1^{\max}, E_2^{\max}, \cdots, E_N^{\max}, 0]$.

- **Action:** The action at the l-th step is an integer value in the action set $a_l \in \mathcal{A} = \{1, 2, \cdots, N\}$ such that $a_l = i$ means that device n_i is scheduled for transmission. In other words, at each state, the UAV adds a device to the end of the scheduling policy.

- **Reward:** The reward for performing action a_l in state $\boldsymbol{s}_l$ is defined as the reduction in the objective function in (4.3) due to implementing the action a_l, i.e., $\mathcal{R}_l = O(\hat{\boldsymbol{\mu}}_l, \boldsymbol{u}_l) - O(\hat{\boldsymbol{\mu}}_{l+1}, \boldsymbol{u}_{l+1})$, where $O(\hat{\boldsymbol{\mu}}_l, \boldsymbol{u}_l)$ is the objective function for all the updates from the initial state up to the l-th state while $O(\hat{\boldsymbol{\mu}}_{l+1}, \boldsymbol{u}_{l+1})$ is the objective function for all updates from the initial state up to the l-th state including a_l. It is worth mentioning that if no update has been scheduled (i.e., $\hat{\boldsymbol{\mu}}_0 = \boldsymbol{u}_0 = \emptyset$), then the time-average CAAoI of each physical process will be $T/2$ and the objective function reaches its maximum of 1 (i.e., $O(\hat{\boldsymbol{\mu}}_0, \boldsymbol{u}_0) = 1$).

It has been shown that Q-learning converges to the optimal solution if each state-action pair is visited in a sufficient number of training iterations [61]. However, iterating over all the state-action pairs becomes unattainable as the dimensionality of the state space increases. Consequently, a DRL model is considered, in which the Q-function is approximated by a DNN such that the Q-function becomes $Q(\boldsymbol{s}, \boldsymbol{a} \mid \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is the weight vector of the DNN [64]. The training steps of the proposed DRL algorithm are described in Algorithm 6, which starts by initializing the NN and the replay buffer **B** (lines 1). The algorithm iterates over E episodes.

For each episode, the algorithm iterates such that the l-th training iteration is performed by selecting the action (lines 6-8), applying the action to the environment (lines 9-14), and receiving the immediate reward and the next state (lines 16-18). After storing the transition tuple $(\boldsymbol{s}_l, \boldsymbol{a}_l, r_l, \boldsymbol{s}_{l+1})$ in the replay memory **B** (line 18), randomly selected samples of M

Algorithm 6 DRL algorithm for CAAoI information gathering.

1: Initialize the DNN weights θ and the replay buffer **B**;

2: **For** $episode = 1$ **to** E **do**

3: Initialize the environment and receive the initial state $\boldsymbol{s}_0$;

4: Set $l = 0$; $\hat{\boldsymbol{\eta}} = \mathbf{0}_{1\times N}$; $\hat{\boldsymbol{\mu}}_0 \leftarrow \emptyset$; $\boldsymbol{u}_0 \leftarrow \emptyset$; $\mathcal{R}(\hat{\boldsymbol{\mu}}_0, \boldsymbol{u}_0) = 1$;

5: **Repeat:**

6: Select an action a_l

7: With probability ϵ, select a random action $a_l \in \mathcal{A}$

8: Otherwise, select $a_l = \arg\max_{\forall a \in \mathcal{A}} Q(\boldsymbol{s}_l, a \mid \boldsymbol{\theta})$;

9: Execute action a_l: $i = a_l$;

10: $t_{l+1} = t_l + \Upsilon_i + \bar{\Upsilon}_i$;

11: **If** $E_i(l) \geq P_i \Upsilon_i$ and $t_{l+1} \leq T$

12: Append i to $\hat{\boldsymbol{\mu}}_l$ such that $\hat{\boldsymbol{\mu}}_{l+1} = [\hat{\boldsymbol{\mu}}_l,\ i]$; $\hat{\eta}_i = \hat{\eta}_i + 1$;

13: Append the timestamp $u = t_l + \Upsilon_i$ to $\boldsymbol{u}_{l+1} = [\boldsymbol{u}_l,\ u]$;

14: $E_i(l+1) = E_i(l) - P_i \Upsilon_i$;

15: **End If**

16: Observe the reward $\mathcal{R}_l = O(\hat{\boldsymbol{\mu}}_l, \boldsymbol{u}_l) - O(\hat{\boldsymbol{\mu}}_{l+1}, \boldsymbol{u}_{l+1})$;

17: Observe the next state $\boldsymbol{s}_{l+1}$;

18: Store the transition $(\boldsymbol{s}_l, a_l, \mathcal{R}_l, \boldsymbol{s}_{l+1})$ in **B**;

19: $l = l + 1$;

20: **Until** $\boldsymbol{s}_{l+1}$ is a terminal state.

21: Sample a random mini-batch of M transitions from **B**;

22: For each transitions in M obtain y_m such that

$$y_m = \begin{cases} \mathcal{R}_m, & \text{if } s_{m+1} \text{ is a terminal state,} \\ \mathcal{R}_m + \gamma \max_{\forall a \in \mathcal{A}} Q(\boldsymbol{s}_l, a \mid \boldsymbol{\theta}), & \text{otherwise.} \end{cases}$$

23: Update the weight of the DNN network by minimizing the loss $L(\boldsymbol{\theta}) = \frac{1}{M} \sum_{m=1}^{M} (y_m - Q(\boldsymbol{s}_m, \boldsymbol{a}_m \mid \boldsymbol{\theta}))^2$;

24: **End for**

transition tuples are utilized to train the DNN network (lines 21-23).

4.3.4 Simulation Results

The conventional AoI and CAAoI of the considered system model are evaluated. The devices are distributed in a 1×1 km^2 area and the main parameters are listed in Table 4.2.

Table 4.2: Simulation parameters of the age- and correlation-aware information gathering framework.

Parameter	**Value**	**Parameter**	**Value**	**Parameter**	**Value**
P_i	20 dBm	h	200 m	B	200 kHz
I_k	2560 Bytes	P_n	-110 dBm	$v_{\max}$	12 m/s
f_c	2 GHz	ζ_{LoS}	0 dB	ζ_{NLoS}	20 dB
T	30 min	β_1	10	β_2	0.03
γ_1/γ_2	1	A	100	I	10^3
B	10^5	mini-batch	32	episodes	10^6

Figure 4.4 illustrates both the conventional AoI and CAAoI of the considered system model with $P = 3$ physical processes that are monitored by spatially-correlated devices. It is seen that as N increases, both the CAAoI and conventional AoI reduce and the gap difference between them decreases as well. This is attributed to the fact that increasing N increases the diversity. Device diversity augments the diversity in the gathered information, which reduces the CAAoI. While increasing N, the inter-device distance also reduces, which enables the UAV to gather updates about the three physical processes more-frequently. To gain a deeper insight into such behavior, the right-side y-axis of Fig. 4.4 illustrates the corresponding total uncorrelated information gathered at the UAV. It can be seen that the uncorrelated information corresponding to the conventional AoI does not change versus N. This can be explained, as in order to reduce the conventional AoI, the

UAV gathers data from the closest subset of devices and keeps receiving replicas from the same devices. On the other hand, such replicas do not reduce the CAAoI. Thus, the UAV tries to gather data from all the available devices to minimize the CAAoI, which increases the gathered uncorrelated information.

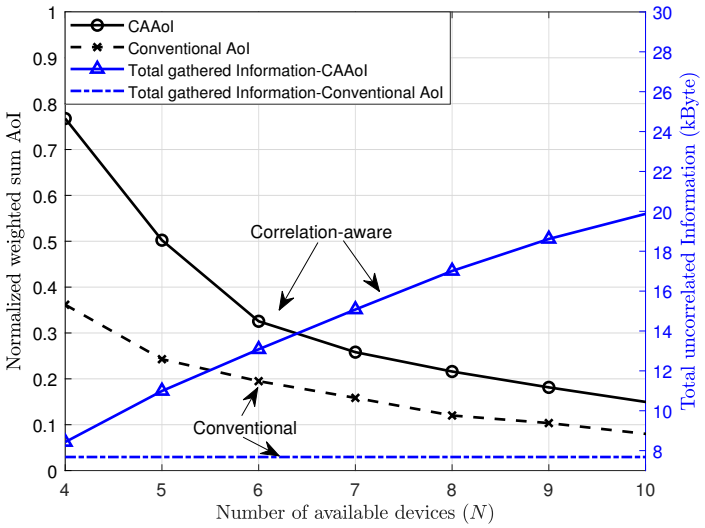

Figure 4.4: CAAoI and conventional AoI of the considered system model versus N with $P = 3$ process, $\xi_s = 1$, $\xi_t = 0$, and $\rho_s^k = 100$.

Figure 4.5 portrays the CAAoI versus ρ_t^k and ρ_t^k. It is clear that as the correlation among the data decreases (for small values of ρ_t^k and ρ_t^k), the novelty of each update increases, which reduces the CAAoI. The opposite is also valid, i.e., as the correlation in the data increases, the novelty of the updates decreases and the CAAoI increases. It is known that the online implementation time of the ACO is higher than that of the DRL. However, it is worth noting that the latter does not involve offline training effort and as illustrated in Fig. 4.6, it provides a slightly better performance in comparison with the former.

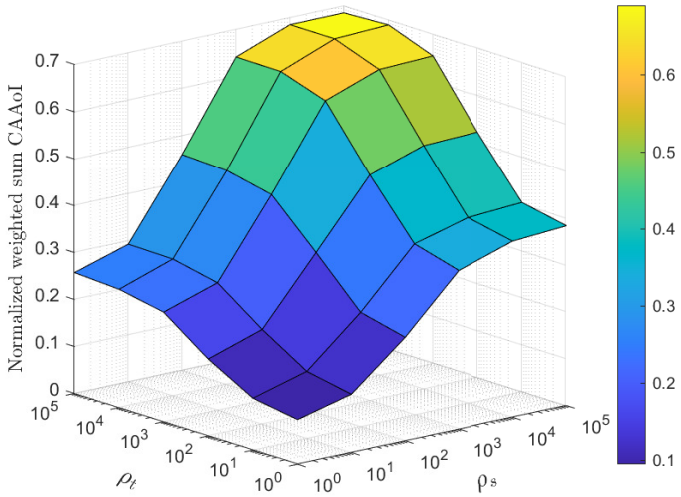

Figure 4.5: CAAoI of the considered system model versus $\rho_s = \rho_s^k$ and $\rho_t = \rho_t^k$ with $N = 20$ devices, $P = 3$ process, $\xi_s = 1$, and $\xi_t = 1$.

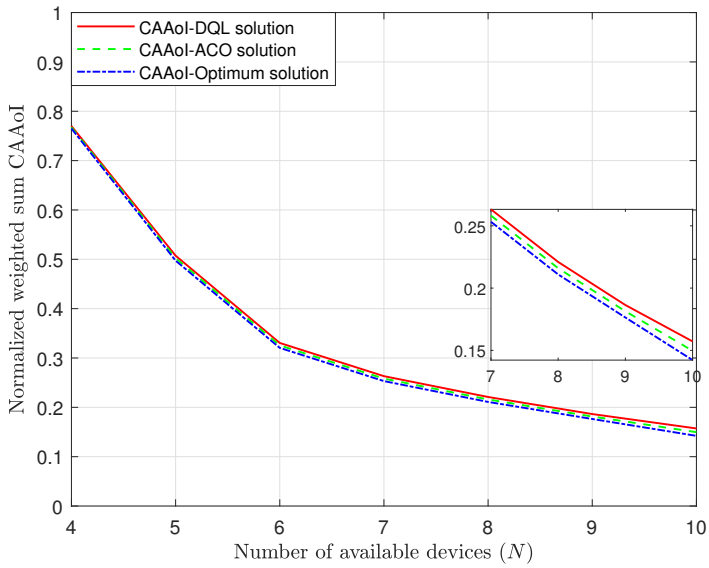

Figure 4.6: Performance of the ACO algorithm, DRL approach, and the exhaustive search approach.

4.4 Concluding Remarks

In this chapter, an age-optimal information gathering framework for linear networks has been proposed. In this framework, an AUV travels to collect data from a linearly-deployed set of devices to maintain freshness of the status of a set of physical processes of interest. An optimization problem has been formulated with the objective of minimizing the normalized weighted sum AoI. A DRL solution based on the DDPG actor-critic method has been developed. Simulation results have illustrated that the proposed framework maintains freshness and fairness of the physical processes of interest and shows robustness under different scenarios when compared with the baseline and K-means clustering approaches. Results also have shown that the proposed DRL-based solution optimizes the location and stopping time for each data gathering point, as well as the number of the data gathering points.

Moreover, the AoI concept has been extended by introducing a new CAAoI metric to capture both the freshness and diversity of gathered information. The CAAoI of an information gathering scenario has been studied, in which a UAV monitors a set of physical processes by gathering information from a set of devices. An ACO algorithm has been developed to minimize the CAAoI. Results have illustrated that the proposed CAAoI enables the UAV to maintain the freshness and diversity of the gathered information.

Chapter 5

Energy-Efficient Data Dissemination

5.1 Background and Motivation

Recently, numerous practical applications, including online gaming and on-demand video streaming, have driven communication networks to be more information-centric [65]. Information-centric networking refers to a novel networking paradigm that handles the information distribution in network rather than the host-to-host connectivity. In this paradigm, based on the devices' download requests and mobility, a decision-maker determines how to manage the resources in the network to optimize several objectives, including network load reduction, increased energy efficiency, and low dissemination latency. Placing the information near devices at intermediate or edge servers is the most attractive technique to eliminate redundant traffic and efficiently fulfill the device demands without duplicate transmissions from remote servers [66]. Consequently, heavy traffic in the network's backhaul can be managed during off-peak times, thus alleviating the backhaul's load during peak times. It is noteworthy to mention that information placement has its own challenging

issues, including determining which devices can be used as servers and determining the information that should be placed at each server. These issues become more sophisticated in mobile server scenarios and as the diversity of device requests increases.

Unmanned aerial vehicles (UAVs) have arisen as a favorable option for on-demand information aggregation/dissemination in current and future wireless networks [67]. Besides being agile, flexible, and mobile, UAVs can achieve line-of-sight (LoS) communication links with the ground devices, mitigating signal blockage and shadowing. Moreover, UAVs can be steered towards potential ground devices that are located in a geographical area with limited terrestrial infrastructure to establish reliable communication. The UAV role in communication networks can be classified into four main categories: (1) Aerial base station, in which a UAV is utilized to provide ubiquitous coverage when terrestrial base stations are completely out of service or to supplement the existing terrestrial base stations [68]; (2) UAV-aided relaying, in which a UAV is utilized as a relaying node that provides LoS connections between two or more distant devices [69]; (3) UAV-aided information aggregation/dissemination, in which a UAV aggregates/disseminates information from/to ground devices [10, 70]; (4) UAV-aided networks with caching, in which cache-enabled UAVs are deployed to alleviate the congested traffic on the network backhaul at peak time [71–73]. Keeping in mind the energy and payload limitations, the coverage range and communication performance of a single UAV network scenario is limited, which makes utilizing multiple UAVs a reasonable option to boost the UAV-enabled communication network performance [74].

Trajectory planning for UAV-enabled communication networks with single/multiple UAV(s) has recently attracted significant research interest to avail the additional degrees of freedom gained by the mobility of UAV(s) [75]. To reduce the impact of the UAV's

movement on the air-to-ground communication channel, such as the Doppler shift, in many scenarios, a UAV stops at a finite number of points to communicate with ground devices. Consequently, formulating the trajectory as a discrete-domain (combinatorial) optimization problem is a plausible approach. It is worth noting that computational intelligence, which refers to a set of nature-inspired solution approaches, such as genetic algorithms (GAs) [17] and ant colony optimization (ACO) [10], has been adopted as computationally efficient trajectory planning approaches [76].

5.1.1 Related Work

Various UAV-enabled scenarios with energy consumption minimization objective have been considered in the literature. In [77], the authors investigated the energy consumption minimization in a UAV-aided sensor network, where a set of devices act as cluster heads and receive data from other devices. To optimize the energy dissipation of a single UAV scenario, a combinatorial optimization problem was formulated to select the cluster heads from devices and plan the UAV's trajectory while visiting the selected cluster heads. In [78], an energy minimization problem was formulated to plan the trajectory of a single UAV, allocate time for the devices, and minimize the mission time. The scenario of a single UAV that cooperates with the ground base station to disseminate information with the aim of maximizing the delivered information to a set of ground devices was considered in [79]. The authors in [80] focused on optimizing a single UAV trajectory and the resource assignment to maximize the delivered information to spatially distributed ground devices. In [81], a UAV-aided multicasting scenario was considered, in which a single UAV delivers a common file to a group of ground users. The UAV's trajectory was planned to minimize the

trip time and guarantee that each device receives the file. In [10], the problem of optimizing the energy consumption of delivering a set of files to a set of ground devices using a single UAV was investigated. The UAV's trajectory was designed such that the UAV delivers the information to a subset of devices which forward the requested information to other devices. To overcome the endurance issue in a single UAV network that dispatches to serve a group of devices, the authors of [71] proposed a proactive caching scheme with the objective of minimizing a weighted sum of the file retrieval and caching costs.

A multi-UAV deployment strategy in a UAV-aided mobile edge computing network was considered in [82]. The location of each UAV along with the power control and user association were optimized to minimize the energy consumption. In [83], an efficient UAV placement to maximize the user coverage probability and reduce the inter-cell interference was considered. The scenario of a group of UAVs acting as relays to deliver information in a vehicular ad-hoc network was considered in [84]. The authors aimed to minimize the total transmission delay and maximize the total throughput. In [85], the trajectory of a set of UAVs was designed to maximize the sum-rate in uplink communication network. In [75], a sense-and-send protocol for multi-UAV networks was developed, in which the UAVs' trajectories are determined for sensing and transmission. A user-centric information framework UAV-enabled network was developed in [72]. In this framework, a set of cache storage units are mounted on UAVs to act as flying cache servers that predict the mobility pattern and information request distribution of users. In [73], the scenario of a UAV-enabled cache-assisted transmission for scalable videos was considered. In their framework, UAVs act as small-cell base stations that provide videos to mobile users.

This chapter addresses the problem of minimizing the energy expenditure of disseminating a library of files to a set of devices. Each device makes random requests from the

files' library. Two frameworks are proposed, the first framework is a two-tiered information dissemination system using a UAV. The second framework aims to optimally place the data in a fleet of UAVs and find the optimum traveling path of each UAV.

5.2 Energy-Efficient Data Dissemination Using a UAV

5.2.1 System Model

The considered system consists of a UAV which hovers at a fixed altitude h and is responsible for disseminating a library $\mathcal{F} = \{f_j\}_{j=1}^{F}$ of F files to a set $\mathcal{N} = \{n_i\}_{i=1}^{N}$ of N devices. Each device is interested in downloading a sub-set of the files $\mathcal{F}_i \subseteq \mathcal{F}$. To represent the files required by devices, $\boldsymbol{W} = [w_{ij}]_{N \times F}$ is defined such that

$$w_{ij} = \begin{cases} 1, & \text{if } n_i \text{ wants to download } f_j, \\ 0, & \text{otherwise.} \end{cases} \tag{5.1}$$

A two-tiered data dissemination scenario is considered: the devices in the first tier $\mathcal{T}_1$ receive the data directly from the UAV, while each device in the second tier $\mathcal{T}_2$ receives the data via one of the devices in the first tier, such that $\mathcal{T}_1 \cup \mathcal{T}_2 = \mathcal{N}$ and $\mathcal{T}_1 \cap \mathcal{T}_2 = \emptyset$. The devices are located at $\boldsymbol{\psi} = [\psi_i]_{N \times 3}$, where $\psi_i = \{x_i, y_i, 0\}$ represents the coordinates of device n_i. Assuming each device n_i has a transmission range r_i, the communication link matrix $\mathbf{C} = [c_{ik}]_{N \times N}$ is defined such that $c_{ik} = 1$ if device n_k is within the transmission range of device n_i (i.e., the distance between n_i and n_k, i.e., ℓ_{ik}, is less than r_i), and $c_{ik} = 0$ otherwise.

Furthermore, the device classification decision is denoted as $\boldsymbol{T} = [T_{i1}\ T_{i2}]_{N \times 2}$, in which $T_{i1} = 1$ and $T_{i2} = 0$ if n_i is classified as $\mathcal{T}_1$ or $T_{i1} = 0$ and $T_{i2} = 1$ if n_i is classified as $\mathcal{T}_2$.

It is clear that $\boldsymbol{T}$ classifies each device; however, it fails to associate each device in $\mathcal{T}_2$ with one of the devices in $\mathcal{T}_1$. Consequently, $\boldsymbol{\mu} = [\mu_{ik}]_{N\times N}$ is defined such that

$$\mu_{ik} = \begin{cases} 1, & \text{if } n_k \in \mathcal{T}_2 \text{ and receives data via } n_i \in \mathcal{T}_1, \\ 0, & \text{otherwise.} \end{cases} \tag{5.2}$$

Initially, the UAV is placed at a docking station ψ_0 which is the initial position to which it has to return after completing the data dissemination. The UAV stops above each device $n_i \in \mathcal{T}_1$; each stop is referred to as dissemination point. At the dissemination point $\dot{\psi}_i = \{x_i + e_i, y_i + e_i, h\}$ (e_i is the UAV's positioning error when hovers above n_i), the UAV transmits files required by n_i and all other devices in $\mathcal{T}_2$ which receive their data via n_i. The UAV's data dissemination trip $\bar{\boldsymbol{\psi}}$ is defined as the traveling path that starts at ψ_0, passed through all dissemination points, and returns to the docking station, i.e., $\bar{\boldsymbol{\psi}} = [\bar{\psi}_i]_{(V+2)\times 3}$, where $V = \sum_{i=1}^{N} T_{i1}$, and $\bar{\psi}_1 = \bar{\psi}_{V+2} = \psi_0$, with $\bar{\psi}_2, \bar{\psi}_3, \ldots, \bar{\psi}_{V+1}$ as an ordered set of all dissemination points.

For the terrestrial communication link between device n_i and device n_k, the average data rate is

$$\bar{R}_{ik} = B \log_2 \left(1 + \frac{P_{T_i}}{\varphi_{ik} N_0}\right), \tag{5.3}$$

where P_{T_i} is the transmit power of device n_i and $\varphi_{ik} = 1/(\gamma_0 |\zeta_{ik}|^2 \ell_{ik}^{-\epsilon})$ is the average channel path-loss, with γ_0 as the average reference channel power gain, ϵ as the path-loss exponent, and ζ_{ik} accounting for the small-scale channel fading from n_i to n_k [79, 86]. It is worth mentioning that both the UAV to devices channel and the inter-device channel are orthogonal. Additionally, it is assumed that the inter-device communication is performed using a time division multiple access scheme.

The energy consumed by the UAV at the location above the device n_i depends on the size of the files required by n_i and other devices with $\mu_{ik} = 1$. For a given $\boldsymbol{T}$ and $\boldsymbol{\mu}$, the data dissemination point above node n_i is associated with a vector $D^{(i)} = [d_j^{(i)}]_{1\times F}$, such that $d_j^{(i)} = T_{i1}w_{ij} \vee (\vee_{k=1}^{N} \mu_{ik}w_{kj})$, where $X \vee Y \vee \cdots \vee Z = 0$ if each of $X, Y, \ldots Z$ equals 0, and $X \vee Y \vee \cdots \vee Z = 1$ otherwise. In other words, $d_j^{(i)} = 1$ if n_i or any other device with $\mu_{ik} = 1$ requires to download f_j, and $d_j^{(i)} = 0$ otherwise. Consequently, the energy consumed by the UAV to communicate with the device n_i can be expressed as $\frac{T_{i1}P_T}{R} \sum_{j=1}^{F} d_j^{(i)} L_j$, where L_j is the size of file F_j in bits and P_T is the transmitted power of the UAV.

5.2.2 Problem Formulation

Classifying a large number of devices as $\mathcal{T}_1$ devices leads to a larger traveling time for the UAV, and consequently, an increased total energy consumption. However, it may ensure that the required files reach the corresponding devices either directly from the UAV or via devices in $\mathcal{T}_1$. On the other hand, a fewer number of devices in $\mathcal{T}_1$ decreases the total consumed energy. However, some devices may receive the required files neither from the UAV nor via the small set of devices in $\mathcal{T}_1$. Consequently, device classification should be intelligently implemented such that all required files are delivered and the total consumed energy is reduced. Moreover, the association of devices in $\mathcal{T}_2$ with those in $\mathcal{T}_1$ should be carefully performed such that the total number of files transmitted by the UAV at each data dissemination point is minimum. For a given $\boldsymbol{T}$ and $\boldsymbol{\mu}$, the total consumed energy by device

n_i can be calculated as:

$$\mathcal{E}_i\left(\boldsymbol{T},\boldsymbol{\mu}\right) = T_{i1}\left(\sum_{j=1}^{F} d_j^{(i)}\frac{P_r L_j}{R_{ii}} + \sum_{k=1}^{N}\sum_{j=1}^{F}\mu_{ik}w_{kj}\frac{P_{T_i}L_j}{\bar{R}_{ik}}\right) + T_{i2}\left(\sum_{k=1}^{N}\sum_{j=1}^{F}\mu_{ki}w_{ij}\frac{P_r L_j}{\bar{R}_{ki}}\right), \tag{5.4}$$

where P_r denotes the power consumed by the receiver circuitry of the device and R_{ii} is the data rate between device n_i and the UAV located at dissemination point $\dot{\psi}_i$, which can be obtained using (2.8). The energy consumed by the UAV during the dissemination trip $\bar{\boldsymbol{\psi}}$ can be written as:

$$\mathcal{E}_{\text{UAV}}\left(\boldsymbol{T},\boldsymbol{\mu},\bar{\boldsymbol{\psi}}\right) = \underbrace{\sum_{l=1}^{V+1}\left(P_{\text{hov}}+P_{\text{mov}}\right)\frac{\|\bar{\psi}_l-\bar{\psi}_{l+1}\|_2}{v}}_{\text{Path energy dissipation}} + \underbrace{\sum_{i=1}^{N}\sum_{j=1}^{F} T_{i1}d_j^{(i)'}\frac{\left(P_{\text{hov}}+P_T\right)L_j}{R_{ii}}}_{\text{Data dissemination energy dissipation}} \tag{5.5}$$

where v is the traveling speed of the UAV and P_{hov} and P_{mov} are the the UAV's hovering and moving power consumption and can be obtained using (2.6) and (2.7), respectively. The objective function is written as a weighted sum of the energy consumed by the UAV and devices as follows:

$$O\left(\boldsymbol{T},\boldsymbol{\mu},\bar{\boldsymbol{\psi}}\right) = \lambda\mathcal{E}_{\text{UAV}}\left(\boldsymbol{T},\boldsymbol{\mu},\bar{\boldsymbol{\psi}}\right) + (1-\lambda)\sum_{i=1}^{N}\mathcal{E}_i\left(\boldsymbol{T},\boldsymbol{\mu}\right), \tag{5.6}$$

where $0 \leq \lambda \leq 1$ is the relative weigh. Consequently, the optimization problem is formulated as

$$\mathbf{P1}\ \min_{\boldsymbol{T},\boldsymbol{\mu},\bar{\boldsymbol{\psi}}}\ O\left(\boldsymbol{T},\boldsymbol{\mu},\bar{\boldsymbol{\psi}}\right) \tag{5.7a}$$

$$\text{s.t.}\quad T_{i1}+T_{i2}=1,\ \forall\ 1\leq i\leq N, \tag{5.7b}$$

$$\sum_{i=1}^{N}\mu_{ik}T_{i1} = T_{k2},\ \forall\ 1\leq k\leq N, \tag{5.7c}$$

$$\mu_{ik}c_{ik} = \mu_{ik},\ \forall\ 1\leq i,k\leq N, \tag{5.7d}$$

$$T_{i1},T_{i2},\mu_{ik}\in\{0,1\},\ \forall\ 1\leq i,k\leq N. \tag{5.7e}$$

Constraint (5.7b) guarantees that each device belongs either to $\mathcal{T}_1$ or $\mathcal{T}_2$. Constraints (5.7c) guarantees that each device in $\mathcal{T}_2$ is associated with one device in $\mathcal{T}_1$. Constraint (5.7d) guarantees that there is a communication link between each device in $\mathcal{T}_2$ and the associated device in $\mathcal{T}_1$.

The optimum solution of (5.7) can be obtained by: searching over all possible classification decisions; for each classification decision, searching over all possible device association decisions and obtaining the optimum UAV's itinerary. The complexity of finding such a solution is prohibitive for a reasonable number of devices. Consequently, next section presents an efficient algorithm to solve (5.7).

5.2.3 Proposed Solution Approach

In the proposed ACO algorithm, a colony of A ants collaborate to solve (5.7). Each ant $a \in A$ travels through a three-step tour, in which it obtains $\boldsymbol{T}^{(a)}$ to classify the devices as $\mathcal{T}_1^{(a)}$ or $\mathcal{T}_2^{(a)}$, obtains $\boldsymbol{\mu}^{(a)}$ to associate devices in $\mathcal{T}_2^{(a)}$ with those in $\mathcal{T}_1^{(a)}$, and determines the corresponding UAV's data dissemination trip $\bar{\boldsymbol{\psi}}^{(a)}$, respectively.

In the first step, the probability that ant a selects device n_i to receive data from the UAV (i.e., probability of $T_{i1}^{(a)} = 1$) is calculated as

$$\dot{\pi}_i^{(a)} = \frac{\left(\dot{\tau}_i\right)^{\alpha} \left(\dot{\varrho}_i\right)^{\beta}}{\left(\dot{\tau}_i\right)^{\alpha} \left(\dot{\varrho}_i\right)^{\beta} + \left(\dot{\tau}_{0i}\right)^{\alpha} \left(\dot{\varrho}_{0i}\right)^{\beta}}, \tag{5.8}$$

where $\dot{\tau}_{0i}$ and $\dot{\tau}_i$ represent the trail pheromone, and $\dot{\varrho}_i$ and $\dot{\varrho}_{0i}$ are the attractiveness of classifying n_i tier $\mathcal{T}_1$ and tier $\mathcal{T}_2$, respectively. These are set to $\dot{\varrho}_i = \dot{\varrho}_{0i} = 1$ to give similar attractiveness of classifying n_i in $\mathcal{T}_1$ or $\mathcal{T}_2$.[1] Ant a obtains the device classification decision $\boldsymbol{T}^{(a)} = [T_{i1}^{(a)}\ T_{i2}^{(a)}]_{N\times 2}$, with $T_{i2}^{(a)} = 1 - T_{i1}^{(a)}$.

[1] α and β are parameters to control the influence of the pheromone and attractiveness, respectively.

In the second step, ant a associates each device in $\mathcal{T}_2^{(a)}$ with one in $\mathcal{T}_1^{(a)}$. It is worth noting that $\sum_{l=1}^{N} c_{lk} T_{l1}^{(a)} = 0$ indicates that there is no communication link between device n_k and any device in the first tier; in such a case, ant a reclassifies n_k to receive the files from the UAV directly by setting $T_{k1}^{(a)} = 1$ and $T_{k2}^{(a)} = 0$. The device association decision $\boldsymbol{\mu}^{(a)}$ is obtained such that the probability to set $\mu_{ik}^{(a)} = 1$ is calculated as

$$\ddot{\pi}_{ik}^{(a)} = \frac{c_{ik} T_{i1}^{(a)} T_{k2}^{(a)} \left(\ddot{\tau}_{ik}\right)^{\alpha} \left(\ddot{\varrho}_{ik}\right)^{\beta}}{\sum_{l=1}^{N} c_{lk} T_{l1}^{(a)} \left(\ddot{\tau}_{lk}\right)^{\alpha} \left(\ddot{\varrho}_{lk}\right)^{\beta}}, \tag{5.9}$$

where $\ddot{\tau}_{ik}$ is the trail pheromone and $\ddot{\varrho}_{ik}$ is the attractiveness of associating n_i with n_k. The latter is set to be

$$\ddot{\varrho}_{ik} = 1 + \frac{\sum_{j=1}^{F} w_{ij} w_{kj}}{F} \tag{5.10}$$

to provide more attractiveness to associating each device in $\mathcal{T}_2$ with the one in $\mathcal{T}_1$ that requires a similar set of files, and $\ddot{\varrho}_{ik} = 0$ if $i = k$.[1]

In the third step, the UAV's dissemination trip corresponding to ant a, $\bar{\boldsymbol{\psi}}^{(a)}$, is obtained. The probability with which ant a chooses ψ_k as the next dissemination point in the UAV's trip while being in dissemination point ψ_i, is calculated as

$$\dot{\ddot{\pi}}_{ik}^{(a)} = \frac{s_k^{(a)} T_{k1}^{(a)} (\dot{\ddot{\tau}}_{ik})^{\alpha} (\dot{\ddot{\varrho}}_{ik})^{\beta}}{\sum_{l=1}^{N} s_l^{(a)} T_{11}^{(a)} (\dot{\ddot{\tau}}_{lk})^{\alpha} (\dot{\ddot{\varrho}}_{lk})^{\beta}}, \tag{5.11}$$

where $s_k^{(a)}$ is a variable introduced to prevent the selection of ψ_k more than once, such that $s_k^{(a)} = 1$ if the dissemination point ψ_k has not been selected yet and $s_k^{(a)} = 0$ otherwise, $\dot{\ddot{\tau}}_{ik}$ is the trail pheromone, and $\dot{\ddot{\varrho}}_{ij}$ is the attractiveness of selecting ψ_k as the next dissemination point. This is set to

$$\dot{\ddot{\varrho}}_{ik} = \frac{\ell}{\ell_{ik}}, \tag{5.12}$$

which suggests the greedy heuristic of minimizing the UAV's traveling distances between dissemination points, with ℓ as a constant [43].[1] It is worth noting that based on the above described procedure, $\boldsymbol{T}^{(a)}$, $\boldsymbol{\mu}^{(a)}$, and $\bar{\boldsymbol{\psi}}^{(a)}$, satisfy the constraints in (5.7). In the proposed ACO, a global updating rule is adopted in which the globally best and second-best ants (i.e., ants which obtain the lowest and second-lowest $O(\boldsymbol{T}^{(a)}, \boldsymbol{\mu}^{(a)}, \bar{\boldsymbol{\psi}}^{(a)})\ \forall a \in A$) are allowed to deposit their pheromone [44]. The pheromone in each step is updated as follows:

$$\dot{\tau}_i \leftarrow (1-\sigma)\,\dot{\tau}_i + T_{i1}^{(a)}\Delta\tau^{(a)},\ \forall 1 \leq i \leq N, \tag{5.13a}$$

$$\dot{\tau}_{0i} \leftarrow (1-\sigma)\,\dot{\tau}_{0i} + T_{i2}^{(a)}\Delta\tau^{(a)},\ \forall 1 \leq i \leq N, \tag{5.13b}$$

$$\ddot{\tau}_{ik} \leftarrow (1-\sigma)\,\ddot{\tau}_{ik} + \mu_{ik}^{(a)}\Delta\tau^{(a^*)},\ \forall 1 \leq i,k \leq N, \tag{5.13c}$$

$$\dddot{\tau}_{ik} \leftarrow (1-\sigma)\,\dddot{\tau}_{ik} + u_{ik}^{(a)}\Delta\tau^{(a)},\ \forall 1 \leq i,k \leq N, \tag{5.13d}$$

where σ is the pheromone evaporation coefficient, and $u_{ik}^{(a)} = 1$ if the dissemination points ψ_i and ψ_k are successive points in the trip $\bar{\boldsymbol{\psi}}^{(a)}$ and $u_{ik}^{(a)} = 0$ otherwise. Further, $\Delta\tau^{(a)}$ is the incremental pheromone deposited by ant a, calculated as follows

$$\Delta\tau^{(a)} = \frac{E}{O(\boldsymbol{T}^{(a)}, \boldsymbol{\mu}^{(a)}, \bar{\boldsymbol{\psi}}^{(a)})}, \tag{5.14}$$

where E is a constant. The proposed ACO algorithm iterates I iterations and returns the best solution found so far. Algorithm 7 summarizes the proposed ACO approach.

5.2.4 Complexity Analysis

An ant a performs $\mathcal{O}(N)$, $\mathcal{O}(N^2F)$, and $\mathcal{O}(N^2)$ operations to obtain $\boldsymbol{T}^{(a)}$, $\boldsymbol{\mu}^{(a)}$, and $\bar{\boldsymbol{\psi}}^{(a)}$, respectively. Evaluating the objective function requires $\mathcal{O}(N^2F^2 + NF^2) = \mathcal{O}(N^2F^2)$ operations. Consequently, the computational complexity of the proposed ACO algorithm

Algorithm 7 ACO algorithm for data dissemination using a UAV.

Input: N, ψ_0, $\dot{\boldsymbol{\psi}}$, h, $\boldsymbol{W}$, $\boldsymbol{C}$, A, I, α, and β;
Initialize $\dot{\tau}_i$, $\dot{\tau}_{0i}$, $\ddot{\tau}_{ik}$, and $\dddot{\tau}_{ik}$ $\forall 1 \leq i, k \leq N$;
$O \leftarrow \infty$;
for $Iteration = 1$ to I **do**
  $O_1 \leftarrow \infty$; $O_2 \leftarrow \infty$;
  for $a = 1$ to A **do**
  Obtain $\boldsymbol{T}^{(a)}$, $\boldsymbol{\mu}^{(a)}$, and $\bar{\boldsymbol{\psi}}^{(a)}$ using (5.8), (5.9), and (5.11), respectively;
  Evaluate $O(\boldsymbol{T}^{(a)}, \boldsymbol{\mu}^{(a)}, \bar{\boldsymbol{\psi}}^{(a)})$ using (5.6);
  if $O > O(\boldsymbol{T}^{(a)}, \boldsymbol{\mu}^{(a)}, \bar{\boldsymbol{\psi}}^{(a)})$
    $\boldsymbol{T}^{*} \leftarrow \boldsymbol{T}^{(a)}$; $\boldsymbol{\mu}^{*} \leftarrow \boldsymbol{\mu}^{(a)}$; $\bar{\boldsymbol{\psi}}^{*} \leftarrow \bar{\boldsymbol{\psi}}^{(a)}$;
  end if
  if $O_1 > O(\boldsymbol{T}^{(a)}, \boldsymbol{\mu}^{(a)}, \bar{\boldsymbol{\psi}}^{(a)})$
    $\boldsymbol{T}^{(a_1)} \leftarrow \boldsymbol{T}^{(a)}$; $\boldsymbol{\mu}^{(a_1)} \leftarrow \boldsymbol{\mu}^{(a)}$; $\bar{\boldsymbol{\psi}}^{(a_1)} \leftarrow \bar{\boldsymbol{\psi}}^{(a)}$;
    else if $O_2 > O(\boldsymbol{T}^{(a)}, \boldsymbol{\mu}^{(a)}, \bar{\boldsymbol{\psi}}^{(a)})$
      $\boldsymbol{T}^{(a_2)} \leftarrow \boldsymbol{T}^{(a)}$; $\boldsymbol{\mu}^{(a_2)} \leftarrow \boldsymbol{\mu}^{(a)}$; $\bar{\boldsymbol{\psi}}^{(a_2)} \leftarrow \bar{\boldsymbol{\psi}}^{(a)}$;
  end if
  Deposit pheromone of a_1 and a_2 using (5.13);
  end for
end for
Return $\boldsymbol{T}^{*}$, $\boldsymbol{\mu}^{*}$, and $\bar{\boldsymbol{\psi}}^{*}$.

is $\mathcal{O}(N^4F^3AI + N^2I) = \mathcal{O}(N^4F^3AI)$; this is significantly lower than the complexity of exhaustive search approach, which is $\mathcal{O}(2^N[\frac{N(N-1)}{2} + N!]N^2F^2) = \mathcal{O}(2^N N!N^2F^2)$.

5.2.5 Simulation Results

The energy expenditure of the proposed framework is compared with a baseline approach in which the UAV stops above each device to disseminate data. In obtaining these results, it is considered that the N devices are placed at random in a 1×1 km^2 area. The default simulation parameters summarized in Table 5.1 are considered in the results, unless otherwise stated.

Table 5.1: Simulation parameters of the data dissemination using a UAV framework.

Parameter	**Value**	**Parameter**	**Value**	**Parameter**	**Value**
P_T	30 dBm	P_{T_i}	20 dBm	B	200 kHz
L_j	3 kbytes	N_0	-110 dBm	ϵ	3
$\hat{\alpha}$	10	$\hat{\beta}$	0.03	ζ_{NLoS}	20 dB
ζ_{LoS}	0 dB	e_i	$\sim U(0,2)$ m	f_c	2 GHz
γ_0	-60 dB	r_i	300 m	P_r	0.0126 W
P_{max}	5 W	P_{stop}	0 W	$v_{\text{max}} = v$	12 m/s
h	100 m	M	0.5 kg	r	20 cm
p	4	α	1	β	1
ℓ	10^3	E	10^3	$\lvert\zeta_{ik}\rvert^2$	$\sim$ Exp(1)
σ	0.1	A	100	I	10^3

Figures 5.1 illustrates the objective function in (5.6) versus the number of devices N, for both baseline approach and proposed framework. It is seen that the total energy expenditure of the proposed framework is less and flattens as N increases. This is due to the fact that as N increases, the number of inter-device connections increases, which is advantageous to the proposed framework.

In Fig. 5.2, it is seen that for small values of r_i, the proposed framework consumes more energy, while it preserves more energy as r_i increases. The energy expenditure of the baseline approach is not a function of r_i.

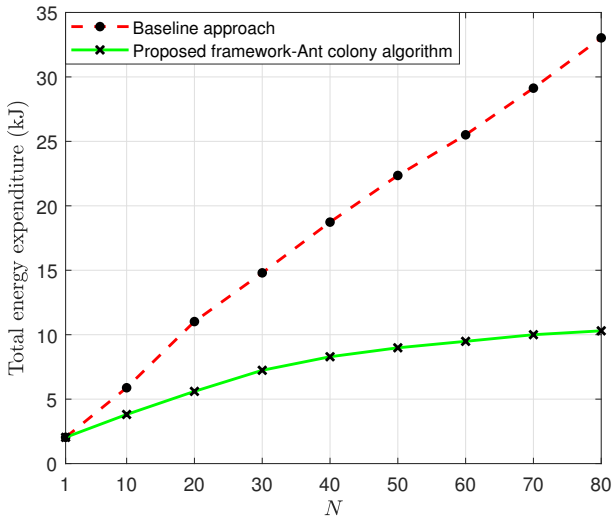

Figure 5.1: Energy expenditure versus the number of devices N with $F_i = 20$ files, $F = 70$ files, and $\lambda = 0.5$.

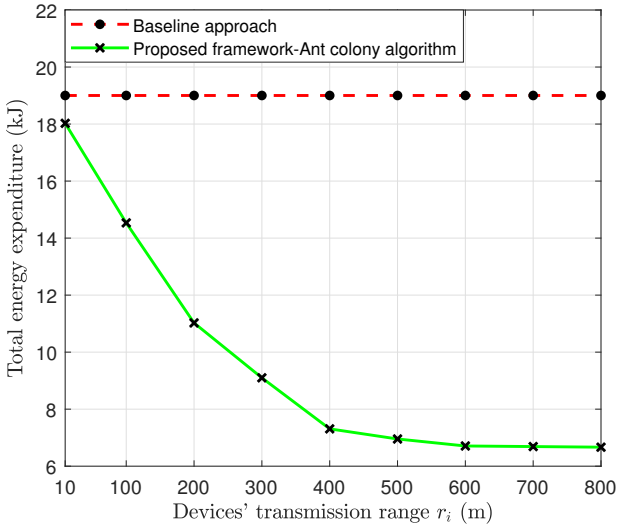

Figure 5.2: Energy expenditure versus the devices' transmission range r_i with $N = 40$, $F_i = 20$ files, $F = 70$ files, and $\lambda = 0.5$.

5.3 Energy-Efficient Information Placement and Delivery Using UAVs

5.3.1 System Model

The considered system consists of a fleet $\mathcal{U} = \{u_k\}_{k=1}^{K}$ of K UAV-mounted servers, each with storage capacity s_k. The UAVs hover to deliver the files library $\mathcal{F}$ to a set of devices $\mathcal{N}$. Each device requires a subset of files, which is represented by (5.1). The UAVs collectively have sufficient storage capacity for the entire files library, i.e., $\sum_{k=1}^{K} s_k \geq \sum_{j=1}^{F} L_j$. The initial locations of the UAVs are given by $\dot{\boldsymbol{\psi}}_0 = [\dot{\psi}_0^k]_{N\times 3}$, where $\dot{\psi}_0^k = \{x_0^k, y_0^k, h^k\}$ represents the coordinates of the initial location of UAV u_k. Let us define the file placement decision $\boldsymbol{\eta} = \left[\eta_{jk}\right]_{K\times F}$ such that

$$\eta_{jk} = \begin{cases} 1, & \text{if file } f_j \text{ is placed in UAV } u_k, \\ 0, & \text{otherwise.} \end{cases} \tag{5.15}$$

A placement decision $\boldsymbol{\eta}$ is feasible if $\sum_{j=1}^{F} \eta_{jk} L_j \leq s_k \; \forall \; u_k \in \mathcal{U}$. The devices are assigned to the UAVs using a device-UAV association decision $\hat{\boldsymbol{\mu}} = [\hat{\mu}_{ik}]_{N\times K}$ such that

$$\hat{\mu}_{ik} = \begin{cases} 1, & \text{if UAV } u_k \text{ delivers files to device } n_i, \\ 0, & \text{otherwise.} \end{cases} \tag{5.16}$$

A feasible $\hat{\boldsymbol{\mu}}$ should guarantee that each device receives the required files, i.e.,

$$\sum_{j=1}^{F} w_{ij} \leq \sum_{j=1}^{F} \sum_{k=1}^{K} w_{ij} \hat{\mu}_{ik} \eta_{jk}, \; \forall n_i \in \mathcal{N}. \tag{5.17}$$

To deliver the files to devices, each UAV u_k moves and hovers above each device with $\hat{\mu}_{ik} = 1$. To represent the path of each UAV, $\boldsymbol{\lambda} = \{\bar{\boldsymbol{\psi}}^{(1)}, \bar{\boldsymbol{\psi}}^{(2)}, \ldots, \bar{\boldsymbol{\psi}}^{(K)}\}$ is defined with

$\bar{\boldsymbol{\psi}}^k$ as the information delivery trip of UAV u_k that starts at $\dot{\psi}_0^k$, passes above all associated devices, and ends at $\dot{\psi}_0^k$.

For a given placement and delivery decision $(\boldsymbol{\eta}, \hat{\boldsymbol{\mu}}, \boldsymbol{\lambda})$, the energy consumed by UAV u_k can be expressed as:

$$\mathcal{E}_k\left(\boldsymbol{\eta}, \hat{\boldsymbol{\mu}}, \boldsymbol{\lambda}\right) = \underbrace{\sum_{l=1}^{V^{(k)}+1} \left(P_{\text{hov}}^{(k)} + P_{\text{mov}}^{(k)}\right) \frac{\|\bar{\psi}_l^{(k)} - \bar{\psi}_{l+1}^{(k)}\|_2}{v^{(k)}}}_{\text{Traveling path energy consumption}} + \underbrace{\sum_{i=1}^{N}\sum_{j=1}^{F} w_{ij}\hat{\mu}_{ik}\eta_{jk} \frac{\left(P_{\text{hov}}^{(k)} + P_{T_k}\right) L_j}{R_{ki}}}_{\text{Information transmission energy consumption}}. \tag{5.18}$$

5.3.2 Problem Formulation

The devices dissipate energy during the information reception, which is negligible in comparision to the UAVs' transmission, hovering, and travailing energy consumption. Based on this, the objective function is written as

$$O\left(\boldsymbol{\eta}, \hat{\boldsymbol{\mu}}, \boldsymbol{\lambda}\right) = \sum_{k=1}^{K} \mathcal{E}_k\left(\boldsymbol{\eta}, \hat{\boldsymbol{\mu}}, \boldsymbol{\lambda}\right). \tag{5.19}$$

Consequently, the formulated optimization problem is expressed as

$$\mathbf{P1} \min_{\boldsymbol{\eta}, \hat{\boldsymbol{\mu}}, \boldsymbol{\lambda}} \; O\left(\boldsymbol{\eta}, \hat{\boldsymbol{\mu}}, \boldsymbol{\lambda}\right) \tag{5.20a}$$

$$\text{s.t.} \quad \sum_{j=1}^{F} \eta_{jk} L_j \leq s_k \; \forall \; u_k \in \mathcal{U}, \tag{5.20b}$$

$$\sum_{j=1}^{F} w_{ij} \leq \sum_{j=1}^{F}\sum_{k=1}^{K} w_{ij}\hat{\mu}_{ik}\eta_{jk}, \; \forall n_i \in \mathcal{N}, \tag{5.20c}$$

$$n_i \in \lambda^{(k)}, \; \forall \; \hat{\mu}_{ik} = 1 \; \forall \; u_k \; \in \mathcal{U}, \tag{5.20d}$$

$$\eta_{jk}, \hat{\mu}_{ik} \in \{0, 1\}, \; \forall n_i \in \mathcal{N}, f_j \in \mathcal{F}, u_k \in \mathcal{U}. \tag{5.20e}$$

Here, constraint (5.20b) guarantees that each UAV can accommodate the placed information. Constraint (5.20c) guarantees that each device receives all required files. Finally,

constraint (5.20d) guarantees that each UAV visits all associated devices.

5.3.3 Proposed Solution Approaches

5.3.3.1 Multi-Chromosome Genetic Algorithm Approach

In the proposed multi-chromosome genetic algorithm (MCGA), a potential solution is represented by an individual which consists of a set of $2K$ chromosomes. The chromosomes are divided equally into two groups: Group 1 chromosomes $\mathcal{C} = \{C_1, C_2, \ldots, C_K\}$ and group 2 chromosomes $\dot{\mathcal{C}} = \{\dot{C}_1, \dot{C}_2, \ldots, \dot{C}_K\}$. The unique genes of chromosomes in group 1 are devices' indexes associated with the UAVs; while those in group 2 are files indices placed at the UAVs. Figure 5.3 illustrates an individual that represents a prospective solution for a system model of $K = 4$ UAVs, $N = 14$ devices, and a library of $F = 10$ files.

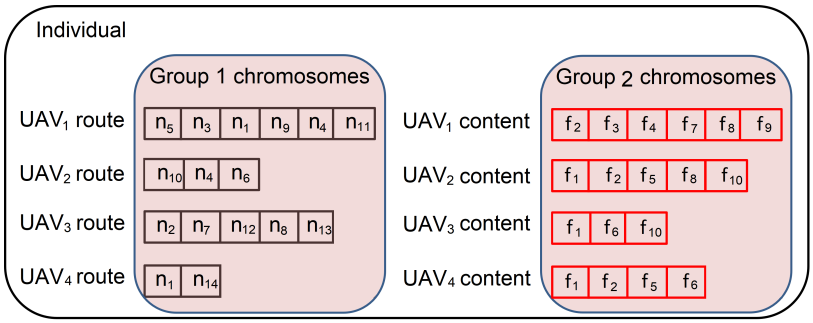

Figure 5.3: Example of an individual representation with $K = 4$ UAVs, $N = 14$ devices, and a catalogue of $C = 10$ content items.

Two sets of mutation operators can be performed within each group of chromosomes as follows:

- A cross-chromosome mutation operator modifies two chromosomes at once; this in-

cludes: (1) A swap operator, which is realized by transposing randomly selected sequences of genes from two chromosomes. If one of the selected gene sequences is empty, the operator becomes an insertion of the non-empty sequence to the other chromosome; (2) Crossover operator which does a one-point crossover between two chromosomes; and (3) Grouping operator which is realized by moving the genes from one chromosome to another, such that the source chromosome becomes empty and the destination chromosome contains a unique genes of the two chromosomes.

- In-chromosome mutation operator is performed on a single chromosome; this includes the gene sequence inversion or flip operator, the swapping two genes operator, and the change of the location of a set of genes within the chromosome. It is worth noting that the in-chromosome operators do not affect the solution quality when applied to the $\dot{\mathcal{C}}$ chromosomes; thus, in-chromosome operators are not applied to this group of chromosomes.

Algorithm 8 illustrates the pseudocode of the proposed MCGA, which receives the system model parameters and starts by initializing a population of P individuals and evaluates the fitness of each individual in the population (lines $1-3$). The fitness of a feasible individual I is defined as

$$\Theta_I = \frac{1}{O(\boldsymbol{\eta}^I, \hat{\boldsymbol{\mu}}^I, \boldsymbol{\lambda}^I)}, \tag{5.21}$$

and $\Theta_I = 0$ if I does not satisfy the constraints in (5.20). The algorithm performs M iterations; in each iteration, A individuals with the highest fitness values are selected from P (line 6). Each selected individual undergoes s_1 cross-chromosome operations and s_2 in-chromosome operations (line 7). Each operation leads to a child individual, which is evaluated using (5.21) and embedded to the population in lieu of the individual with lowest

fitness value (lines $8-12$). The algorithm traces and returns the best solution yet obtained (lines $13-18$).

Algorithm 8 MCGA algorithm for information placement and delivery using UAVs.

1: **Input** $\boldsymbol{\psi}$, $\dot{\boldsymbol{\psi}}$, $\boldsymbol{R}$, $s_k \; \forall \; u_k \; \in \; \mathcal{U}$, and $L_j \; \forall \; f_j \; \in \; \mathcal{F}$

2: Generate an initial population of P individuals;

3: Obtain the corresponding $\boldsymbol{\eta}^I, \hat{\boldsymbol{\mu}}^I, \boldsymbol{\lambda}^I$ of each individual I in P and evaluate its fitness using (5.21);

4: $O \leftarrow \infty$;

5: **for** Iteration$= 1$ to M **do**

6: Select A individuals for reproduction;

7: For each selected individual, generate $S = s_1 \times s_2$ children individuals by applying s_1 cross-chromosome operations to both $\mathcal{C}$ and $\dot{\mathcal{C}}$, and s_2 in-chromosome operations to $\mathcal{C}$;

8: **for** $s = 1$ to S **do**

9: Obtain the corresponding $\boldsymbol{\eta}^{I_s}, \hat{\boldsymbol{\mu}}^{I_s}, \boldsymbol{\lambda}^{I_s}$;

10: Evaluate the fitness of the child individual I_s;

11: Replace the individual with the lowest fitness value in the population by I_s;

12: **end for**

13: $I^* = \arg \max_{\forall I \in P} \{\Theta_I\}$

14: **if** $O > O(\boldsymbol{\eta}^{I^*}, \hat{\boldsymbol{\mu}}^{I^*}, \boldsymbol{\lambda}^{I^*})$

15: $\boldsymbol{\eta}^* \leftarrow \boldsymbol{\eta}^{I^*}$; $\hat{\boldsymbol{\mu}}^* \leftarrow \hat{\boldsymbol{\mu}}^{I^*}$; $\boldsymbol{\lambda}^* \leftarrow \boldsymbol{\lambda}^{I^*}$;

16: **end if**

17: **end for**

18: **Return** $\boldsymbol{\eta}^*$, $\hat{\boldsymbol{\mu}}^*$, and $\boldsymbol{\lambda}^*$.

5.3.3.2 Hybrid Multi-Chromosome Genetic Algorithm-Ant Colony Optimization (MCGA-ACO) Solution Approach

Similar to the MCGA, in MCGA-ACO the group 1 chromosomes of an individual are obtained using cross-chromosome and in-chromosome mutation operations. However, an ACO agent utilizes both the group 1 chromosomes and the distributed memory (pheromone) to obtain the group 2 chromosomes. The ACO agent places a gene j (index of file f_j) in

chromosome $\dot{C}_k$ (in UAV u_k) with probability

$$\pi_{jk} = \frac{\left(\tau_{jk}\right)^{\alpha}\left(\varrho_{jk}\right)^{\beta}}{\sum_{l=1}^{K}\left(\tau_{jl}\right)^{\alpha}\left(\varrho_{jl}\right)^{\beta}}, \tag{5.22}$$

where τ_{jk} represents the trail pheromone and ϱ_{jk} represents the attractiveness of placing f_j in UAV u_k; it is set to be

$$\varrho_{jk} = H\left(s_k - \sum_{\hat{j}=1}^{F}(1-\eta_{\hat{j}k}^{I})L_{\hat{j}}\right)\sum_{i=1}^{N}\left[\hat{\mu}_{ik}^{I}w_{ij}\prod_{\hat{k}=1}^{K}(1-\hat{\mu}_{i\hat{k}}^{I}\eta_{j\hat{k}}^{I})\right], \tag{5.23}$$

where $H(v)$ is a function whose value is zero for $v < 0$ and one for $v \geq 0$. In (5.23), ϱ_{jk} suggests more attractiveness to place the files requested by the devices served by the UAV u_k. Once an individual is obtained, the deposited pheromone $\Delta\tau$ is set to

$$\Delta\tau = \frac{\delta}{O(\boldsymbol{\eta}^{I}, \hat{\boldsymbol{\mu}}^{I}, \boldsymbol{\lambda}^{I})}, \tag{5.24}$$

where δ is a constant. The trail pheromone is updated as follows:

$$\tau_{jk} \leftarrow (1-\sigma)\,\tau_{jk} + \eta_{jk}^{I}\Delta\tau, \tag{5.25}$$

where σ represents the pheromone evaporation parameter.

Algorithm 9 illustrates the pseudocode of the proposed MCGA-ACO solution, which receives the system model parameters and starts by initializing the trail pheromone τ_{jk} and generates the group 1 chromosomes of the individuals in the initial population (lines 1-3). The group 2 chromosomes of the individuals are obtained using (5.22) and the fitness of each individual is evaluated. The algorithm performs M iterations; in each iteration, A individuals with the highest fitness values are selected from P (line 7). The group 1 chromosomes of each selected individual undergoes S cross-chromosome and in-chromosome operations (line 8). Each operation generates the group 1 chromosomes of a child individual and the ACO agent generates the group 2 chromosomes using (5.22). The child

individual is embedded to the population in lieu of the individual with lowest fitness value (lines $9-15$). The algorithm traces and returns the best solution yet obtained (lines $17-19$).

Algorithm 9 Hybrid MCGA-ACO algorithm for information placement and delivery using UAVs.

1: **Input** $\boldsymbol{\psi}$, $\dot{\boldsymbol{\psi}}$, $\boldsymbol{R}$, $s_k \; \forall \; u_k \; \in \; \mathcal{U}$, and $L_j \; \forall \; f_j \; \in \; \mathcal{F}$

2: $\tau_{jk} = 1 \; \forall 1 \leq j \leq F, \; 1 \leq k \leq K$;

3: Generate group 1 chromosomes of the initial population;

4: Obtain the group 2 chromosomes of each individual using (5.22);

5: Obtain the corresponding $\boldsymbol{\eta}^I, \hat{\boldsymbol{\mu}}^I, \boldsymbol{\lambda}^I$ of each individual I in P, evaluate its fitness using (5.21), and update τ_{jk} using (5.25);

6: **for** Iteration$= 1$ to M **do**

7: Select A individuals for reproduction;

8: For each selected individual, generate S children individuals by applying S cross-chromosome and in-chromosome operations to $\mathcal{C}$;

9: **for** $s = 1$ to S **do**

10: Obtain the group 2 chromosomes of the child individual I_s using (5.22);

11: Obtain the corresponding $\boldsymbol{\eta}^{I_s}, \hat{\boldsymbol{\mu}}^{I_s}, \boldsymbol{\lambda}^{I_s}$;

12: Evaluate the fitness of the child individual I_s;

13: Update τ_{jk} using (5.25);

14: Replace the individual with the lowest fitness value in the population by I_s;

15: **end for**

16: **end for**

17: $I^* = \arg \max_{\forall I \in P} \{\Theta_I\}$

18: $\boldsymbol{\eta}^* \leftarrow \boldsymbol{\eta}^{I^*}; \; \hat{\boldsymbol{\mu}}^* \leftarrow \hat{\boldsymbol{\mu}}^{I^*}; \; \boldsymbol{\lambda}^* \leftarrow \boldsymbol{\lambda}^{I^*};$

19: **Return** $\boldsymbol{\eta}^*$, $\hat{\boldsymbol{\mu}}^*$, and $\boldsymbol{\lambda}^*$.

5.3.3.3 Multi-Chromosome Genetic Algorithm with Heuristic File Placement Solution Approach

Similar to the MCGA, the group 1 chromosomes of the children individuals are obtained using cross-chromosome and in-chromosome mutation operations. The files are placed heuristically by sorting them in a descending order according to a weight. This weight is

defined as follows

$$\Theta_{jk} = \sum_{i=1}^{N} \left[\hat{\mu}_{ik}^{I} w_{ij} \prod_{\hat{k}=1}^{K} (1 - \hat{\mu}_{i\hat{k}}^{I} \eta_{j\hat{k}}^{I}) \right], \tag{5.26}$$

which suggests a higher priority to place files in UAV u_k based on the requirements of devices served by u_k and have not been scheduled to be delivered by other UAVs.

Algorithm 10 illustrates the pseudocode of the proposed MCGA with heuristic file placement solution, which receives the system model parameters and starts by generating the group 1 chromosomes of the individuals in the initial population (lines 1-2). The group 2 chromosomes of each individual are obtained using (5.26) and the fitness of each individual is evaluated. Similar to algorithm 2, the algorithm performs M iterations, where, in each iteration, A individuals with the highest fitness values are selected from P (line 7). The group 1 chromosomes of each selected individual undergoes S cross-chromosome and in-chromosome operations (line 8). Each operation generates group 1 chromosomes of a child individual while group 2 chromosomes is obtained using the weighting criteria in (5.26). Similar to algorithms 1 and 2, the child individual is embedded to the population in lieu of the individual with lowest fitness value (lines $9 - 15$). The algorithm traces and returns the best solution yet obtained (lines $17 - 19$).

5.3.4 Complexity Analysis

Generating an individual in the initial population of MCGA algorithm needs $\mathcal{O}\left(K[N+F]\right)$ operations. A mutation operator leads to an offspring individual that requires $\mathcal{O}\left(K[N+F]\right)$ operations. Evaluating the fitness and feasibility of an individual requires $\mathcal{O}\left(6KNF\right)$ and $\mathcal{O}\left(3KNF\right)$ operations, respectively. Consequently, the computational complexity of the MCGA approach can be expressed as $\mathcal{O}\left(MSK^2NF[N+F]AP\log(P)\right)$.

Algorithm 10 MCGA with heuristic file placement algorithm.

Input $\boldsymbol{\psi}$, $\dot{\boldsymbol{\psi}}$, $\boldsymbol{R}$, $s_k \; \forall \; u_k \; \in \; \mathcal{U}$, and $L_j \; \forall \; f_j \; \in \; \mathcal{F}$

Generate group 1 chromosomes of the initial population;

Obtain the group 2 chromosomes of each individual using (5.26);

Obtain the corresponding $\boldsymbol{\eta}^I, \hat{\boldsymbol{\mu}}^I, \boldsymbol{\lambda}^I$ of each individual I in P and evaluate its fitness using (5.21);

for Iteration$= 1$ to M **do**

Select A individuals for reproduction;

For each selected individual, generate S children individuals by applying S cross-chromosome and in-chromosome operations to $\mathcal{C}$;

  for $s = 1$ to S **do**

  Obtain the group 2 chromosomes of the child individual I_s using (5.26);

  Obtain the corresponding $\boldsymbol{\eta}^{I_s}, \hat{\boldsymbol{\mu}}^{I_s}, \boldsymbol{\lambda}^{I_s}$;

  Evaluate the fitness of the child individual I_s;

  Replace the individual with the lowest fitness value in the population by I_s;

  end for

end for

$I^* = \arg \max_{\forall I \in P} \{\Theta_I\}$

$\boldsymbol{\eta}^* \leftarrow \boldsymbol{\eta}^{I^*}; \; \hat{\boldsymbol{\mu}}^* \leftarrow \hat{\boldsymbol{\mu}}^{I^*}; \; \boldsymbol{\lambda}^* \leftarrow \boldsymbol{\lambda}^{I^*}$

Return $\boldsymbol{\eta}^*$, $\hat{\boldsymbol{\mu}}^*$, and $\boldsymbol{\lambda}^*$.

Generating group 1 chromosomes in an individual requires $\mathcal{O}\left(KN\right)$ operations while obtaining group 2 chromosomes using the ACO agent requires $\mathcal{O}\left(K^3NF\right)$ operations. Deposing the pheromone requires $\mathcal{O}\left(FK\right)$ operations. Consequently, the complexity of the MCGA-ACO solution approach can be expressed as $\mathcal{O}\left(K^5N^2F^3MSAP\log(P)\right)$. On the other hand, obtaining group 2 chromosomes in MCGA with heuristic file placement algorithm requires $\mathcal{O}\left(K^2NF^2\log(F)\right)$ operations. Consequently, the computational complexity of the MCGA with heuristic file placement algorithm can be expressed as $\mathcal{O}\left(PK^2NF^2\log(F) + MSK^2NF^2\log(F)AP\log(P)\right) = \mathcal{O}\left(MSK^2NF^2\log(F)AP\log(P)\right)$.

5.3.5 Simulation Results

This section introduces simulation results to evaluate the energy consumption of the considered system model with three solution approaches. It is considered that the N devices are distributed randomly in a 1×1 km^2 area. Unless otherwise stated, the default numerical values of the system parameters are listed in Table 5.2.

Table 5.2: Simulation parameters of the information placement and delivery using UAVs framework.

Parameter	**Value**	**Parameter**	**Value**	**Parameter**	**Value**
P_{T_k}	30 dBm	σ	0.1	B	200 kHz
L_j	10 kbytes	N_0	-110 dBm	α	1
α_1	10	α_2	0.03	ζ_{NLoS}	20 dB
ζ_{LoS}	0 dB	e_{ki}	$\sim U(0,2)$ m	f_c	2 GHz
$P_k^{\max}$	5 W	P_k^{stop}	0 W	$v_k^{\max} = v_k$	12 m/s
h_k	100 m	M_k	0.5 kg	ρ_k	20 cm
p_k	4	α	1	β	1
M	400 iteration	A	10 individuals	S	10

Figure 5.4 shows the energy consumption versus the average number of files required by each device. The performance of the three solution approaches are compared with the optimum solution, which is obtained through exhaustive search. It is clear that the proposed solutions provide near optimum performance. Furthermore, it can be noticed that the MCGA-ACO approach outperforms both the MCGA and the MCGA with heuristic file placement approaches.

The energy consumption versus the number of IoT devices N is illustrated in Fig. 5.5. It is noticed that the energy consumption increases linearly with the number of devices. Similar to the results in Fig. 5.4, the MCGA-ACO approach outperforms both the MCGA

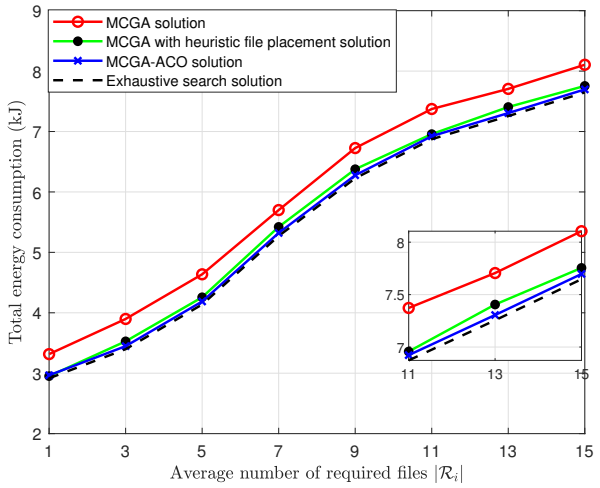

Figure 5.4: Energy consumption versus the average number of required files $|\mathcal{R}_i|$ with $N = 5$ devices, $K = 2$ UAVs, and $F = 15$ files.

and the MCGA with heuristic file placement approaches, which indicates that the ACO agent provides better file placement.

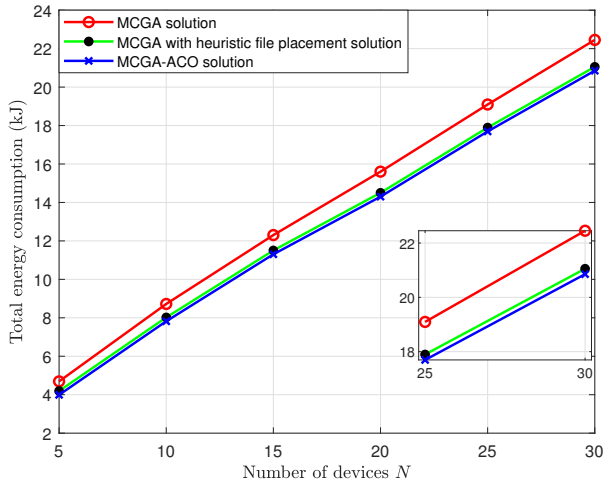

Figure 5.5: Energy consumption versus the number of devices N with $K = 5$ UAVs, $F = 20$ files, and each device requests $|\mathcal{R}_i| = 5$ files.

5.4 Concluding Remarks

In this chapter, two energy-efficient data dissemination frameworks have been developed. In the first framework, the devices are classified as devices which receive data directly from a UAV and via other devices. An optimization problem was formulated to minimize the energy expenditure of the UAV and devices. An ACO algorithm was developed to solve the optimization problem. Simulation results revealed that the proposed ACO algorithm provides a near-optimum performance, and the proposed framework reduces the energy expenditure when compared with a baseline approach.

The second framework has been developed for energy-efficient information placement and delivery in a multi-UAV network. The storage capacity of the UAVs has been considered and a combinatorial optimization problem has been formulated to find the optimum set of contributing UAVs, place the files, and plan the trajectory of each contributing UAV. Three computationally efficient solutions have been developed to solve the optimization problem. Simulation results have shown that the proposed framework preserves energy and optimizes the number of contributing UAVs. Moreover, results have illustrated that the developed solution approaches provide near-optimal performance.

Chapter 6

Task Offloading for Mobile Edge Computing

6.1 Background and Motivation

In recent years, there has been a tremendous growth of computationally-intensive mobile applications, such as 3D modeling, online gaming, and mobile augmented reality. However, mobile devices (such as tablets and smartphones) are normally constrained by the limited resources, e.g., computation capability of local CPUs and capacity-limited battery, and thus, restrict the users to fully enjoy highly computational demanding applications on their own devices. Mobile cloud computing (MCC) has emerged as a solution to provide elastic computing power to resource-constrained devices via offloading the computational-intensive tasks to powerful distant centralized servers [3,87]. However, locating servers far away from the end-user has limitations, such as high transmission latency, communication bottlenecks, and security issues (e.g., some data should not be offloaded to servers that are

located outside of national territory) [88]. Thus, MCC is not appropriate for mobile applications that are latency or security critical [89]. This motivated the idea of moving the function of clouds towards the network edges and gave birth to a new computing infrastructure, namely MEC [6, 7]. MEC provides lower offloading latency and jitter when compared to MCC. Moreover, while MCC is a centralized computation offloading approach, MEC provides a distributed approach which makes it a suitable solution for offloading computational-intensive and time-sensitive mobile applications [90, 91].

In MEC, based on local CPU availability or energy consumption consideration, mobile devices perform task computations locally or offload them to MEC servers. This decision plays a critical role in determining the computation efficiency, especially as the task offloading requires data transmission over the wireless channel. Two main computation task offloading policies can be found in the literature, namely partial offloading and binary offloading. In the former, a portion of the computation is performed locally at the mobile device and the other portion is offloaded to MEC servers [92]. In the latter, however, the mobile device determines whether a task should be computed locally or offloaded to MEC servers [93]. From the user's point of view, the most important requirements of task offloading are low energy consumption, low offloading error and low latency, mandating ultra-reliable and low-latency task offloading [94]. In recent communication networks, computing servers are deployed at the edge of the network to facilitate and accelerate connection. These servers are deployed in high numbers and have limited capabilities when compared to conventional, backbone cloud servers (which earned them the nickname of cloudlets) [95, 96].

Such dense deployment of the MEC servers gives the leverage of diversity and the ability of dividing a task into sub-tasks and offloading them to multiple MEC servers coop-

eratively to further shorten the latency [97]. In this respect, two main offloading schemes can be considered, namely sequential offloading and parallel offloading. For the former, the sub-tasks are offloaded in a time-sequential manner to servers over a shared communication channel [11, 98]. For the latter, the sub-tasks are offloaded simultaneously to servers over orthogonal communication channels [99] or using non-orthogonal multiple access [100]. Moreover, two important aspects should be considered in such offloading schemes: (1) The computational capabilities and channel qualities are different for the MEC servers. A low computation latency can be provided by an MEC server with high computational capability; however, it might encounter a high communication latency and high offloading failure probability due to a poor communication link between the mobile device and the MEC server; (2) Dividing the task into sub-tasks yields a sophisticated scenario of dependency among sub-tasks. In many applications, the inter-sub-tasks dependency cannot be ignored, since it has a significant effect on the offloading and computation procedure. There are three main models of dependency among the sub-tasks, namely sequential dependency in which each sub-task depends on the output of the previous sub-task; parallel dependency in which a set of sub-tasks depends on the output of a previous sub-task; and general dependency in which a sub-task may depend on the output of one or more of the previous sub-tasks [101, 102]. Therefore, finding efficient scheduling decisions to assign a set of inter-dependent sub-tasks to a set of MEC servers is a challenging problem.

In the real world, the scheduling process is an approach through which a number of tasks are assigned to resources (servers) in order to complete the task execution process. Scheduling problems are typically formalized in terms of combinatorial optimization theory. Due to the high complexity and exponential growth search space of the scheduling problems, exhaustive search or random search methods are computationally demanding,

thus rendering them impractical. In this context, the GA was introduced as a robust global search method that searches for better solutions using a fitness score, which is obtained by evaluating an objective function without other derivative or auxiliary information [103, 104]. Other interesting scheduling methods, inherited from the graph theory, have emerged in the literature. These methods provide near optimum solutions with reduced computational complexity by avoiding the direct evaluation of the objective function in each iteration [105–107].

6.1.1 Related Work

Several research works (e.g., [99, 100, 108–113], and references therein) have proposed task offloading frameworks and algorithms to prolong the battery lifetime of the mobile devices. In [108], experimental results show that up to 50% of battery life can be preserved through remote processing of tasks. In [109] and [110], a single mobile device and single MEC server computation offloading problem was investigated. Wang *et al.* in [109] proposed partial computation offloading by jointly optimizing the consumed energy at the mobile device, offloading ratio, and computational delay. The authors in [110] investigated the optimal partial computation offloading jointly with the selection of constellation size and transmit power to optimize the consumed energy at mobile devices under latency constraint. A cooperative fog computing-based vehicular network architecture was proposed in [114], for the Internet-of-vehicles big data in a smart city. A dynamic network virtualization technique to achieve parallel computation in satellite-terrestrial networks was proposed in [115]. This technique integrates network resources and provides a cooperative parallel computation offloading model. In [116], a task scheduling approach with stochas-

tic time cost for computation offloading was proposed to minimize the maximum tolerable delay by considering both the average delay and delay jitter. In [117], a multi-layer edge computing framework was proposed to assign tasks to each layer optimally. A heterogeneous multilayer MEC framework was proposed in [118], in which tasks that cannot be timely processed at the edge are offloaded to upper layer MEC servers and cloud center. The authors aimed at minimizing the offloading latency by jointly coordinating the task assignment, computing, and transmission resources in each layer.

The scenario of multiple mobile devices sharing a single MEC server for computation offloading was investigated in [99, 100, 111–113]. You *et al.* in [99] studied the total energy consumption minimization problem by considering both orthogonal frequency-division multiple access and time-division multiple access under latency constraint. An optimal priority policy is provided, which gives priority to mobile devices according to their local computing energy consumption and channel gains. In [100], a multiuser MEC system with one server was studied, in which users can simultaneously offload their computation tasks to a multi-antenna server over the same time/frequency resources based on non-orthogonal multiple access. The authors assumed both constant computational delay and downlink transmission delay, and focused their study on the energy consumption in the uplink phase. A joint optimization of computational and radio resources, aimed at optimizing mobile devices' energy consumption under power and latency constraints, was proposed in [111]. A distributed game theoretic approach for decision making to offload computation among multiple mobile devices was proposed in [112]. The authors showed that the game always admits a Nash equilibrium, achieves superior computation offloading performance, and scales well as the number of mobile devices increases. A cooperative offloading framework in which multiple mobile devices cooperate with each other to im-

prove the computation capability of an MEC system was proposed in [113]. In all these works, a mobile device can be associated with one MEC server. However, in dense deployment of MEC servers, as envisioned in future networks, offloading a computational task to multiple nearby MEC servers can potentially improve the offloading process. In [97], a mobile device that offloads a set of independent tasks to a set of MEC servers in parallel via orthogonal sub-channels was studied. The authors aimed to minimize both mobile device's energy consumption and total tasks' execution latency. A sequential task offloading framework was proposed in [98]. In this framework, a mobile device segments a task into sub-tasks and offloads them to multiple servers in sequence. From a practical point of view, a task cannot be segmented arbitrarily and such a framework cannot handle the dependency among the sub-tasks.

This chapter introduces two frameworks to minimize both the latency and the offloading error probability. The first framework allocates a computationally-intensive to a set of MEC servers in sequential offloading scheme. The second framework tackles a more realistic scenario, in which the task consists of a set of inter-dependent sub-tasks. Parallel and sequential offloading schemes are studied, and the general dependency among sub-tasks is considered.

6.2 Task Allocation for Collision-free Sequential Offloading in MEC

A more realistic sequential task offloading scenario is tackled, in which the feedback from the servers to the device is considered. Including the feedback into the sequential task of-

floading framework imposes the possibility of transmission collision between the feedback of a given server and the uplink/downlink transmission to/of another server.

6.2.1 System Model

A delay-sensitive and computationally-intensive task ($\mathcal{T}$) is offloaded by a user equipment (UE) to a set $\mathcal{S} = \{s_i\}_{i=1}^{I}$ of I MEC servers. Each server is equipped with a CPU that helps offload and compute the UE's task. The tuple $\mathcal{T} = \{U, D, C\}$ is introduced to represent the task $\mathcal{T}$ in which U is the size of the input data (in bits), D is the output computed result (in bits), and C is the required CPU cycles. The output computed result is modeled as $D = \beta U$, where β ($\beta > 0$) is the output to input ratio of the task [119]. The number of CPU cycles C is modeled as $C = \alpha U$, where α ($\alpha > 0$) depends on the task's computational complexity [109]. A server s_i is represented by a tuple $s_i = \{Ru_i, Rd_i, f_i\}$, where Ru_i is the uplink data rate, Rd_i is the downlink data rate, and f_i is the computational speed of the CPU in the i-th server.

To select the servers' offloading sequence and obtain an ordered set of servers $\mathcal{S}^*$, a weight w_i is assigned to each server and the servers are sorted in an ascending order of w_i. A weighting scheme is considered which reflects the computational capabilities and quality of transmission links:

$$w_i = \frac{U}{Ru_i} + \frac{\alpha U}{f_i} + \frac{\beta U}{Rd_i}. \tag{6.1}$$

The task $\mathcal{T}$ can be divided into M ($M \leq I$) non-overlapping sub-tasks and distributed to servers. The task allocation vector $\boldsymbol{\eta} = [\eta_1, \eta_2, \ldots, \eta_N]^\mathsf{T}$ is defined such that $\sum_{i=1}^{N} \eta_i = 1$ and $\eta_i \geq 0$ is the portion of the task that is offloaded to the i-th server. Task partitioning causes overhead, and consequently, δ ($\delta \geq 1$) is introduced to represent the ratio of the

transmitted data size to the original task data size due to segmentation overhead. The tuple $\tau_i = \{u_i(\boldsymbol{\eta}), c_i(\boldsymbol{\eta}), d_i(\boldsymbol{\eta})\}$ is introduced to represent the sub-task that is offloaded to the i-th server in which $u_i(\boldsymbol{\eta}) = \eta_i \delta U$ is the size of the sub-task's input data (in bits), $c_i(\boldsymbol{\eta}) = \eta_i \alpha U$ is the required CPU cycles, and $d_i(\boldsymbol{\eta}) = \eta_i \beta U$ is the output computed result (in bits).

6.2.2 Communication Model

The UE offloads the sub-tasks using the entire channel bandwidth in a sequential technique. The end-to-end delay consists of two delay components: (1) *Task transmission delay*: the uplink and downlink transmission delay of the i-th sub-task can be expressed as $\mathcal{D}u_i(\boldsymbol{\eta}) = \frac{u_i(\boldsymbol{\eta})}{Ru_i}$ and $\mathcal{D}d_i(\boldsymbol{\eta}) = \frac{d_i(\boldsymbol{\eta})}{Rd_i}$, respectively. In sequential task offloading, the uplink transmission of the i-th sub-task will not start until the uplink transmission of all the previous $(i-1)$ sub-tasks is finished. In other words, the waiting time of the i-th sub-task is $\mathcal{W}_i(\boldsymbol{\eta}) = \sum_{j=1}^{i-1} \mathcal{D}u_j(\boldsymbol{\eta})$ and $\mathcal{W}_1(\boldsymbol{\eta}) = 0$. (2) *Computing delay*: The CPU in the i-th server computes the sub-task with a computational speed f_i (in cycles per second). Consequently, the computing delay in the i-th server is $\mathcal{D}c_i(\boldsymbol{\eta}) = \frac{c_i(\boldsymbol{\eta})}{f_i}$. The delay for completing the i-th sub-task can be expressed as

$$\mathcal{D}_i(\boldsymbol{\eta}) = \mathcal{W}_i(\boldsymbol{\eta}) + \mathcal{D}u_i(\boldsymbol{\eta}) + \mathcal{D}c_i(\boldsymbol{\eta}) + \mathcal{D}d_i(\boldsymbol{\eta}), \tag{6.2}$$

and the total latency for completing the task is given by

$$\mathcal{L}(\boldsymbol{\eta}) = \max_{\forall i \in \mathcal{S}} \{\mathcal{D}_i(\boldsymbol{\eta})\}. \tag{6.3}$$

The transmission failure probability of offloading the i-th sub-task can be written as

$$P_i(\boldsymbol{\eta}) = P_i^u(\boldsymbol{\eta}) + \left[1 - P_i^u(\boldsymbol{\eta})\right] P_i^d(\boldsymbol{\eta}), \tag{6.4}$$

where $P_i^u(\boldsymbol{\eta})$ is the uplink transmission error defined as [98]:

$$P_i^u(\boldsymbol{\eta}) = 1 - (1-q)^{\frac{u_i(\boldsymbol{\eta})}{\varrho_i^u}}, \tag{6.5}$$

and $P_i^d(\boldsymbol{\eta})$ is the downlink transmission error, similarly defined as:

$$P_i^d(\boldsymbol{\eta}) = 1 - (1-q)^{\frac{d_i(\boldsymbol{\eta})}{\varrho_i^d}}, \tag{6.6}$$

with ϱ_i^u and ϱ_i^d as the uplink and downlink transport block size, respectively, and q as the target block error rate. Consequently, the task offloading failure probability can be expressed as

$$\begin{aligned} P(\boldsymbol{\eta}) &= 1 - \prod_{i=1}^{N} \left(1 - P_i(\boldsymbol{\eta})\right) \\ &= 1 - (1-q)^{U \sum_{i=1}^{N} \eta_i \left(\frac{\delta}{\varrho_i^u} + \frac{\beta}{\varrho_i^d}\right)}. \end{aligned} \tag{6.7}$$

6.2.3 Problem Formulation

Offloading the task to fewer participating servers with better communication channels reduces the task offloading failure probability. On the other hand, a small number of participating servers increases the latency. Moreover, the servers with good channel quality are not always the best in terms of the computational speed. Our objective is to minimize $\mathcal{L}(\boldsymbol{\eta})$ and $P(\boldsymbol{\eta})$ simultaneously. Hence, the problem is a multi-objective optimization problem. To tackle the trade-off between $\mathcal{L}(\boldsymbol{\eta})$ and $P(\boldsymbol{\eta})$, the weighted sum method is considered. Keeping in mind that $\mathcal{L}(\boldsymbol{\eta})$ and $P(\boldsymbol{\eta})$ have different orders of magnitude and ranges, they should be transformed such that they have similar ranges [42]. The latency-reliability cost

function is defined as

$$\Psi\left(\boldsymbol{\eta}\right)=\lambda\frac{\mathcal{L}\left(\boldsymbol{\eta}\right)}{L}+\left(1-\lambda\right)\frac{P\left(\boldsymbol{\eta}\right)}{E}, \tag{6.8}$$

where

$$L=\max_{\forall i\in\mathcal{S}}\{\frac{\delta U}{Ru_i}+\frac{\alpha U}{f_i}+\frac{\beta U}{Rd_i}\}, \tag{6.9}$$

and

$$E=\max_{\forall i\in\mathcal{S}}\{1-\left(1-q\right)^{\frac{\delta U}{\varrho_i^u}}\left(1-q\right)^{\frac{\beta U}{\varrho_i^d}}\} \tag{6.10}$$

are the highest values of $\mathcal{L}\left(\boldsymbol{\eta}\right)$ and $P\left(\boldsymbol{\eta}\right)$, respectively. L and E represent the latency and error encountered in offloading the entire task to the server that causes the highest latency and to the server that causes the highest offloading error, respectively. Furthermore, $0\leq\lambda\leq1$ is the relative weight.

Let us assume that the first M servers in $\mathcal{S}^*$ contribute to the task offloading; a transmission collision occurs if at least one of the following two events happens: (1) One of the contributing servers finishes its sub-task computational and sends back the results while the task uplink offloading has not finished yet; (2) Two or more contributing servers send back the results simultaneously. To avoid such events and guarantee non-overlapping transmissions, the following constraints are introduced

$$\mathcal{D}u_1\left(\boldsymbol{\eta}\right)+\mathcal{D}c_1\left(\boldsymbol{\eta}\right)\geq\sum_{j=1}^{M}\mathcal{D}u_j\left(\boldsymbol{\eta}\right)\Rightarrow\mathcal{D}c_1\left(\boldsymbol{\eta}\right)\geq\sum_{j=2}^{M}\mathcal{D}u_j\left(\boldsymbol{\eta}\right) \tag{6.11}$$

and

$$\mathcal{D}u_i\left(\boldsymbol{\eta}\right)+\mathcal{D}c_i\left(\boldsymbol{\eta}\right)\geq\mathcal{D}c_{i-1}\left(\boldsymbol{\eta}\right)+\mathcal{D}d_{i-1}\left(\boldsymbol{\eta}\right),\forall\,2\leq i\leq M. \tag{6.12}$$

Constraint (6.11) guarantees that the first contributing server sends the feedback after all the other contributing servers finish their uplink transmissions. Constraint (6.12) guarantees that any subsequent contributing server sends its feedback after the previous server finishes the downlink transmission. A collision-free task allocation is shown in Fig. 6.1.

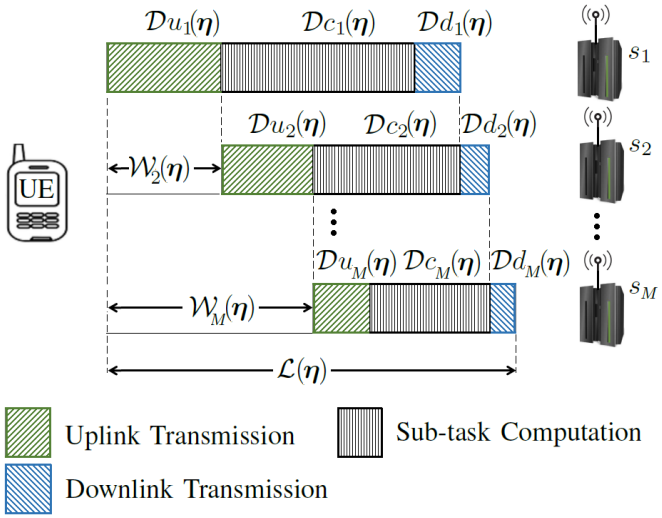

Figure 6.1: Sequential and collision-free task makespan of offloading the task to M servers.

The server contribution decision variable vector is introduced, $\boldsymbol{\gamma} = [\gamma_i]_{1\times N}$, where the binary decision variable γ_i is defined as

$$\gamma_i = \begin{cases} 1, & \text{if } s_i \text{ is contributing to the task offloading} \\ 0, & \text{otherwise.} \end{cases} \tag{6.13}$$

Consequently, the number of contributing servers $M = \sum_{i=1}^{N} \gamma_i$ and the optimization problem

is formulated as

$$\mathbf{P1} \min_{\boldsymbol{\eta},\boldsymbol{\gamma}} \quad \Psi\left(\boldsymbol{\eta}\right), \tag{6.14a}$$

$$\text{s.t.} \quad \mathcal{D}c_1\left(\boldsymbol{\eta}\right) \geq \sum_{j=2}^{N} \mathcal{D}u_j\left(\boldsymbol{\eta}\right), \tag{6.14b}$$

$$\mathcal{D}u_i\left(\boldsymbol{\eta}\right) + \mathcal{D}c_i\left(\boldsymbol{\eta}\right) \geq \gamma_i \left(\mathcal{D}u_{i-1}\left(\boldsymbol{\eta}\right) + \mathcal{D}c_{i-1}\left(\boldsymbol{\eta}\right)\right), \tag{6.14c}$$

$$\forall\, 2 \leq i \leq N,$$

$$\eta_i \leq \gamma_i \leq \eta_i U,\ \forall s_i \in \mathcal{S}^*, \tag{6.14d}$$

$$\gamma_i \geq \gamma_{i+1},\ \forall 1 \leq i \leq N-1, \tag{6.14e}$$

$$\sum_{i=1}^{N} \eta_i = 1, \tag{6.14f}$$

$$\eta_i \geq 0,\ \gamma_i \in \{0,1\}\ \forall s_i \in \mathcal{S}^*. \tag{6.14g}$$

Constraints (6.14b) and (6.14c) guarantee collision-free task offloading. In (6.14c), if server s_i is contributing to task offloading, i.e., $\gamma_i = 1$, the size of sub-task τ_i should be chosen such that (6.12) is satisfied to guarantee non-overlapping transmissions. Constraint (6.14d) ensures that if server s_i receives no sub-task, then it should not be selected as a contributing server. It also guarantees that $\gamma_i = 1$ only if server s_i is contributing to the task offloading. Constraint (6.14e) guarantees the offloading sequence and server priority by ensuring that the allowable $\boldsymbol{\gamma}$ vectors can never have $\gamma_i = 0$ and $\gamma_{i+1} = 1$ for any two consecutive elements. Therefore, $\boldsymbol{\gamma}$ guarantees that the first M servers in $\mathcal{S}^*$ contribute in the offloading. Consequently, only N feasible possibilities of $\boldsymbol{\gamma}$, i.e., $\boldsymbol{\gamma}^1 = [1,0,0,...0]^T$, $\boldsymbol{\gamma}^2 = [1,1,0,0,...0]^T$ up to $\boldsymbol{\gamma}^N = [1,1,1,1,...1]^T$. Constraints (6.14f) and (6.14g) guarantee the offloading of the whole task. The optimization problem in (6.14) is a non-convex mixed integer non-linear program which cannot be directly solved by the convex optimiza-

tion techniques. The next section introduces an optimal solution to the problem (6.14).

6.2.4 Solution Approaches

6.2.4.1 Benchmark Optimal Solution Algorithm

In this section, an exact solution algorithm is introduced to find an optimal solution for (6.14). For a given fixed $\boldsymbol{\gamma} = \hat{\boldsymbol{\gamma}}$, it is important to note that:

$$\mathcal{L}(\boldsymbol{\eta}) = \mathcal{D}_{\hat{M}}(\boldsymbol{\eta}), \tag{6.15}$$

where $\hat{M} = \sum_{j=1}^{N} \hat{\gamma}_j$, and hence, for iterations $1 \leq i \leq N$, the following *non-linear program* (NLP) is solved:

$$\mathbf{P2} \min_{\boldsymbol{\eta}} \quad \lambda \frac{\mathcal{D}_{\hat{M}}(\boldsymbol{\eta})}{L} + (1-\lambda)\frac{P(\boldsymbol{\eta})}{E}, \tag{6.16a}$$

$$\text{s.t.} \quad \text{(6.14b), (6.14c), (6.14d), (6.14f) and (6.14g).} \tag{6.16b}$$

It is straightforward to show that $P(\boldsymbol{\eta})$ in (6.16a) is concave, since it can be re-written in the form $1 - \exp\left[\mathbf{a}^T\boldsymbol{\eta}\right]$, where $\mathbf{a} = [a_i]_{1\times N}$ with $a_i = \ln(1-q)\, U\left(\frac{\delta}{\varrho_i^u} + \frac{\beta}{\varrho_i^d}\right)$. Since $\exp(.)$ is known to be a convex function, then a composition with an affine mapping is also convex, i.e., $\exp\left[\mathbf{a}^T\boldsymbol{\eta}\right]$ is convex [120, Section 3.2.2] and following this directly, $P(\boldsymbol{\eta})$ is concave. The linear term $\lambda\frac{\mathcal{D}_{\hat{M}}(\boldsymbol{\eta})}{L}$ is both concave and convex, while $(1-\lambda)\frac{P(\boldsymbol{\eta})}{E}$ is concave; hence, the sum of the two terms is concave in $\boldsymbol{\eta}$.

The constraint sets (6.14b), (6.14c), (6.14d), (6.14f) and (6.14g) in (6.16) are linear and yield a polyhedron feasible region. Consequently, the optimal solution lies in one of the extreme points (vertices) of the polyhedron [121, Section 7.8]. Algorithm 11 is designed to obtain the optimal solution of (6.14), in which the main steps are described as follows. The decision vector $\boldsymbol{\gamma}$ is set in each iteration i to $\boldsymbol{\gamma}^i = \sum_{j=1}^{i} \mathbf{e}_j$, where $\mathbf{e}_j$ is an N-element vector

whose j-th element is the only non-zero element, which equals 1. In the i-th iteration, the feasible set of solutions for $\boldsymbol{\eta}$ is a polyhedron $\mathcal{P}_i$ whose set of vertices $\mathcal{V}_i$ are all enumerated using primal-dual polytope method [122]. For each vertex $\mathbf{v}_i^k \in \mathcal{V}_i, \quad k = 1, 2, \ldots, |\mathcal{V}_i|$ (where $|.|$ is the cardinality of a set), the corresponding task allocation vector is obtained as $\boldsymbol{\eta} = \left[\mathbf{v}_i^{kT}, \mathbf{0}_{1\times(N-i)}\right]^T$ and the objective function is evaluated. The algorithm terminates after examining the vertices of N polyhedrons and returns $(\boldsymbol{\eta}^*, \boldsymbol{\gamma}^*)$ optimal for (6.14). It is worth mentioning that identifying all vertices of a polyhedron is NP hard [123].

Algorithm 11 Optimum Solution Algorithm.

Input: U, N, $\mathcal{S}$, δ, α, β, and λ;
Calculate L and E using (6.31) and (6.10), respectively;
Obtain $\mathcal{S}^*$ by sorting s_i in $\mathcal{S}$ in an ascending order of corresponding w_i;
$\Psi^* \leftarrow +\infty$; $\boldsymbol{\gamma}^0 \leftarrow \left[\mathbf{0}_{1\times N}\right]^T$;
for $i = 1$ to N **do**
  $\boldsymbol{\gamma}^i = \boldsymbol{\gamma}^{i-1} + \mathbf{e}_i$;
  $\mathcal{V}_i \leftarrow$ enumerates the vertices of polyhedron of constraints (6.14b), (6.14c), (6.14d), (6.14f) and (6.14g);
  for $k = 1$ to $|\mathcal{V}_i|$ **do**
    $\boldsymbol{\eta} = \left[\mathbf{v}_i^k, \mathbf{0}_{1\times(N-i)}\right]^T$;
    Evaluate $\Psi(\boldsymbol{\eta})$;
    if $\Psi^* > \Psi(\boldsymbol{\eta})$
      $\Psi^* \leftarrow \Psi(\boldsymbol{\eta})$; $\boldsymbol{\eta}^* \leftarrow \boldsymbol{\eta}$; $\boldsymbol{\gamma}^* \leftarrow \boldsymbol{\gamma^i}$;
    end if
  end for
end for
Return Ψ^*, $\boldsymbol{\eta}^*$, and $\boldsymbol{\gamma}^*$.

The next section introduces a more computationally-efficient heuristic solution to the problem (6.14).

6.2.4.2 Heuristic Solution

In this section, a sub-optimal solution is proposed to solve (6.14). Our strategy is described as follows. For a given $\mathcal{S}^*$ and M contributing servers, the collision-free task allocation that provides minimum latency can be achieved by replacing the inequalities in (6.11) and (6.12) by equalities. After simple algebraic manipulations, the following set of equations can be obtained

$$\begin{aligned}
\eta_1 \frac{\alpha}{f_1} &= \eta_2 \frac{\delta}{Ru_2} + \sum_{j=3}^{M} \eta_j \frac{\delta}{Ru_j}, \\
\eta_2 \frac{\alpha}{f_2} &= \sum_{j=3}^{M} \eta_j \frac{\delta}{Ru_j} + \eta_1 \frac{\beta}{Rd_1}, \\
&\cdots \\
\eta_M \frac{\alpha}{f_M} &= \sum_{j=1}^{M-1} \eta_j \frac{\beta}{Rd_j}.
\end{aligned} \tag{6.17}$$

Further straightforward manipulations of (6.17) yield

$$\begin{aligned}
\left(\frac{\alpha}{f_1} + \frac{\beta}{Rd_1}\right) \eta_1 &= \left(\frac{\delta}{Ru_2} + \frac{\alpha}{f_2}\right) \eta_2, \\
\left(\frac{\alpha}{f_2} + \frac{\beta}{Rd_2}\right) \eta_2 &= \left(\frac{\delta}{Ru_3} + \frac{\alpha}{f_3}\right) \eta_3, \\
&\cdots, \\
\left(\frac{\alpha}{f_{M-1}} + \frac{\beta}{Rd_{M-1}}\right) \eta_{M-1} &= \left(\frac{\delta}{Ru_M} + \frac{\alpha}{f_M}\right) \eta_M.
\end{aligned} \tag{6.18}$$

Based on (6.18), (6.14f), and (6.14g), the sub-optimal collision-free task allocation results as:

$$\eta_i^* = \begin{cases} \left(1 + \sum_{i=2}^{M} \frac{\prod_{j=1}^{i-1}\left(\frac{\alpha}{f_j}+\frac{\beta}{Rd_j}\right)}{\prod_{j=2}^{i}\left(\frac{\delta}{Ru_j}+\frac{\alpha}{f_j}\right)}\right)^{-1}, & i = 1 \\ \frac{\prod_{j=1}^{i-1}\left(\frac{\alpha}{f_j}+\frac{\beta}{Rd_j}\right)}{\prod_{j=2}^{i}\left(\frac{\delta}{Ru_j}+\frac{\alpha}{f_j}\right)}\eta_1^*, & 2 \leq i \leq M \\ 0, & M < i \leq N. \end{cases} \tag{6.19}$$

To tackle the latency-reliability trade-off, the number of contributing servers M varies from 1 to N. Using the sorted servers and (6.19), Algorithm 12 finds $\boldsymbol{\eta}$ that minimizes the latency-reliability cost function heuristically.

Algorithm 12 Heuristic Solution Algorithm.

Input: U, N, $\mathcal{S}$, δ, α, β, and λ;

Calculate L and E using (6.31) and (6.10), respectively;

Obtain $\mathcal{S}^*$ by sorting s_i in $\mathcal{S}$ in an ascending order of corresponding w_i;

$\Psi^* \leftarrow +\infty$;

$\boldsymbol{\gamma}^0 \leftarrow \left[\mathbf{0}_{1\times N}\right]^T$;

for $M = 2$ to N **do**

$\boldsymbol{\gamma}^M = \boldsymbol{\gamma}^{M-1} + \mathbf{e}_M$;

Calculate $\boldsymbol{\eta}$ according to (6.19);

Evaluate $\Psi(\boldsymbol{\eta})$;

if $\Psi^* > \Psi(\boldsymbol{\eta})$ and (6.14d) satisfied

$\Psi^* \leftarrow \Psi(\boldsymbol{\eta})$; $\boldsymbol{\eta}^* \leftarrow \boldsymbol{\eta}$; $\boldsymbol{\gamma}^* \leftarrow \boldsymbol{\gamma}^M$;

end if

end for

Return Ψ^*, $\boldsymbol{\eta}^*$ and $\boldsymbol{\gamma}^*$.

Calculating $\boldsymbol{\eta}$ according to (6.19) requires $\mathcal{O}\left(N^2\right)$ product operations and evaluating the objective function requires $\mathcal{O}(N)$ product operations in (6.34) and $\mathcal{O}(N)$ **max(.)** op-

erations in (6.3). Consequently, the computational complexity of finding a solution using Algorithm 12 is $\mathcal{O}\left(N^2\right)$ product and **max(.)** operations. Therefore, this algorithm is suitable for online implementation.

6.2.5 Simulation Results

In this section, the LTE system configuration is considered with 20 MHz channel bandwidth and 100 resource blocks. It is assumed that the SNR at the servers and the UE is uniformly distributed in the interval $[0, 30]$ dB. The modulation and coding scheme (MCS) is adjusted dynamically to guarantee that q does not exceed 10^{-7} and the transport block size in each link is calculated based on the SNR-MCS mapping [98, 124]. The computational speed of the servers is uniformly distributed in the interval $\left[1 \times 10^9, 10 \times 10^9\right]$ cycles per second. The task size U is set to 1 Mbits, $\delta = 1$, $\alpha = 1900/8$ cycles per bit, and $\beta = 0.2$. These parameters and $\lambda = 0.5$ are considered in the following results, unless otherwise stated.

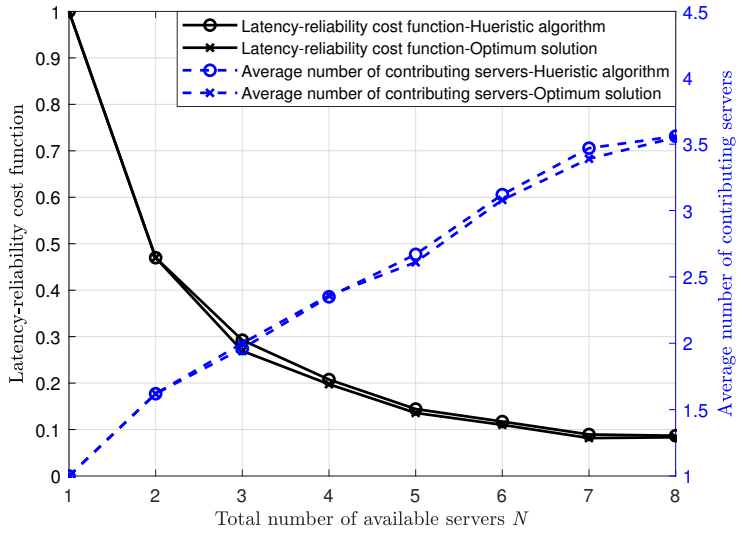

Figure 6.2: The effect of the number of available servers N.

Figure 6.2 illustrates the performance of the optimal and sub-optimal solutions versus the number of available servers N. It is seen that the latency-reliability cost function decreases as N increases and the proposed sub-optimal solution achieves near-optimal performance. It is clear that the number of the contributing servers increases as the the number of available servers increases.

The latency and offloading failure probability of both weighted sum with $\lambda = 0.5$ and latency-reliability product cost functions are illustrated in Fig. 6.3. It is clear that for $\lambda = 0.5$, the weighted sum achieves lower latency. On the other hand, the latency-reliability product cost function achieves lower offloading failure probability.

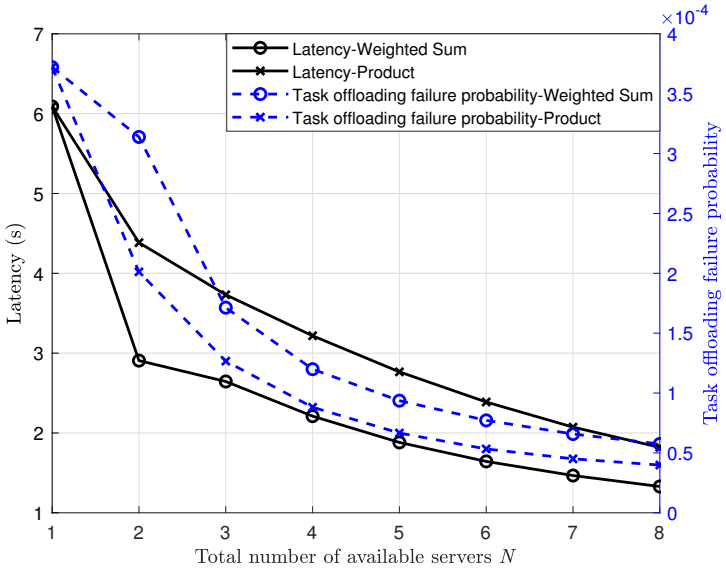

Figure 6.3: Latency and offloading failure probability of weighted sum with $\lambda = 0.5$ and latency-reliability product cost functions.

6.3 Scheduling for Mobile Edge Computing with Inter-task Dependency

6.3.1 System Model and Offloading Schemes

A delay-sensitive and computationally-intensive task ($\mathcal{T}$) is offloaded by a mobile device to a set $\mathcal{S} = \{s_i\}_{i=1}^{I}$ of I MEC servers. The packet error rate (PER) of the uplink and downlink of server s_i are p_i and q_i, respectively. The size of the input data and output computed result of $\mathcal{T}$ are U and D packets, respectively, each packet of N_p bits. The task $\mathcal{T}$ consists of J $(J \leq I)$ sub-tasks $\mathcal{T} = \{\tau_j\}_{j=1}^{J}$. The tuple $\tau_j = \{u_j, c_j, d_j\}$ is introduced to represent the j-th sub-task in which u_j is the input data size (in packets), c_j is the number of CPU cycles that is required to process the sub-task, and d_j is the output computed result (in packets). The number of CPU cycles c_j is modeled as $c_j = \alpha_j N_p u_j$, where α_j (in cycles per bit) depends on the computational complexity of the sub-task. The uplink transmission delay and the computation delay of offloading the sub-task τ_j to the i-th server are $du_{ij} = \frac{u_j N_p}{R u_i}$ and $dc_{ij} = \frac{c_j}{f_i}$, respectively. The offloading decision $\boldsymbol{\mu} = \left[\mu_{ij}\right]_{I \times J}$ is defined such that:

$$\mu_{ij} = \begin{cases} 1, & \text{if } \tau_j \text{ is assigned to } s_i, \\ 0, & \text{otherwise.} \end{cases} \tag{6.20}$$

Based on the offloading decision $\boldsymbol{\mu}$, the uplink transmission delay and the computation delay of the $j-$th sub-task τ_j can be expressed as $\mathcal{D}u_j(\boldsymbol{\mu}) = \sum_{i=1}^{I} \mu_{ij} du_{ij}$ and $\mathcal{D}c_j(\boldsymbol{\mu}) = \sum_{i=1}^{I} \mu_{ij} dc_{ij}$, respectively.

The dependency among the sub-tasks cannot be ignored in many applications, as it has significant effect on the offloading and computation procedure. the general dependency

model is considered, in which the computation of a sub-task τ_l may depend on the output computed result of one or more of the previous sub-tasks [101, 102]. To address the inter-dependency among the sub-tasks, the sub-task dependency matrix $\mathbf{x} = \left[x_{lj}\right]_{J\times J}$ is introduced such that:

$$x_{lj} = \begin{cases} 1, & \text{if } \tau_j \text{ depends on the result of } \tau_l \ (l < j) \\ 0, & \text{otherwise.} \end{cases} \tag{6.21}$$

The backhaul links among the servers are considered identical, reliable, and very high speed links. Accordingly, the transmission delay and failure probability of exchanging the intermediate results among servers is negligible.

6.3.1.1 Parallel Offloading Scheme

In the parallel offloading scheme, the mobile device offloads the sub-tasks to servers simultaneously via orthogonal sub-channels. The scheduling problem of this offloading scheme consists of assigning sub-tasks to servers under the following constraints:

- Each sub-task can be offloaded to only one server.
- Each server can handle only one sub-task.
- If $x_{lj} = 1$, then the sub-task τ_j can be offloaded to a server; however, the server holds the computing of τ_j until the computation of τ_l is finished.

A sub-task τ_j ($j > 1$) may depend on one or more of the previous $(j-1)$ sub-tasks; as such, the server holds the computing of τ_j until the computation of all sub-tasks with $x_{lj} = 1$ in $\mathbf{x}$ is finished. The holding delay of the j-th sub-task can be expressed as:

$$\mathcal{D}hp_j\left(\boldsymbol{\mu}\right) = \max_{\forall l<j} \left\{ x_{lj} \left[\left(\mathcal{D}u_l\left(\boldsymbol{\mu}\right) + \mathcal{D}hp_l\left(\boldsymbol{\mu}\right) + \mathcal{D}c_l\left(\boldsymbol{\mu}\right) \right) - \mathcal{D}u_j\left(\boldsymbol{\mu}\right) \right]^+ \right\}, \tag{6.22}$$

where $[v]^+ \triangleq \max\{v, 0\}$. The downlink transmission delay of offloading the sub-task τ_j to the $i-$th server is $dd_{ij} = \frac{d_j N_p}{R d_i}$. Based on the scheduler decision $\boldsymbol{\mu}$, the downlink transmission delay of the $j-$th sub-task τ_j can be expressed as $\mathcal{D}d_j(\boldsymbol{\mu}) = \sum_{i=1}^{I} \mu_{ij} dd_{ij}$. The delay of the parallel offloading and computing the j-th sub-task can be expressed as

$$\mathcal{D}p_j(\boldsymbol{\mu}) = \mathcal{D}u_j(\boldsymbol{\mu}) + \mathcal{D}c_j(\boldsymbol{\mu}) + \mathcal{D}hp_j(\boldsymbol{\mu}) + \mathcal{D}d_j(\boldsymbol{\mu}). \tag{6.23}$$

The total latency of completing the task with parallel offloading is given by

$$\mathcal{L}p(\boldsymbol{\mu}) = \max_{j \in \mathcal{T}} \{\mathcal{D}p_j(\boldsymbol{\mu})\}. \tag{6.24}$$

The uplink transmission failure probability and downlink transmission failure probability of offloading the j-th sub-task under the offloading decision $\boldsymbol{\mu}$ can be written as

$$P_j^u(\boldsymbol{\mu}) = 1 - \prod_{i=1}^{I} (1 - p_i)^{\mu_{ij} u_j}, \tag{6.25}$$

and

$$P_j^d(\boldsymbol{\mu}) = 1 - \prod_{i=1}^{I} (1 - q_i)^{\mu_{ij} d_j}, \tag{6.26}$$

respectively. The failure probability of offloading the j-th sub-task can be written as

$$P_j(\boldsymbol{\mu}) = P_j^u + \left[1 - P_j^u\right] P_j^d. \tag{6.27}$$

Consequently, the failure probability of offloading task $\mathcal{T}$ with the parallel offloading scheme can be expressed as

$$P_p(\boldsymbol{\mu}) = 1 - \prod_{j=1}^{J} \left(1 - P_j(\boldsymbol{\mu})\right). \tag{6.28}$$

6.3.1.2 Sequential Offloading Scheme

In the sequential offloading scheme, the mobile device offloads the sub-tasks to the servers in a time-sequential manner via a shared channel. The scheduling problem of this offloading scheme consists of assigning sub-tasks to servers under the following constraints:

- Each sub-task can be offloaded to only one server.
- Each server can handle (receive or compute) only one sub-task at any time instant.
- If $x_{lj} = 1$, then the sub-task τ_j can be offloaded to a server; however, the server holds the computing of τ_j until the computation of τ_l is finished.

In the sequential task offloading, the uplink transmission of the j-th sub-task will not start until uplink offloading of all previous $(j-1)$ sub-tasks is finished. In other words, the waiting time of the j-th sub-task before transmission is $\mathcal{W}_j(\boldsymbol{\mu}) = \sum_{l=1}^{j-1} \mathcal{D}u_l(\boldsymbol{\mu}), 2 \leq j \leq J$, and $\mathcal{W}_1(\boldsymbol{\mu}) = 0$. A sub-task τ_j $(j > 1)$ may depend on one or more of the previous $(j-1)$ sub-tasks. Since the server holds the computing of τ_j until the computation of all sub-tasks with $x_{lj} = 1$ in $\mathbf{x}$ is finished, the additional holding delay of the j-th sub-task with sequential offloading can be expressed as:

$$\mathcal{D}hs_j(\boldsymbol{\mu}) = \max_{\forall l<j} \left\{ x_{lj} \left[\mathcal{D}hs_l(\boldsymbol{\mu}) + \mathcal{D}c_l(\boldsymbol{\mu}) - \sum_{r=l+1}^{j} \mathcal{D}u_r(\boldsymbol{\mu}) \right]^+ \right\}. \tag{6.29}$$

The delay for offloading and computing the j-th sub-task can be expressed as

$$\mathcal{D}s_j(\boldsymbol{\mu}) = \mathcal{W}_j(\boldsymbol{\mu}) + \mathcal{D}u_j(\boldsymbol{\mu}) + \mathcal{D}c_j(\boldsymbol{\mu}) + \mathcal{D}hs_j(\boldsymbol{\mu}). \tag{6.30}$$

One server s_k is selected as a *sink server* to collect and send back the results to the mobile device. The total latency for completing the task is given by

$$\mathcal{L}s(\boldsymbol{\mu}) = \max_{j \in \mathcal{T}} \{\mathcal{D}s_j(\boldsymbol{\mu})\} + \mathcal{D}_{D_k}, \tag{6.31}$$

where $\mathcal{D}_{D_k} = \frac{DN_p}{Rd_k}$ is the downlink transmission delay of the resulted data D by the sink server s_k.

The failure probability of the uplink transmission of task $\mathcal{T}$ can be written as

$$P^u(\boldsymbol{\mu}) = 1 - \prod_{j=1}^{J} \left(1 - P_j^u(\boldsymbol{\mu})\right). \tag{6.32}$$

The failure probability of the downlink transmission of task $\mathcal{T}$ can be written as

$$P^d = 1 - (1 - q_k)^D, \tag{6.33}$$

where the $k-$th server is selected as the sink server. Consequently, the task offloading failure probability with sequential offloading scheme can be expressed as

$$P_s(\boldsymbol{\mu}) = P^u(\boldsymbol{\mu}) + \left[1 - P^u(\boldsymbol{\mu})\right] P^d. \tag{6.34}$$

6.3.2 Problem Formulations

6.3.2.1 Problem Formulation for Parallel Offloading Scheme

The objective is to minimize both the total latency for completing a task and the offloading failure probability simultaneously. To tackle the trade-off between latency and reliability and to give them a similar significance, the weighted product method is considered [125], in which the latency-reliability cost function of the parallel offloading scheme can be formulated as $\Psi_p(\boldsymbol{\mu}) = \mathcal{L}_p(\boldsymbol{\mu}) P_p(\boldsymbol{\mu})$.[1]

Consequently, the optimization problem is formulated as shown in (6.35). Constraints (6.35b) and (6.35c) guarantee that each sub-task is offloaded to only one server, and no two sub-tasks are offloaded to the same server, respectively. In other words, at most one sub-task is offloaded to each server.

[1]For error-free environment, the cost function is $\Psi_p(\boldsymbol{\mu}) = \mathcal{L}_p(\boldsymbol{\mu})$.

$$\textbf{P1} \min_{\boldsymbol{\mu}} \ \Psi_p(\boldsymbol{\mu}), \tag{6.35a}$$

$$\text{s.t.} \quad \sum_{i=1}^{I} \mu_{ij} = 1, \ \forall j \in \mathcal{T}, \tag{6.35b}$$

$$\sum_{j=1}^{J} \mu_{ij} \leq 1, \ \forall i \in \mathcal{S}, \tag{6.35c}$$

$$\mu_{ij} \in \{0,1\}, \ \forall i \in \mathcal{S} \text{ and } j \in \mathcal{T}. \tag{6.35d}$$

6.3.2.2 Problem Formulation for Sequential Offloading Scheme

The latency-reliability cost function of the sequential offloading scheme can be formulated as $\Psi_s(\boldsymbol{\mu}) = \mathcal{L}_s(\boldsymbol{\mu}) P_s(\boldsymbol{\mu})$.[2] The optimization problem is formulated as

$$\textbf{P2} \min_{\boldsymbol{\mu}} \ \Psi_s(\boldsymbol{\mu}), \tag{6.36a}$$

$$\text{s.t.} \quad \sum_{i=1}^{I} \mu_{ij} = 1, \ \forall j \in \mathcal{T}, \tag{6.36b}$$

$$\mu_{i(j-1)} + \mu_{ij} \leq 1, \ \forall i \in \mathcal{S}, \ 2 \leq j \leq J, \tag{6.36c}$$

$$\mu_{ij}\mu_{il} \left[\sum_{r=j+1}^{l-1} \mathcal{D}u_r(\boldsymbol{\mu}) - \left(\mathcal{D}c_j(\boldsymbol{\mu}) + \mathcal{D}hs_j(\boldsymbol{\mu}) \right) \right] \geq 0, \tag{6.36d}$$

$$\forall i \in \mathcal{S}, \ 1 \leq j \leq J-2, j+2 \leq l \leq J,$$

$$\mu_{ij} \in \{0,1\}, \ \forall i \in \mathcal{S} \text{ and } j \in \mathcal{T}. \tag{6.36e}$$

Constraint (6.36b) guarantees that each sub-task is offloaded to only one server. Furthermore, constraints (6.36c) and (6.36d) guarantee that no two successive sub-tasks are offloaded to the same server and no sub-task is offloaded to a server that still handles another

[2]For error-free environment, the cost function is $\Psi_s(\boldsymbol{\mu}) = \mathcal{L}_s(\boldsymbol{\mu})$.

sub-task, respectively. For the sequential offloading scheme, the sink server s_k can be selected such that

$$s_k = \arg\min_{i \in \mathcal{S}} \{\mathcal{D}_{Di} \left(1 - (1 - q_i)^D\right)\}, \tag{6.37}$$

and for error free environment, $s_k = \arg\min_{i \in \mathcal{S}} \{\mathcal{D}_{Di}\}$.

It is worth noting that the size of the search space for the optimization problems in (6.35) and (6.36) is 2^{IJ}. Consequently, the complexity of finding the optimum solution using exhaustive search is prohibitive for a reasonable number of servers and sub-tasks. The next two sections present more efficient methods to solve the problems using GA and conflict graph models, respectively.

6.3.3 Proposed Solution Approaches

6.3.3.1 Genetic Algorithm Approach

In the proposed GA, a potential offloading decision is represented as a set of J parameters known as genes g_i, each gene representing an association of a sub-task and a server. These genes are joined together to form a string of integers known as a chromosome $\mathbf{g}$. A chromosome (J-dimensional vector of integer numbers) identifies the servers, as assigned to the vector elements represented by the sub-tasks.

For the parallel offloading scheme, the chromosome representation described above ensures that constraints (6.35b) and (6.35d) are automatically satisfied since each sub-task is associated with exactly one server. However, this representation does not guarantee that constraint (6.35c) is fulfilled. A given chromosome is feasible for the parallel offloading scheme if its genes (servers) are distinct. In other words, if the number of the contributing servers is J. The feasibility indicator of a given chromosome for parallel offloading $\mathbf{g}_k$ is

defined as

$$\zeta_{p_k} = \begin{cases} 1, & \text{if } \mid \bigcup_{i=1}^{J} g_i \mid = J, \\ 0, & \text{otherwise.} \end{cases} \tag{6.38}$$

It is clear that $\zeta_{p_k} = 0$ if and only if the chromosome $\mathbf{g}_k$ violates constraint (6.35c). The raw fitness of a chromosome $\mathbf{g}_k$ with parallel offloading scheme is defined as

$$\theta_{p_k} = \frac{1}{\Psi_p\left(\boldsymbol{\mu}_k\right)}, \tag{6.39}$$

where $\boldsymbol{\mu}_k$ is the offloading decision associated with the chromosome $\mathbf{g}_k$. The fitness of a chromosome $\mathbf{g}_k$ is then defined as

$$\Theta_{p_k} = \zeta_{p_k}\theta_{p_k}. \tag{6.40}$$

Hence, the chromosome with the highest $\Theta_{p_k} > 0$ is feasible and minimizes the objective function in (6.35).

For the sequential offloading scheme, the chromosome representation ensures that constraints (6.36b) and (6.36e) are automatically satisfied since each sub-task is associated with exactly one server. However, this representation does not guarantee that constraints (6.36c) and (6.36d) are fulfilled. Algorithm 13 returns the feasibility indicator for a given chromosome $\mathbf{g}_k$ in the sequential offloading scheme. In this algorithm, if $\xi' = 0$ then $\mathbf{g}_k$ violates (6.36c). A given chromosome is feasible for the sequential offloading scheme if Algorithm 13 returns $\zeta_{sk} = 1$.

The raw fitness of a chromosome $\mathbf{g}_k$ with sequential offloading scheme is defined as

$$\theta_{sk} = \frac{1}{\Psi_s\left(\boldsymbol{\mu}_k\right)}, \tag{6.41}$$

where $\boldsymbol{\mu}_k$ is the offloading decision associated with the chromosome $\mathbf{g}_k$. The fitness of a

Algorithm 13 Chromosome Feasibility for Sequential Offloading Scheme

input $\mathbf{g}_k$, du_{ij}, dc_{ij}, and $\mathbf{x}$
$\zeta_{sk} = 1$
$\xi' = \prod_{j=1}^{J-1} \mid g_j - g_{j+1} \mid$
if $\xi' = 0$ **then**
  $\zeta_{sk} \leftarrow 0$
  Return ζ_{sk}
else
  Obtain $\boldsymbol{\mu}_k$ associates with $\mathbf{g}_k$
  $\boldsymbol{\mu} \leftarrow \boldsymbol{\mu}_k$.
    if (6.36d) violated **then**
      $\zeta_{sk} \leftarrow 0$
      Return ζ_{sk}
    end if
end if
Return ζ_{sk}

chromosome $\mathbf{g}_k$ is defined as

$$\Theta_{sk} = \zeta_{sk}\theta_{sk}. \tag{6.42}$$

Hence, the chromosome with the highest $\Theta_{sk} > 0$ is feasible and minimizes the objective function in (6.36).

The adopted GA consists of the following steps:

1. Generate an initial population of Ξ randomly constructed chromosomes. Each chromosome in the initial population is generated by randomly associating a server with each sub-task.

2. Evaluate the fitness and the feasibility indicators of each chromosome. Two values are associated with each chromosome, namely the raw fitness value and the feasibility indicator [104].

3. Select two chromosomes as parents for reproduction. The binary tournament selection is adopted, in which two chromosomes are chosen from the population randomly and the one with the highest raw fitness value is selected as a parent [104]. The same procedure is applied to select the other parent.

4. Generate a child chromosome from the parents by first applying a *crossover* operation. One-point crossover operation is adopted, in which a crossover point $1 \leq j \leq J$ is randomly selected. The child is structured from the first j genes which are taken from one of the parents, and from the remaining $J-j$ genes which are taken from the other parent. The crossover operation is followed by a *mutation* operation, in which two randomly selected genes in the child are exchanged (i.e., exchanging assigned servers between two randomly selected sub-tasks). After generating the child, the fitness and the feasibility indicator of the child chromosome are evaluated.

5. Replace a chromosome in the population by the child chromosome. The adopted replacement procedure is as follows. The chromosome with zero feasibility indicator and lowest raw fitness value is replaced by the child. If all the chromosomes in the population are feasible, the chromosome with the lowest raw fitness value is replaced by the child.

6. Repeat steps 3 and 4 until N offspring chromosomes have been generated without enhancing the best chromosome found so far or a maximum number of offspring chromosomes M has been generated.

6.3.3.2 Conflict Graphs Solution Approach

In this section, heuristic solutions to the formulated optimization problems using techniques inherited from graph theory are introduced. To find an optimized offloading decision, a representation of all feasible offloading schedules should be first design. *Offloading conflict graph* models are proposed to represent the offloading conflicts. The offloading conflict graph is an undirected graph in which each vertex has two indices representing an association of a server and a sub-task. Each undirected edge between two vertices is an offloading conflict between the two corresponding associations.

The proposed conflict graph of the parallel offloading scheme $\mathcal{G}_p$ is constructed as follows:

Vertex Set: The vertex set consists of IJ vertices in which each vertex v_{ij} represents the offloading of sub-task τ_j to server s_i.

Scheduling Conflict Edges: Any two vertices v_{ij} and v_{kl} in $\mathcal{G}_p$ are set adjacent by a scheduling conflict edge if one of the following cases occurs:

1. $j = l \Rightarrow$ A sub-task cannot be handled by more than one server.

2. $i = k \Rightarrow$ A server cannot handle more than one sub-task.

Note that for the parallel offloading scheme, the holding time of a sub-task does not affect the feasibility of a solution. Given this configuration of the parallel offloading conflict graph, any independent set[3] of size J in $\mathcal{G}_p$ will represent a candidate feasible decision for the parallel offloading scheme. To select the offloading decision that provides minimum latency and guarantees minimum failure probability, a weight wp_{ij} is assigned to each

[3] An independent set in a graph is a set of vertices such as no edge exists between any pair of vertices in the set [126].

vertex in $\mathcal{G}_p$. This weight reflects both the latency and failure probability, being defined as

$$wp_{ij} = \left(du_{ij} + dc_{ij} + dd_{ij}\right)\left(1 - \left(1 - p_i\right)^{u_j}\left(1 - q_i\right)^{d_j}\right). \tag{6.43}$$

For error-free environment, the weight simply becomes

$$wp_{ij} = \left(du_{ij} + dc_{ij} + dd_{ij}\right). \tag{6.44}$$

The proposed conflict graph of the sequential offloading scheme $\mathcal{G}_s$ is constructed as follows:

Vertex Set: The vertex set consists of IJ vertices in which each vertex v_{ij} represents the offloading of sub-task τ_j to server s_i.

Scheduling Conflict Edges: Any two vertices v_{ij} and v_{kl} in $\mathcal{G}_s$ are set adjacent by a scheduling conflict edge if one of the three cases below occurs:

1. $j = l \Rightarrow$ A sub-task or the feedback cannot be handled by more than one server.

2. $i = k$ and $\mid j - l \mid = 1. \Rightarrow$ Two successive sub-tasks cannot be offloaded to the same server.

3. $i = k$ and $l > j + 1$ and $dc_{ij} + \widetilde{dh}_{ij} > \frac{\sum_{r=j+1}^{l-1} N_p u_r}{\max\limits_{\forall \varrho \in \mathcal{S}\setminus\{i\}} \{Ru_\varrho\}}$,
   where $\widetilde{dh}_{ij}$ is an approximation of the holding time of sub-task τ_l in server s_i, which can be expressed as

$$\widetilde{dh}_{ij} = \max_{\forall \sigma < j} \left\{ x_{\sigma j} \left[\widetilde{dh}_{i\sigma} + \mathcal{D}c_{i\sigma} - \left(\frac{\sum_{r=\sigma+1}^{j-1} N_p u_r}{\max\limits_{\forall \varrho \in \mathcal{S}\setminus\{i\}} \{Ru_\varrho\}} + \frac{N_p u_j}{Ru_i} \right) \right]^+ \right\}. \tag{6.45}$$

   $\Rightarrow$ A server cannot handle a new sub-task if it still computes another sub-task.

Given this configuration of the offloading conflicts, any independent set of size J in $\mathcal{G}_s$ will represent a candidate offloading decision. To select the offloading decision that provides minimum latency and guarantees minimum failure probability, a weight ws_{ij} is assigned to each vertex in $\mathcal{G}_s$. This weight reflects both the latency and failure probability, being defined as

$$ws_{ij} = \left(du_{ij} + dc_{ij}\right)\left(1 - (1 - p_i)^{u_j}\right). \tag{6.46}$$

For error-free environment, $ws_{ij} = \left(du_{ij} + dc_{ij}\right)$. The algorithm in the following section is designed to select the independent set that represents the offloading decision which provides minimum latency and guarantees minimum offloading failure probability.

In case of parallel or sequential offloading scheme, the following substitutions are performed $\mathcal{G} = \mathcal{G}_p$ and $w'_{ij} = wp_{ij}\ \forall v_{ij} \in \mathcal{G}_p$ or $\mathcal{G} = \mathcal{G}_s$ and $w'_{ij} = ws_{ij}\ \forall v_{ij} \in \mathcal{G}_s$, respectively. For a given sub-task τ_j, the vertex with the minimum weight $w'_{ij}\ \forall i \in \mathcal{S}$ represents the association between τ_j and the server that provides low latency and offloading failure error for offloading τ_j. Since each server has different channel state and computation speed and the association of an arbitrarily chosen sub-task with the best available server may prevent other highly demanding sub-tasks to be offloaded to the best available server, a given sub-task τ_j prioritizes to be associated with the best available server if:

- The sub-task τ_j is highly demanding (i.e., it consists of large number of packets and it requires a high number of CPU cycles). To sort the sub-tasks based on their demand, a raw prioritization weight is assigned to each sub-task such that $\Delta_j = w'_{ij}\ \forall j \in \mathcal{T}$ and i is fixed. Note that sorting the sub-tasks based on a weight calculated with respect to any $i \in \mathcal{S}$ leads to the same prioritized set of sub-tasks. This is because in each case, all sub-tasks will experience the same communication conditions and

undergo the same computation capabilities. The sub-task with the highest $\Delta_j \; \forall j \in \mathcal{T}$ is the most demanding sub-task.

- A high number of sub-tasks depend on the output computed results of τ_j (i.e., τ_j has the largest $\kappa_j = \sum_{l=1}^{J} x_{jl}\Delta_l \;\; \forall j \in \mathcal{T}$).

Using this sub-task prioritization and the definition of the vertices weights, Algorithm 14 executes iteratively a greedy minimum weighted vertex search approach to select J-vertices independent set Γ, which represents the best offloading decision. The sub-tasks are sorted in a descending order according to a prioritization weight of $\Delta_j + \kappa_j$. The index of the sorted sub-tasks is $j'(1), j'(2), j'(3), \ldots, j'(J)$ and the sub-task with $j'(1)$ has the higher priority to be associated with the best available server. Each iteration is implemented as follows: the vertex with the minimum weight in the vertices set $v^*_{ij'(l)}$ with $l = 1$ and $\forall\, 1 \leq i \leq I$ will be picked and added to Γ. The selected vertex $v^*_{ij'(l)}$ and all the vertices that are adjacent to it (symbolized in the algorithm by $\mathcal{N}_{\mathcal{G}}\left(v^*_{ij'(l)}\right)$) are eliminated from the graph $\mathcal{G}$. This elimination is performed to guarantee that the next picked vertex in the next iteration is not in scheduling conflict with the already selected ones in Γ. The algorithm continues by increasing l by one until $l = J$.

6.3.3.3 Complexity Analysis

This section introduces the computational complexity of the GA and the conflict-graph algorithms. The basic operations performed in the GA algorithm are selection operation, crossover operation, mutation operation, chromosome replacement, and fitness evaluation. The basic operations performed in the conflict-graph algorithm are generating the vertices set, building the adjacency edges, and finding the independent set.

Algorithm 14 Minimum Weighted Vertex Search Algorithm.

1: **Input**: $\mathcal{G}_p$ and $wp_{ij}\ \forall v_{ij} \in \mathcal{G}_p$ OR $\mathcal{G}_s$ and $ws_{ij}\ \forall v_{ij} \in \mathcal{G}_s$. The sub-task dependency matrix $\mathbf{x}$.

2: Initialization:

- $\mathcal{G} \leftarrow \mathcal{G}_p$ OR $\mathcal{G}_s$
- $w'_{ij} \leftarrow wp_{ij}$ OR ws_{ij}
- Calculate $\Delta_j = w'_{1j}$ and $\kappa_j = \sum_{l=1}^{J} x_{jl}\Delta_l\ \forall j \in \mathcal{T}$
- Sort the sub-tasks in a descending order according to a prioritization weight of $\Delta_j + \kappa_j$
- The index of the sorted sub-tasks is $j'(1), j'(2), j'(3) \ldots, j'(J)$
- Set the selected independent set $\Gamma = \emptyset$

3: **for** $l = 1$ to J **do**

4: $v^*_{ij'(l)} \leftarrow \min_{i \in \mathcal{S}} \{w_{ij'(l)}\}$

5: $\Gamma = \Gamma \cup v^*_{ij'(l)}$

6: $\mathcal{G} \leftarrow \mathcal{G} \setminus \left(v^*_{ij'(l)} \cup \mathcal{N}_{\mathcal{G}}(v^*_{ij'(l)})\right)$

7: **end for**

8: **Return** Γ.

In the GA algorithm, generating a child chromosome requires the following operations. The crossover process has a complexity of $\mathcal{O}(J)$, the mutation process has also a complexity of $\mathcal{O}(J)$ [127], and chromosome replacement requires $\mathcal{O}(\Xi)$ operations. Consequently, generating the offspring chromosomes requires $\mathcal{O}\left([J + \Xi] M\right)$ operations. To evaluate the fitness of a chromosome, the following operations are required. Calculating the holding time of each sub-task requires $\mathcal{O}(J)$ operations. Consequently, calculating the total latency requires $\mathcal{O}\left(J^2\right)$ operations. Calculating the offloading failure probability requires $\mathcal{O}(IJ)$ operations. The time complexity of finding the feasibility indicator ζ_{p_k} is $\mathcal{O}(J)$. Since $J \leq I$, evaluating the fitness of a given chromosome has a time complexity of $\mathcal{O}\left(J^2 + IJ + J\right) = \mathcal{O}(IJ)$. As the fitness of each chromosome in the initial population and the offspring is evaluated, the computational complexity of the GA with the parallel offloading scheme is $\mathcal{O}\left(IJ\left[\Xi + M\right] + \left[J + \Xi\right] M\right) = \mathcal{O}\left(IJ\left[\Xi + M\right] + \Xi M\right)$.

The vertex set size of the conflict-graph is $\mathcal{O}(IJ)$. To build the adjacency edges of the conflict-graph for the parallel offloading, all the vertices representing a given sub-task are connected to each other and all the vertices representing a given server are connected to each other. This means that a total of $\mathcal{O}(IJ + IJ) = \mathcal{O}(IJ)$ operations are needed to build the adjacency edges. On the other hand, the complexity of the minimum weighted vertex search algorithm can be computed as follows. Computing the weights of the vertices requires $\mathcal{O}(IJ)$ operations. Sorting the sub-tasks has a time complexity of $\mathcal{O}(J \log J)$. The time complexity of selecting the minimum weighted vertex for each sub-task is $\mathcal{O}(I)$. Consequently, the complexity of the minimum weighted vertex search algorithm is $\mathcal{O}(IJ + J \log J + IJ) = \mathcal{O}(IJ)$ and the computational complexity of the conflict-graph algorithm is $\mathcal{O}(IJ + IJ) = \mathcal{O}(IJ)$. It is worth mentioning that representing the offloading decisions as vectors of integers reduces the search space to $\mathcal{O}\left(I^J\right)$. Consequently, the computational complexity of finding the optimal solution using the exhaustive search is $\mathcal{O}\left(IJI^J\right) = \mathcal{O}\left(JI^{J+1}\right)$.

Similar to the parallel offloading case, calculating the total latency and offloading failure probability for the sequential offloading scheme require $\mathcal{O}(J^2)$ and $\mathcal{O}(IJ)$ operations, respectively. The time complexity of finding the feasibility indicator ζ_{s_k} is $\mathcal{O}\left(IJ^2\right)$. Consequently, evaluating the fitness of a given chromosome has a time complexity of $\mathcal{O}\left(J^2 + IJ + IJ^2\right) = \mathcal{O}\left(IJ^2\right)$ and the computational complexity of the GA with the parallel offloading scheme is $\mathcal{O}\left(IJ^2\left[\Xi + M\right] + \left[J + \Xi\right]M\right) = \mathcal{O}\left(IJ^2\left[\Xi + M\right] + \Xi M\right)$.

The vertex set size of the conflict-graph is $\mathcal{O}(IJ)$. To build the adjacency edges of the conflict-graph for sequential offloading, all the vertices representing a given sub-task are connected to each other, which requires a total of $\mathcal{O}(IJ)$ operations. For all vertices representing a server the following steps are performed: (1) Connect any two ver-

tices representing two successive sub-tasks, which requires $\mathcal{O}(J)$ operations; (2) Calculate $\widetilde{dh}_{ij}$ for each two vertices, which requires $\mathcal{O}(IJ)$ operations. This means that a total of $\mathcal{O}(IJ + [J + IJ]I) = \mathcal{O}(I^2J)$ operations are needed to build the adjacency edges. Consequently, the computational complexity of the conflict-graph algorithm is $\mathcal{O}(I^2J + IJ) = \mathcal{O}(I^2J)$. It is worth mentioning that the computational complexity of finding the optimal solution using the exhaustive search is $\mathcal{O}\left(IJ^2I^J\right) = \mathcal{O}\left(J^2I^{J+1}\right)$.

Table 6.1 summarizes the time complexity of the algorithms. To introduce a quantitative measure of the complexity of the algorithms, Table 6.1 shows the average execution time of each algorithm on a personal computer equipped with an Intel(R) Core(TM) i5-4570 CPU, working at a clock frequency of 3.2 GHz, and 8 GB of RAM. The algorithms have been implemented in MATLAB. The system model consists of $I = 10$ servers, $J = 10$ sub-tasks, $\Xi = 10IJ$ chromosomes [132], and $N = 1000$ chromosomes, and $M = 10^5$ chromosomes. It is clear that the conflict graph solution is more complexity-efficient than that of the GA algorithm for both parallel and sequential offloading schemes.

Table 6.1: Time complexity of the MEC scheduling algorithms.

Algorithm	**Complexity**	**Av. Execution Time**
GA-parallel offloading	$\mathcal{O}\left(IJ\left[\Xi + M\right] + \Xi M\right)$	4.4 ms
Graph-parallel offloading	$\mathcal{O}(IJ)$	1.3 ms
GA-sequential offloading	$\mathcal{O}\left(IJ^2\left[\Xi + M\right] + \Xi M\right)$	6.5 ms
Graph-sequential offloading	$\mathcal{O}(I^2J)$	1.8 ms

6.3.4 Simulation Results

This section presents simulation results to evaluate the performance of the proposed algorithms and studies the effect of the main parameters on the latency and offloading failure probability. In obtaining these results, the servers are uniformly distributed with distances from the mobile device as 200 m ~ 400 m. The channel gain is $h_i = \delta_i^{-\varepsilon}$, where δ_i is the distance between the mobile device and the i-th server and $\varepsilon = 4$ is the path loss exponent [112]. The channel bandwidth is 1 MHz and the spectral density of the additive noise is $N_0 = -173$ dBm/Hz. The transmit power of the mobile device and the servers is 100 mW [119]. For the parallel offloading scheme, orthogonal sub-channels are established each with 1 MHz bandwidth. The remaining parameters are as follows. The computation speed of the MEC server is generated randomly from a uniform distribution ranging from 2×10^9 to 8×10^9 cycles per second [97], the packet size is $N_p = 20$ bytes [128], task input and output data are $U = 1000$ packets [98, 129] and $D = 200$ packets [119], respectively, and the computational complexity of a sub-task α_j in cycles per bit follows a Gamma distribution with shape and scale parameter of 4 and 200, respectively [130, 131]. To represent the dependency among the sub-tasks, a given sub-task depends on any of the previous sub-tasks with probability π. The initial population size is $10IJ$ chromosomes [132], and N and M equal 10^3 and 10^5, respectively. The default simulation parameters, which are summarized in Table 6.2, are considered in the results, unless otherwise stated.

Figure 6.4 illustrates the performance of the GA, conflict graph, and optimum solution (obtained using exhaustive search) versus the task data size U. It is seen that the proposed algorithms achieve near-optimum performance, and the latency-reliability cost function of the sequential offloading is less than that of the parallel offloading scheme.

Table 6.2: Simulation parameters of the MEC scheduling framework.

Parameter	Value	Parameter	Value
Channel bandwidth	1 MHz	Noise spectral density N_0	-173 dBm/Hz
Server transmit power	100 mW	Mobile device transmit power	100 mW
Packet size	20 bytes	Computing speed of the servers	$[2;\ 8] \times 10^9$ cycles/s
Total task input data size U	1000 packets	Total task output data size D	200 packets
Computational complexity parameter	$\alpha_j \sim$ Gamma $(4, 200)$ cycles/bit	Sub-tasks dependency probability π	0.3
Initial population size Ξ	$10IJ$ chromosomes	N	10^3 chromosomes
M	10^5 chromosomes	U	1000 packets

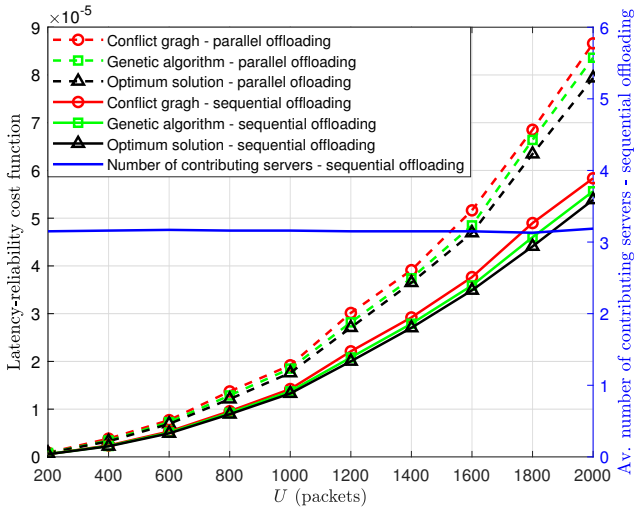

Figure 6.4: Latency-reliability cost function versus the task data size U.

To gain deep insight into this result, Figs. 6.5 and 6.6 show the corresponding latency and offloading failure probability, respectively. It is clear that the parallel offloading provides less latency; however, sequential offloading provides less offloading failure probability. For parallel offloading, all servers contribute to offloading simultaneously, which leads to less latency; however, not all of them have good channel quality, which yields higher failure probability. On the other hand, in sequential offloading, the scheduler explores the

best available servers and less servers are contributing.

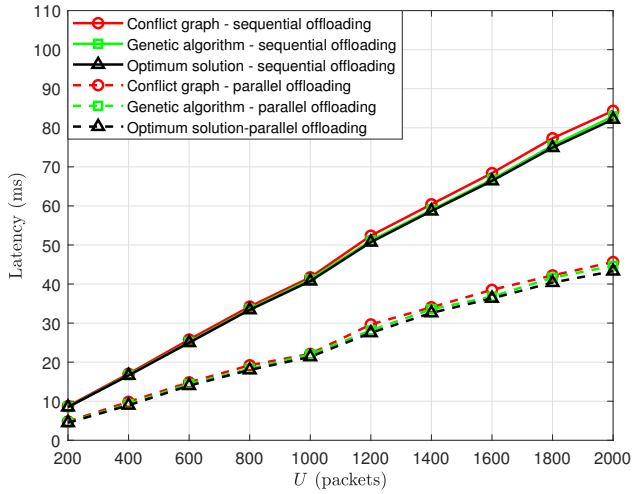

Figure 6.5: Latency versus the task data size U.

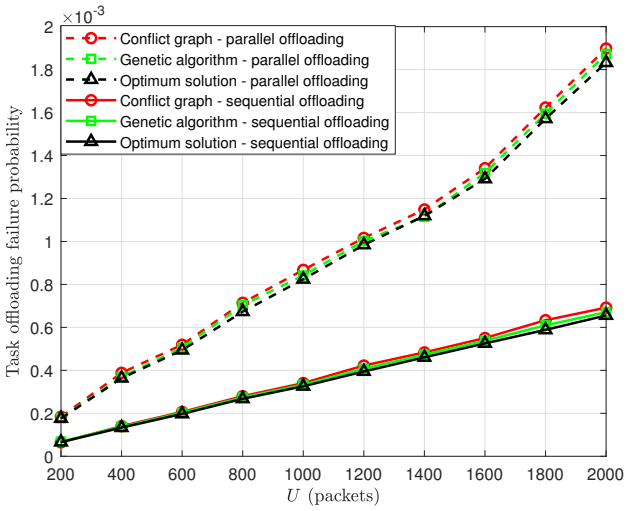

Figure 6.6: Offloading failure probability versus the task data size U.

In Fig. 6.7, all servers are located at the same distance from the mobile device (i.e., $\delta_i = \delta \ \forall s_i \in \mathcal{S}$). Such scenario is suitable for the case where servers are located in close

proximity to each other or they are placed at the arc of a circle whose center is the mobile device. It is noticed that as the distance between the servers and the mobile device increases (i.e., the SNR decreases), the latency increases in both parallel and sequential offloading schemes. However, for the sequential offloading, the latency grows more rapidly. This can be attributed to the fact that reducing the SNR will increase the uplink transmission delay, and consequently, the waiting time of the sub-tasks[4] also increases. It is also noticed that as the distance between the servers and the mobile device increases (i.e., uplink transmission delay increases), the number of contributed servers in sequential offloading decreases. This is because during the long time duration of uplink transmission of a sub-task, more servers will complete the computation of their sub-task and get ready to contribute. Such availability of servers gives the scheduler the ability to offload more sub-tasks to less number of servers (because it selects the best available servers).

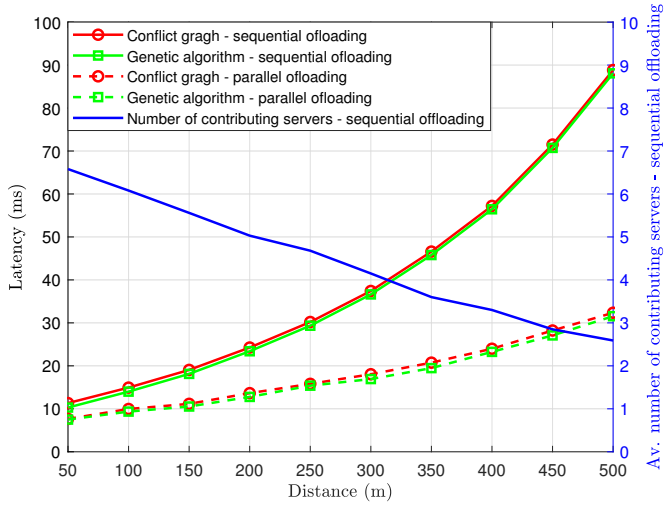

Figure 6.7: Latency versus distance with $I = 10$ servers and $J = 10$ sub-tasks.

[4]The waiting time of a sub-task is an accumulation of the uplink delay of all previous sub-tasks, i.e., $\mathcal{W}_j(\boldsymbol{\mu}) = \sum_{l=1}^{j-1} \mathcal{D}u_l(\boldsymbol{\mu})$.

6.4 Concluding Remarks

This chapter introduced a sequential task offloading scheme that guarantees collision-free offloading to multiple MEC servers. The problem was formulated as a weighted-sum multi-objective optimization problem of latency and offloading failure probability, which enables to develop an exact technique that guarantees the optimal solution. A more computationally-efficient heuristic algorithm was developed for online implementation. Simulation results illustrated that the proposed offloading scheme can effectively reduce both latency and offloading failure probability.

Task scheduling for parallel and sequential offloading of an inter-dependent set of tasks has been consider as well. The problem has been formulated as the joint optimization of latency and offloading failure probability. Heuristic solutions based on genetic algorithm and conflict graph models were developed. Simulation results revealed that the proposed solutions provide performance close to the optimal one, with reduced complexity. Also, findings showed that sequential offloading provides less offloading failure probability and requires a lower number of servers. On the other hand, parallel offloading provides less latency. However, as the dependency among sub-tasks increases, the latency response gap between sequential and parallel schemes decreases.

Chapter 7

Conclusions and Potential Directions of Future Work

This chapter summarizes the work carried out towards the PhD thesis and discusses potential future directions of investigation.

7.1 Summary

In this thesis work, energy-efficient and age-optimum frameworks have been designed for data aggregation and dissemination. Moreover, reliable and low-latency offloading frameworks for sequential and parallel MEC were developed. Both terrestrial and underwater environments were investigated. In the conventional sensor networks, a small number of devices are deployed to collect and upload the data to the reference devices. However, with the exponential growth in both number and computing capability of devices, more efficient frameworks could be exploited for data aggregation, dissemination, and processing.

This motivated me to design a device-role assignment framework, which optimizes the role of each device in the network and enables in-network data processing. More sophisticated scenarios with mobile data aggregator/disseminator scenarios were explored as well. UAVs and AUVs have been considered as mobile data aggregator(s)/disseminator(s) for terrestrial and underwater scenarios, respectively. Different metrics were studied including the overall energy consumption in the data aggregation/ dissemination systems, AoI in the data aggregation systems, and a weighted sum of the latency and offloading error in the MEC systems. A novel metric referred to as the correlation-aware AoI was also proposed to captures both the freshness and diversity in the aggregated data. Computationally efficient solution approaches were developed to find solutions for the proposed frameworks, including GAs, ACO, DRL, and conflict graphs approaches. To show the effectiveness of the developed solution approaches, their performance was compared to baseline approaches. Extensive simulations illustrated that the proposed solution approaches provide performance close to the optimal solutions, which are obtained through exhaustive search or computationally intensive methods.

7.2 Potential Directions of Future Investigation

The work presented in this thesis opens the door for future investigations, such as:

- Optimizing mobile crowd sensing using machine learning (ML)-assisted solutions: Mobile crowd sensing is a promising paradigm to enable massive sensing of the environment. It allows aggregation of a large amount of information by the devices regarding local information, traffic conditions, and device data demands. Considering transmission energy and delay, incentives can be rewarded to devices in return

of their support to improve the network's performance. Therefore, devices have to decide either to participate in the crowd sensing or not. On the other hand, network management servers have to decide incentives for the participating devices based on their partaking. The incentives of each participating device should be proportional to the benefits of its contributions (including its data novelty and/or importance). The intractability of such a problem could be tackled by developing ML-assisted solutions.

- ML-assisted models for data integrity in IoT networks: The number of devices in IoT networks increases exponentially and each device needs to be authenticated to avoid threats and to secure the device's profile that includes confidential and private information. This rises the challenge of how data caching, transmission, scheduling, and processing can be implemented without compromising the integrity. Moreover, the device's communication setup with the network requires important information including the position of the device, social and proximity relationships, and data viewing patterns. In case of lacking integrity features, the device may experience critical problems. Furthermore, storage space should be optimized using efficient method to store rarely accessed data files, or even deliberately delete such data for reclaiming. As a result, effective protection of the correctness of outsourced data is crucial and it is important to find efficient and periodic integrity verifications without a local copy of data files over a prolonged period of time. ML-assisted models could be developed to guarantee the integrity of the data in the IoT networks.

- Caching in aerial networks: Data caching represents a data storage approach such that future requests can be served faster; this data could be the result of an earlier

computation task or a copy of data stored in another place. Despite the high bandwidth of millimeter wave (mmWave), devices' mobility and high path loss restricts the potentials of mmWave networks. The mmWave signals are heavily attenuated by concrete buildings and road infrastructures. UAV-mounted storage servers with intelligent reflecting surface (IRS) capabilities represent a potential solution to provide flexible and high speed data cashing. Moreover, duplication of content placement at various cache resources is necessary to reduce the probability of cache hit misses. Since content placement is based on the prediction of data popularity, it is important to design accurate prediction models using various information such as data request, user mobility, and device's social information. Different optimization approaches could be designed to optimize the UAV trajectory, phase shifts of the IRS's elements, and the spatial reuse and dynamics control of mmWave communications.

Bibliography

[1] W. Saad, M. Bennis, and M. Chen, "A vision of 6G wireless systems: Applications, trends, technologies, and open research problems," *IEEE Network*, vol. 34, pp. 134–142, May 2020.

[2] R. D. Yates and S. K. Kaul, "The age of information: Real-time status updating by multiple sources," *IEEE Trans. Inform. Theory*, vol. 65, pp. 1807–1827, Mar. 2019.

[3] T. K. Rodrigues, K. Suto, H. Nishiyama, J. Liu, and N. Kato, "Machine learning meets computation and communication control in evolving edge and cloud: Challenges and future perspective," *IEEE Commun. Surveys Tuts*, vol. 22, pp. 38–67, Mar. 2020.

[4] M. C. Domingo, "An overview of the internet of underwater things," *J. Netw. Comput. Appl*, vol. 35, pp. 1879–1890, Nov. 2012.

[5] S. Kaul, R. Yates, and M. Gruteser, "Real-time status: How often should one update?," in *Proc. IEEE INFOCOM*, pp. 2731–2735, Orlando, FL, USA, 2012.

[6] S. Barbarossa, S. Sardellitti, and P. D. Lorenzo, "Communicating while computing: Distributed mobile cloud computing over 5G heterogeneous networks," *IEEE Signal Process. Mag.*, vol. 31, pp. 45–55, Nov. 2014.

[7] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet Things J.*, vol. 5, pp. 450–465, Feb. 2018.

[8] A. A. Al-Habob, O. A. Dobre, and H. V. Poor, "Energy-efficient spatially-correlated data aggregation using unmanned aerial vehicles," in *Proc. IEEE Int. Symp. Personal, Indoor Mobile Radio Commun.*, pp. 1–6, London, UK, 2020.

[9] A. A. Al-Habob, O. A. Dobre, and H. V. Poor, "Age- and correlation-aware information gathering," *IEEE Wireless Commun. Lett.*, vol. Early Access, pp. 1–5, 2021.

[10] A. A. Al-Habob, O. A. Dobre, S. Muhaidat, and H. V. Poor, "Energy-efficient data dissemination using a UAV: An ant colony approach," *IEEE Wireless Commun. Lett.*, vol. 10, pp. 16–20, Jan. 2021.

[11] A. A. Al-Habob, A. Ibrahim, O. A. Dobre, and A. G. Armada, "Collision-free sequential task offloading for mobile edge computing," *IEEE Commun. Lett.*, vol. 24, pp. 71–75, Jan. 2020.

[12] A. A. Al-Habob, O. A. Dobre, A. G. Armada, and S. Muhaidat, "Task scheduling for mobile edge computing using genetic algorithm and conflict graphs," *IEEE Trans. Veh. Technol.*, vol. 69, Aug. 2020.

[13] A. A. Al-habob, O. A. Dobre, and A. G. Armada, "Sequential task scheduling for mobile edge computing using genetic algorithm," in *Proc. IEEE Globecom*, pp. 1–6, Waikoloa, HI, USA, 2019.

[14] A. A. Al-Habob, O. A. Dobre, and H. V. Poor, "Role assignment for spatially-correlated data aggregation using multi-sink internet of underwater things," *IEEE Trans. Green Commun. Netw.*, vol. 5, pp. 1570–1579, Apr. 2021.

[15] A. A. Al-Habob and O. A. Dobre, "Role assignment for energy-efficient data gathering using Internet of underwater things," in *Proc. IEEE Int. Conf. Commun.*, pp. 1–6, Dublin, Ireland, 2020.

[16] A. A. Al-Habob, O. A. Dobre, and H. V. Poor, "Age-optimal information gathering in linear underwater networks: A deep reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 70, pp. 13129–13138, Dec. 2021.

[17] B. K. Wilburn, M. G. Perhinschi, and J. N. Wilburn, "A modified genetic algorithm for UAV trajectory tracking control laws optimization," *Int. J. Intell. Unmanned Syst.*, vol. 2, pp. 58–90, May 2014.

[18] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[19] S. Pattem, B. Krishnamachari, and R. Govindan, "The impact of spatial correlation on routing with compression in wireless sensor networks," *ACM Trans. Sensor Netw. (TOSN)*, vol. 4, p. 24, Aug. 2008.

[20] D. Hulens, T. Goedemé, and J. Verbeke, "How to choose the best embedded processing platform for on-board UAV image processing?," in *Proc. 15th Int. Conf. Computer Vision Theory Appl.*, pp. 1–10, Berlin, Germany, 2015.

[21] M. B. Ghorbel, D. Rodríguez-Duarte, H. Ghazzai, M. J. Hossain, and H. Menouar, "Joint position and travel path optimization for energy efficient wireless data gathering using unmanned aerial vehicles," *IEEE Trans. Veh. Technol.*, vol. 68, pp. 2165–2175, Mar. 2019.

[22] A. Al-Hourani, S. Kandeepan, and S. Lardner, "Optimal LAP altitude for maximum coverage," *IEEE Wireless Commun. Lett.*, vol. 3, pp. 569–572, Dec. 2014.

[23] M. Alzenad, A. El-Keyi, F. Lagum, and H. Yanikomeroglu, "3-D placement of an unmanned aerial vehicle base station (UAV-BS) for energy-efficient maximal coverage," *IEEE Wireless Commun. Lett.*, vol. 6, pp. 434–437, Aug. 2017.

[24] M. Stojanovic, "On the relationship between capacity and distance in an underwater acoustic communication channel," *SIGMOBILE*, vol. 11, pp. 34–43, Oct. 2007.

[25] L. M. Brekhovskikh, Y. P. Lysanov, and Y. Lysanov, *Fundamentals of Ocean Acoustics*. Springer, 3 ed., 2004.

[26] R. J. Urick, *Principles of Underwater Sound*. New York, NY, USA: McGraw-Hill, 1983.

[27] M. Zorzi, P. Casari, N. Baldo, and A. F. Harris, "Energy-efficient routing schemes for underwater acoustic networks," *IEEE J. Sel. Areas Commun.*, vol. 26, pp. 1754–1766, Dec. 2008.

[28] S. A. Dehkordi, K. Farajzadeh, J. Rezazadeh, R. Farahbakhsh, K. Sandrasegaran, and M. A. Dehkordi, "A survey on data aggregation techniques in IoT sensor networks," *Wireless Net.*, vol. 26, pp. 1243–1263, Feb. 2020.

[29] J. Zhang, P. Hu, F. Xie, J. Long, and A. He, "An energy efficient and reliable in-network data aggregation scheme for WSN," *IEEE Access*, vol. 6, pp. 71857–71870, Nov. 2018.

[30] M. Huang, A. Liu, T. Wang, and C. Huang, "Green data gathering under delay differentiated services constraint for Internet of things," *Wireless Commun. Mobile Comput.*, vol. 2018, Feb. 2018.

[31] E. Fitzgerald, M. Pióro, and A. Tomaszwski, "Energy-optimal data aggregation and dissemination for the Internet of things," *IEEE Internet Things J.*, vol. 5, pp. 955–969, Feb. 2018.

[32] Z. Sun, X. Xing, T. Wang, Z. Lv, and B. Yan, "An optimized clustering communication protocol based on intelligent computing in information-centric Internet of things," *IEEE Access*, vol. 7, pp. 28238–28249, Jan. 2019.

[33] M. F. Ullah, J. Imtiaz, and K. Q. Maqbool, "Enhanced three layer hybrid clustering mechanism for energy efficient routing in IoT," *Sensors*, vol. 19, pp. 1–13, Jan. 2019.

[34] R. W. Coutinho, A. Boukerche, and A. A. Loureiro, "A novel opportunistic power controlled routing protocol for internet of underwater things," *Computer Communications*, vol. 150, pp. 72–82, Jan. 2020.

[35] F. H. Bijarbooneh, W. Du, E. C. Ngai, X. Fu, and J. Liu, "Cloud-assisted data fusion and sensor selection for internet of things," *IEEE Internet Things J.*, vol. 3, pp. 257–268, Jun. 2016.

[36] M. Thammawichai, S. P. Baliyarasimhuni, E. C. Kerrigan, and J. B. Sousa, "Optimizing communication and computation for multi-UAV information gathering applications," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 54, pp. 601–615, Apr. 2018.

[37] S. Rahili, J. Lu, W. Ren, and U. M. Al-Saggaf, "Distributed coverage control of mobile sensor networks in unknown environment using game theory: Algorithms and experiments," *IEEE Trans. Mobile Comput.*, vol. 17, pp. 1303–1313, Jun. 2018.

[38] C.-F. Cheng and L.-H. Li, "Data gathering problem with the data importance consideration in underwater wireless sensor networks," *J. Netw. Comput. Appl.*, vol. 78, pp. 300–312, Jan. 2017.

[39] G. Han, S. Shen, H. Song, T. Yang, and W. Zhang, "A stratification-based data collection scheme in underwater acoustic sensor networks," *IEEE Trans. Veh. Technol.*, vol. 67, pp. 10671–10682, Aug. 2018.

[40] M. Xu and L. Liu, "Sender-receiver role-based energy-aware scheduling for internet of underwater things," *IEEE Trans. Emerg. Topics Comput.*, vol. 7, pp. 324–336, Apr. 2019.

[41] D. Lin and Q. Wang, "An energy-efficient clustering algorithm combined game theory and dual-cluster-head mechanism for WSNs," *IEEE Access*, vol. 7, pp. 49894–49905, Apr. 2019.

[42] R. T. Marler and J. S. Arora, "The weighted sum method for multi-objective optimization: new insights," *Structural and Multidisciplinary Optimization*, vol. 41, pp. 853–862, Jun. 2010.

[43] M. Dorigo and L. M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," *IEEE Trans. Evol. Comput.*, vol. 1, pp. 53–66, Apr. 1997.

[44] K. Doerner, W. J. Gutjahr, R. F. Hartl, C. Strauss, and C. Stummer, "Pareto ant colony optimization: A metaheuristic approach to multiobjective portfolio selection," *Annals of Operations Research*, vol. 131, pp. 79–99, Oct. 2004.

[45] M. Bhardwaj and A. P. Chandrakasan, "Bounding the lifetime of sensor networks via optimal role assignments," in *Proc.. IEEE Annu. Joint Conf. IEEE Comput. Commun.*, pp. 1587–1596, NY, USA, 2002.

[46] S. Kaul, M. Gruteser, V. Rai, and J. Kenney, "Minimizing age of information in vehicular networks," in *Proc. IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, pp. 350–358, Salt Lake City, UT, USA, 2011.

[47] J. T. Gómez, M. Morales-Céspedes, A. G. Armada, and G. Hirtz, "Minimizing age of information on NOMA communication schemes for vehicular communication applications," in *Proc. International Symposium on Communication Systems, Networks and Digital Signal Processing*, pp. 1–6, Porto, Portugal, 2020.

[48] R. Wang, A. Yadav, E. A. Makled, O. A. Dobre, R. Zhao, and P. K. Varshney, "Optimal power allocation for full-duplex underwater relay networks with energy harvesting: A reinforcement learning approach," *IEEE Wireless Commun. Lett.*, vol. 9, pp. 223–227, Feb. 2020.

[49] Y. Sun, J. Cheng, G. Zhang, and H. Xu, "Mapless motion planning system for an autonomous underwater vehicle using policy gradient-based deep reinforcement learning," *J. Intell. Robot. Syst.*, vol. 96, pp. 591–601, Dec. 2019.

[50] L. Paull, M. Seto, and J. J. Leonard, "Decentralized cooperative trajectory estimation for autonomous underwater vehicles," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 184–191, Chicago, IL, USA, 2014.

[51] F. Campagnaro, F. Favaro, F. Guerra, V. S. Calzado, M. Zorzi, and P. Casari, "Simulation of multimodal optical and acoustic communications in underwater networks," in *Proc. IEEE/MTS OCEANS*, pp. 1–7, 2015.

[52] M. Jacobi and D. Karimanzira, "Underwater pipeline and cable inspection using autonomous underwater vehicles," in *Proc. IEEE/MTS OCEANS*, pp. 1–6, 2013.

[53] P. Gjanci, C. Petrioli, S. Basagni, C. A. Phillips, L. Bölöni, and D. Turgut, "Path finding for maximum value of information in multi-modal underwater wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 17, pp. 404–418, Feb. 2018.

[54] I. Jawhar, N. Mohamed, J. Al-Jaroodi, and S. Zhang, "An architecture for using autonomous underwater vehicles in wireless sensor networks for underwater pipeline monitoring," *IEEE Trans. Ind. Informat.*, vol. 15, pp. 1329–1340, Mar. 2019.

[55] J. Gong, T. Chang, C. Shen, and X. Chen, "Flight time minimization of UAV for data collection over wireless sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 36, pp. 1942–1954, Sep. 2018.

[56] A. Ferdowsi, M. A. Abd-Elmagid, W. Saad, and H. S. Dhillon, "Neural combinatorial deep reinforcement learning for age-optimal joint trajectory and scheduling design in UAV-assisted networks," *IEEE J. Sel. Areas Commun.*, vol. 39, pp. 1250–1265, Mar. 2021.

[57] M. A. Abd-Elmagid and H. S. Dhillon, "Average peak age-of-information minimization in UAV-assisted IoT networks," *IEEE Trans. Veh. Technol.*, vol. 68, pp. 2003–2008, Feb. 2019.

[58] C. Zhou, H. He, P. Yang, F. Lyu, W. Wu, N. Cheng, and X. Shen, "Deep RL-based trajectory planning for AoI minimization in UAV-assisted IoT," in *Proc. IEEE Wireless Communications and Signal Processing (WCSP)*, pp. 1–6, Xi'an, China, 2019.

[59] S. F. Abedin *et al.*, "Data freshness and energy-efficient UAV navigation optimization: A deep reinforcement learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, pp. 5994–6006, Dec. 2021.

[60] M. Yi *et al.*, "Deep reinforcement learning for fresh data collection in UAV-assisted IoT networks," in *Proc. IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 716–721, Toronto, ON, Canada, 2020.

[61] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 2018.

[62] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, Feb. 2015.

[63] H. Kim *et al.*, "Simulation and validation of an AUV in variable accelerations," *Int. J. Offshore Polar Eng.*, vol. 25, pp. 35–44, Mar. 2015.

[64] V. Mnih *et al.*, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[65] C. V. N. Index, "Global mobile data traffic forecast update, 2014–2019," *White Paper*, vol. 1, Feb. 2015.

[66] X. Wang, M. Chen, T. Taleb, A. Ksentini, and V. C. Leung, "Cache in the air: Exploiting content caching and delivery techniques for 5G systems," *IEEE Commun. Mag.*, vol. 52, pp. 131–139, Feb. 2014.

[67] Y. Zeng, R. Zhang, and T. J. Lim, "Wireless communications with unmanned aerial vehicles: opportunities and challenges," *IEEE Commun. Mag.*, vol. 54, pp. 36–42, May 2016.

[68] M. Alzenad, A. El-Keyi, F. Lagum, and H. Yanikomeroglu, "3-D placement of an unmanned aerial vehicle base station (UAV-BS) for energy-efficient maximal coverage," *IEEE Wireless Commun. Lett.*, vol. 6, pp. 434–437, Aug. 2017.

[69] Y. Zeng, R. Zhang, and T. J. Lim, "Throughput maximization for UAV-enabled mobile relaying systems," *IEEE Trans. Commun.*, vol. 64, pp. 4983–4996, Dec. 2016.

[70] A. A. Al-Habob, O. A. Dobre, and H. V. Poor, "Age-and correlation-aware information gathering," *IEEE Wireless Commun. Lett.*, vol. Early Access, pp. 1–5, Nov. 2021.

[71] X. Xu, Y. Zeng, Y. L. Guan, and R. Zhang, "Overcoming endurance issue: UAV-enabled communications with proactive caching," *IEEE J. Sel. Areas Commun.*, vol. 36, pp. 1231–1244, Jun. 2018.

[72] M. Chen, M. Mozaffari, W. Saad, C. Yin, M. Debbah, and C. S. Hong, "Caching in the sky: Proactive deployment of cache-enabled unmanned aerial vehicles for optimized quality-of-experience," *IEEE J. Sel. Areas Commun.*, vol. 35, pp. 1046–1061, Mar. 2017.

[73] N. Zhao, F. Cheng, F. R. Yu, J. Tang, Y. Chen, G. Gui, and H. Sari, "Caching UAV assisted secure transmission in hyper-dense networks based on interference alignment," *IEEE Trans. Commun.*, vol. 66, pp. 2281–2294, Jan. 2018.

[74] G. Chmaj and H. Selvaraj, "Distributed processing applications for UAV/drones: a survey," in *Progress in Systems Engineering*, pp. 449–454, Springer, 2015.

[75] J. Hu, H. Zhang, L. Song, R. Schober, and H. V. Poor, "Cooperative internet of UAVs: Distributed trajectory design by multi-agent deep reinforcement learning," *IEEE Trans. Commun.*, vol. 68, pp. 6807–6821, Nov. 2020.

[76] Y. Zhao, Z. Zheng, and Y. Liu, "Survey on computational-intelligence-based UAV path planning," *Knowledge-Based Systems*, vol. 158, pp. 54–64, Jun. 2018.

[77] B. Zhu, E. Bedeer, H. H. Nguyen, R. Barton, and J. Henry, "UAV trajectory planning in wireless sensor networks for energy consumption minimization by deep reinforcement learning," *IEEE Trans. Veh. Technol.*, vol. 70, pp. 9540–9554, Aug. 2021.

[78] Y. Zeng, J. Xu, and R. Zhang, "Energy minimization for wireless communication with rotary-wing UAV," *IEEE Trans. Wireless Commun.*, vol. 18, pp. 2329–2345, Mar. 2019.

[79] Z. Xue, J. Wang, G. Ding, H. Zhou, and Q. Wu, "Cooperative data dissemination in air-ground integrated networks," *IEEE Wireless Commun. Lett.*, vol. 8, pp. 209–212, Feb. 2019.

[80] Z. Xue, J. Wang, G. Ding, H. Zhou, and Q. Wu, "Maximization of data dissemination in UAV-supported Internet of things," *IEEE Wireless Commun. Lett.*, vol. 8, pp. 185–188, Feb. 2019.

[81] Y. Zeng, X. Xu, and R. Zhang, "Trajectory design for completion time minimization in UAV-enabled multicasting," *IEEE Trans. Wireless Commun.*, vol. 17, pp. 2233–2246, Apr. 2018.

[82] Z. Yang, C. Pan, K. Wang, and M. Shikh-Bahaei, "Energy efficient resource allocation in UAV-enabled mobile edge computing networks," *IEEE Trans. Wireless Commun.*, vol. 18, pp. 4576–4589, Jul. 2019.

[83] J. Sun and C. Masouros, "Deployment strategies of multiple aerial bss for user coverage and power efficiency maximization," *IEEE Trans. Commun.*, vol. 67, pp. 2981–2994, Dec. 2019.

[84] F. Zeng, R. Zhang, X. Cheng, and L. Yang, "UAV-assisted data dissemination scheduling in VANETs," in *Proc. IEEE Int. Conf. Commun.*, pp. 1–6, Kansas City, MO, USA, 2018.

[85] F. Jiang and A. L. Swindlehurst, "Optimization of UAV heading for the ground-to-air uplink," *IEEE J. Sel. Areas Commun.*, vol. 30, pp. 993–1005, Jun. 2012.

[86] F. Cheng et al., "UAV trajectory optimization for data offloading at the edge of multiple cells," *IEEE Trans. Veh. Technol.*, vol. 67, pp. 6732–6736, Jul. 2018.

[87] A. U. R. Khan, M. Othman, S. A. Madani, and S. U. Khan, "A survey of mobile cloud computing application models," *IEEE Commun. Surveys Tuts.*, vol. 16, pp. 393–413, First quarter 2014.

[88] M. Mohammadi, A. Al-Fuqaha, S. Sorour, and M. Guizani, "Deep learning for IoT big data and streaming analytics: A survey," *IEEE Commun. Surveys Tuts.*, pp. 1–40, Jun. 2018.

[89] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, pp. 2322–2358, Fourth quarter 2017.

[90] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Commun. Surveys Tuts.*, vol. 19, pp. 1628–1656, Third quarter 2017.

[91] A. A. Al-habob and O. A. Dobre, "Mobile edge computing and artificial intelligence: A mutually-beneficial relationship," *IEEE ComSoc Techncial Committees Newsletter [arXiv preprint arXiv:2005.03100, Apr. 2020]*.

[92] J. Ren, G. Yu, Y. Cai, Y. He, and F. Qu, "Partial offloading for latency minimization in mobile-edge computing," in *Proc. IEEE Global Commun. Conf.*, pp. 1–6, Dec. 2017.

[93] S. Bi and Y. J. Zhang, "Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading," *IEEE Trans. Wireless Commun.*, vol. 17, pp. 4177–4190, Jun. 2018.

[94] E. Bauer and R. Adams, *Reliability and Availability of Cloud Computing*. John Wiley & Sons, 2012.

[95] T. G. Rodrigues, K. Suto, H. Nishiyama, and N. Kato, "Hybrid method for minimizing service delay in edge cloud computing through VM migration and transmission power control," *IEEE Trans. Comput.*, vol. 66, pp. 810–819, May 2017.

[96] T. G. Rodrigues, K. Suto, H. Nishiyama, N. Kato, and K. Temma, "Cloudlets activation scheme for scalable mobile edge computing with transmission power control and virtual machine migration," *IEEE Trans. Comput.*, vol. 67, pp. 1287–1300, Sep. 2018.

[97] T. Q. Dinh, J. Tang, Q. D. La, and T. Q. S. Quek, "Offloading in mobile edge computing: Task allocation and computational frequency scaling," *IEEE Trans. Commun.*, vol. 65, pp. 3571–3584, Aug. 2017.

[98] J. Liu and Q. Zhang, "Offloading schemes in mobile edge computing for ultra-reliable low latency communications," *IEEE Access*, vol. 6, pp. 12825–12837, Feb. 2018.

[99] C. You, K. Huang, H. Chae, and B. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 16, pp. 1397–1411, Mar. 2017.

[100] F. Wang, J. Xu, and Z. Ding, "Multi-antenna NOMA for computation offloading in multiuser mobile edge computing systems," *IEEE Trans. Commun.*, pp. 1–14, Nov. 2018.

[101] M. Jia, J. Cao, and L. Yang, "Heuristic offloading of concurrent tasks for computation-intensive applications in mobile cloud computing," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, pp. 352–357, Apr. 2014.

[102] S. E. Mahmoodi, R. N. Uma, and K. P. Subbalakshmi, "Optimal joint scheduling and cloud offloading for mobile applications," *IEEE Trans. Cloud Comput.*, vol. 7, pp. 301–313, Apr. 2019.

[103] P. Vasant, *Meta-heuristics Optimization Algorithms in Engineering, Business, Economics, and Finance*. Information Science Reference, Sep. 2012.

[104] P. C. Chu and J. E. Beasley, "A genetic algorithm for the generalised assignment problem," *Comput. Oper. Res.*, vol. 24, pp. 17–23, Apr. 1997.

[105] A. A. Al-Habob, Y. N. Shnaiwer, S. Sorour, N. Aboutorab, and P. Sadeghi, "Multi-client file download time reduction from cloud/fog storage servers," *IEEE Trans. Mobile Comput.*, vol. 17, pp. 1924–1937, Aug. 2018.

[106] A. Douik, H. Dahrouj, T. Y. Al-Naffouri, and M. Alouini, "Distributed hybrid scheduling in multi-cloud networks using conflict graphs," *IEEE Trans. Commun.*, vol. 66, pp. 209–224, Jan. 2018.

[107] A. A. Al-Habob, S. Sorour, N. Aboutorab, and P. Sadeghi, "Conflict free network coding for distributed storage networks," in *Proc. IEEE Int. Conf. Commun.*, pp. 5517–5522, Jun. London, UK, 2015.

[108] A. Rudenko, P. Reiher, G. J. Popek, and G. H. Kuenning, "The remote processing framework for portable computer power saving," in *Proc. ACM Symp. Appl. Comp.*, San Antonio, TX, USA, 1999.

[109] Y. Wang, M. Sheng, X. Wang, L. Wang, and J. Li, "Mobile-edge computing: Partial computation offloading using dynamic voltage scaling," *IEEE Trans. Commun.*, vol. 64, pp. 4268–4282, Oct. 2016.

[110] P. Di Lorenzo, S. Barbarossa, and S. Sardellitti, "Joint optimization of radio resources and code partitioning in mobile edge computing," *arXiv preprint arXiv:1307.3835*, 2013.

[111] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 1, pp. 89–103, Jun. 2015.

[112] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, pp. 2795–2808, Oct. 2016.

[113] Z. Sheng, C. Mahapatra, V. C. M. Leung, M. Chen, and P. K. Sahu, "Energy efficient cooperative computing in mobile wireless sensor networks," *IEEE Trans. Cloud Comput.*, vol. 6, pp. 114–126, Jan. 2018.

[114] W. Zhang, Z. Zhang, and H. Chao, "Cooperative fog computing for dealing with big data in the Internet of vehicles: Architecture and hierarchical resource management," *IEEE Commun. Mag.*, vol. 55, pp. 60–67, Dec. 2017.

[115] Z. Zhang, W. Zhang, and F. Tseng, "Satellite mobile edge computing: Improving QoS of high-speed satellite-terrestrial networks using edge computing techniques," *IEEE Network*, vol. 33, pp. 70–76, Jan. 2019.

[116] W. Zhang, Z. Zhang, S. Zeadally, and H. Chao, "Efficient task scheduling with stochastic delay cost in mobile edge computing," *IEEE Commun. Lett.*, vol. 23, pp. 4–7, Jan. 2019.

[117] C. Yao, X. Wang, Z. Zheng, G. Sun, and L. Song, "EdgeFlow: Open-source multi-layer data flow processing in edge computing for 5G and beyond," *IEEE Network*, vol. 33, pp. 166–173, Mar. 2019.

[118] P. Wang, Z. Zheng, B. Di, and L. Song, "HetMEC: Latency-optimal task assignment and resource allocation for heterogeneous multi-layer mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 18, pp. 4942–4956, Aug. 2019.

[119] O. Muñoz, A. Pascual-Iserte, and J. Vidal, "Optimization of radio and computational resources for energy efficiency in latency-constrained application offloading," *IEEE Trans. Veh. Technol.*, vol. 64, pp. 4738–4755, Oct. 2015.

[120] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[121] A. Beck, *Introduction to Nonlinear Optimization: Theory, Algorithms, and Applications with MATLAB*, vol. 19. Siam, 2014.

[122] D. Bremner, K. Fukuda, and A. Marzetta, "Primal-dual methods for vertex and facet enumeration," *Discrete & Computational Geometry*, vol. 20, pp. 333–357, Oct. 1998.

[123] M. E. Dyer, "The complexity of vertex enumeration methods," *Mathematics of Operations Research*, vol. 8, no. 3, pp. 381–402, 1983.

[124] M. Mezzavilla *et al.*, "A lightweight and accurate link abstraction model for the simulation of LTE networks in NS-3," in *Proc. 15th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, pp. 55–60, Paphos, Cyprus, 2012.

[125] R. Marler and J. Arora, "Survey of multi-objective optimization methods for engineering," *Structural and Multidisciplinary Optimization*, vol. 26, pp. 369–395, Apr. 2004.

[126] J. L. Gross, J. Yellen, and P. Zhang, *Handbook of Graph Theory*. Chapman and Hall/CRC, 2013.

[127] K. Dasgupta, B. Mandal, P. Dutta, J. K. Mandal, and S. Dam, "A genetic algorithm (GA) based load balancing strategy for cloud computing," *Proc. Technol.*, vol. 10, pp. 340–347, Dec. 2013.

[128] C. She, C. Yang, and T. Q. S. Quek, "Cross-layer optimization for ultra-reliable and low-latency radio access networks," *IEEE Trans. Wireless Commun.*, vol. 17, pp. 127–141, Jan. 2018.

[129] E. Bastug, M. Bennis, M. Medard, and M. Debbah, "Toward interconnected virtual reality: Opportunities, challenges, and enablers," *IEEE Commun. Mag.*, vol. 55, pp. 110–117, Jun. 2017.

[130] C. You, K. Huang, and H. Chae, "Energy efficient mobile cloud computing powered by wireless energy transfer," *IEEE J. Sel. Areas Commun.*, vol. 34, pp. 1757–1771, May 2016.

[131] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. O. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," *IEEE Trans. Wireless Commun.*, vol. 12, pp. 4569–4581, Sep. 2013.

[132] R. Storn, "On the usage of differential evolution for function optimization," in *Proc. IEEE North American Fuzzy Information Processing*, pp. 519–523, Berkeley, CA, USA, 1996.