An Efficient Optimization of Battery-Drone-Based Transportation Systems for Monitoring Solar Power Plant

Mahdi Jemmali, Ali Kashif Bashir, *Senior Member, IEEE*, Wadii Boulila, *Senior Member, IEEE*, Loai Kayed B. Melhim, *Senior Member, IEEE*, Rutvij H. Jhaveri, *Senior Member, IEEE*, and Jawad Ahmad, *Senior Member, IEEE*

***Abstract*—Nowadays, developing environmental solutions to ensure the preservation and sustainability of natural resources is one of the core research topics for providing a better life quality. Using renewable energy sources, such as solar energy, is one of the solutions that can reduce the overuse of natural resources. This research aims to boost the efficiency of solar energy plants by proposing a novel approach to optimize the total flying time of battery-based drone systems to enhance the performance of solar plant systems. The contribution of the proposed approach is to solve scheduling problems based on timing constraints to monitor the solar plant. The main objective of the proposed approach is to maximize the drone's minimum total flying time, which will increase the availability and reliability of the solar plant monitoring system. Time to empty values is calculated based on battery degradation rates. This problem is proven to be NP-hard. Four categories of enhanced algorithms were developed to solve drones' scheduling problems in handling various tasks within multiple errands in the extent of solar parks in the monitored power plant to achieve the desired objective. Experimental results of the presented algorithms showed that the $M2S$ algorithm has a stable performance behavior in all conducted experiments.**

***Index Terms*—Sustainable transport, decision-making, drone, information processing, solar power surveillance, zero-carbon, energy management.**

Manuscript received 23 April 2022; revised 22 July 2022 and 21 September 2022; accepted 14 October 2022. This work was supported by the Deanship of Scientific Research at Majmaah University under Project R-2022-199. The Associate Editor for this article was Z. Lv. *(Corresponding authors: Rutvij H. Jhaveri; Mahdi Jemmali.)*

Mahdi Jemmali is with the Department of Computer Science and Information, College of Science at Zulfi, Majmaah University, Al-Majmaah 11952, Saudi Arabia, also with the MARS Laboratory, University of Sousse, Sousse 4023, Tunisia, and also with the Department of Computer Science, Higher Institute of Computer Science and Mathematics, University of Monastir, Monastir 5000, Tunisia (e-mail: m.jemmali@mu.edu.sa).

Ali Kashif Bashir is with the Department of Computing and Mathematics, Manchester Metropolitan University, M15 6BH Manchester, U.K. (e-mail: dr.alikashif.b@ieee.org).

Wadii Boulila is with the Robotics and Internet-of-Things Laboratory, Prince Sultan University, Riyadh 12435, Saudi Arabia, and also with the RIADI Laboratory, National School of Computer Science, University of Manouba, Manouba 2010, Tunisia (e-mail: wadii.boulila@riadi.rnu.tn).

Loai Kayed B. Melhim is with the Department of Health Information Management and Technology, College of Applied Medical Sciences, University of Hafr Al Batin, Hafr Al Batin 39524, Saudi Arabia (e-mail: l.banimelhim@uhb.edu.sa).

Rutvij H. Jhaveri is with the Department of Computer Science and Engineering, School of Technology, Pandit Deendayal Energy University, Gandhinagar 382007, India (e-mail: rutvij.jhaveri@sot.pdpu.ac.in).

Jawad Ahmad is with the School of Computing, Edinburgh Napier University, EH14 1DJ Edinburgh, U.K. (e-mail: j.ahmad@napier.ac.uk).

Digital Object Identifier 10.1109/TITS.2022.3219568

I. Introduction

THE Earth contains natural resources such as air, water, soil, plants, and animals that provide the means of life for humans. However, the excessive use of these resources created environmental, economic, and social problems. Some of these problems are the depletion of natural resources, pollution issues, and poisoning of the environment, besides the emergence of abnormal phenomena such as global warming and acid rain. Recently, societies became aware of the excessive use of these resources besides their impact on life [1]. Voices have risen for proper alternatives to preserve and sustain natural resources. Sustainable development (SD) has become a top priority for the world's policymakers. SD is defined as the "development that meets the needs of the present without compromising the ability of future generations to meet their own needs" [2]. As a result, many solutions have been presented, which may be used to address the imbalance caused by humans. Examples of such solutions are green societies, green technologies, renewable energy, and hybrid systems.

Many researchers believe that renewable energy is an essential alternative to respond to the excessive use of natural resources [3]. Renewable energy has several resources such as solar, wind, hydropower, geothermal, biomass, and hybrid energy sources. Solar energy is known as a permanent, available, and environmentally friendly alternative to fossil fuels [4]. Several technologies are utilized to maximize the output of solar energy systems [5]. Several countries are adopting solar energy plant systems as support systems for their energy sources, such as India, China, the Republic of Cuba, the United Arab Emirates, Morocco, and many more examples worldwide. Solar power plants harvest renewable energy from sunlight directly using solar panels photovoltaic, indirectly using the large number of mirrors organized to generate concentrated solar power [6], or by a combination of both technologies [7]. Both technologies depend on exposing a wide area of solar panels or mirrors in mounted solar farms.

In this research, an errand is defined as a path that has a group of mirrors or solar panels that are arranged side by side horizontally or vertically, as shown in Figures 2 and 3. The vast number of solar panels or mirrors spread over a wide area requires continuous monitoring and maintenance tasks to enhance the solar plant's availability, reliability, performance, life cycle, etc. Different approaches were proposed to ensure these tasks [8]. Most of these approaches deploy drone-based

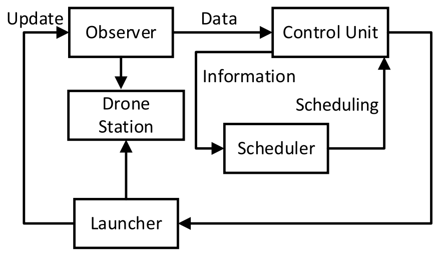


Fig. 1. General overview of the solar plant monitoring system.

cleaning systems. The drone-based systems are a justified choice because of the nature of the required tasks. Such systems are needed to perform the monitoring or the cleaning tasks. In these systems, monitoring and cleaning tasks are assigned to a set of specially equipped drones.

The proposed drone-based system includes many battery-powered drones. Each drone has at least one spare battery system that is fully charged and ready to be installed when required. Moreover, every drone has a specific battery type. For drones with the same battery type, recharging conditions are not the same in the s time slot; due to the different assigned tasks that require various flying times for each drone. Batteries are recharged by the provided charging system that uses solar-generated power. Drones are available and ready to go at any time. Battery-powered drones rely on batteries to deliver the needed energy to fly the drone for a certain amount of time, which is estimated based on the battery degradation rate before batteries require recharging again. Degradation rates are different due to many factors, including the working environment and charging cycles. Nondeterministic battery lifetime produces various flying times and nondeterministic flying distances. The drone's flying time is derived based on the estimated distance that the battery can offer and will be known as the distance to empty. This work objective is to search for an appropriate schedule that ensures drones' maximum operation time based on batteries' time to empty preferences.

The overall structure of the solar plant monitoring system is shown in Figure 1. In this system, all drones are well prepared and ready to go. In the drone station, the observer is responsible for collecting the required data about the drone's status, such as the battery status, charging status, drone readiness, and the expected time to empty. The collected data are sent to the control unit of the solar plant monitoring system. The control unit extracts the required information by executing the received data and then sends this information to the scheduler. The scheduler utilizes the provided information and the developed algorithms to generate new schedules that will be sent to the control unit. The control unit stimulates the drone launcher to perform the launching process based on the provided scheduling.

The solar plant monitoring system provides tasks of different types to the control unit of the drone-based system, these tasks are managed by a set of developed algorithms. These algorithms handle various tasks within many errands, as shown in Figure 3. Each drone is assigned to perform any task for a single errand based on the drone's time to empty criterion. The time required by each drone to finish the assigned tasks in each errand is the total flying time. The main objective of this research is to maximize the minimum total flying time. The systems' reliability and availability are ensured by adopting complex scheduling of the solar plant monitoring tasks, such as monitoring, maintaining, logistics supply, rescuing, and transportation. The performance of the drone-based system increases as the number of performed tasks increases. While reliability and availability of the drone-based system are achieved if, for any time slot, there is a drone flying over the monitored extent (i.e., by increasing the drone flying time).

In drone-based systems, the number of performed tasks depends directly on the flying periods (i.e., if the flying time of these drones increases, the number of completed tasks increases, which increases the efficiency of the monitoring systems). The final objective of this work is to optimize the solar plant monitoring system by maximizing the drone flying time, which is achieved when solving the drone scheduling problem. The scheduling problem will be solved by proposing several algorithms that increase the solar plant monitoring system's ability to handle more monitoring tasks, including cleaning tasks. The required optimization will be achieved by developing a framework to manage and maintain a fleet of drone-based systems, with the goal of increasing the drone flying time. The suggested framework in this context will use several algorithms. The utilization of solar power plants motivates researchers to consider drone-based systems to perform the required tasks. The presented problem has never been studied in the literature to the best of the author's knowledge. The current context is an attempt to bridge the gaps in the solar plant monitoring system by developing six algorithms to make the drone-based system service available whenever needed.

Drones fly by the power provided by a battery. Each drone has a specific battery type. For drones with the same battery type, the available charge in each battery is not necessarily the same in a given time slot. This work objective is to search for an appropriate schedule that maximizes drones' minimum total flying time. Therefore, a load balancing problem is applied. This load balancing is also known as equity distribution or fair distribution. Ensuring a maximum flying time for all drones is equivalent to maximizing the minimum time to empty for drones assigned to each errand. It is worth noting that every errand has a cumulative or a total time to empty. This time corresponds to the sum of all times to empty for the drones assigned to this errand.

The final objective when maximizing the minimum is the load balancing or what is called equity distribution. Several research papers deals with the fair distribution or load balancing that were applied for different domains like healthcare [9], projects assignments and management [10], [11], [12], finance and budgeting [13], parallel machines [14], network [15], and aviation sector [16]. The researchers build a mathematical model of the drone flying distance problem to meet this objective. The minimum cumulative flying time will be maximized by developing six algorithms based on four algorithm categories, the utilization of the iterative methods based on the probabilistic method, the randomized method, the repeating of the resolution of the swapping method, and the dispatching rules method.

In the case of truthful mechanisms, the authors in [17] maximize the minimum completion time of a machine by providing the first constant-factor approximation for deterministic truthful mechanisms for scheduling on a single-parameter mechanism design. Two variants of the MapReduce system were presented by [18]. The first variant was the preemptive reduce tasks, with an optimal solution algorithm on two and three machines. While the second variant was the non-preemptive reduce tasks with an approximation algorithm with a tight worst-case ratio of $\frac{4}{3}$ on two machines. In [19] the researchers addressed maximize of the minimum problem with unknown processing times by presenting ordinal algorithms to solve the problem of assigning all jobs to some machines at time zero, where the sort of the given jobs based on their processing times is known in advance.

On the other hand, several research works utilize the randomization algorithm to solve approximately a given problem. In addition, the author gives the utility of the randomization algorithms in applications. Several other works can be cited to refer the randomization algorithms [13], [14], [20], and [16]. The machine learning approach and the IoT technology used in [21], [22], and [23] can be expanded to be utilized in the various time to empty values for drones.

The use of drones is proposed by several researchers, especially for 5G flying drones in [24]. The critical energy infrastructures were discussed in [25]. The algorithm for task scheduling in IoT applied in [26] can be adopted to solve the studied problem. On the other hand, authors in [27] presented a framework coined SDN-RMbw calculating a path-finding algorithm for new routes. The algorithm proposed in [28] can be enhanced using the randomization method developed for the studied problem. In addition, the user preferences used in the latter paper can be applied in the studied problem to be user preferences or routing drones.

The main contribution of this research is to build algorithms that maximize the drones' total minimum flying time. Four categories were utilized to construct the presented algorithms. Those categories are the randomized method of flying time, the probabilistic algorithm, the swapping and the mixing method between some fixed rules, and the dispatching rules method. The presented algorithms are developed to maximize the drones' total minimum flying time by achieving the following main contributions.

- Maximizing the minimum flying time provides the monitoring system with more capability and flexibility to expand the system's scope of work and perform more tasks.
- The enhancement of the monitoring system improves the performance of the power plant system in general and increases the efficiency of renewable energy systems, which contributes effectively to providing environmentally friendly alternatives for various power systems and will reduce dependence on Fossil fuels to meet the growing energy demand.
- The presented algorithms can be considered as one of the foundations in building energy systems that are an influential contribution to preserving the environment.
- The proposed approach can be adopted in several different fields to enhance the effectiveness of monitoring systems that rely on drones in both military and civilian applications, such as monitoring forests, fires, residential areas, agricultural areas, deserts, and other different monitoring tasks.

The coming sections of this work are structured as follows. Section 2 discusses the problem description. The proposed algorithms will be presented in section 3. Discussion of the experimental results will be detailed in section 4. Finally, the conclusion will be presented in section 5.

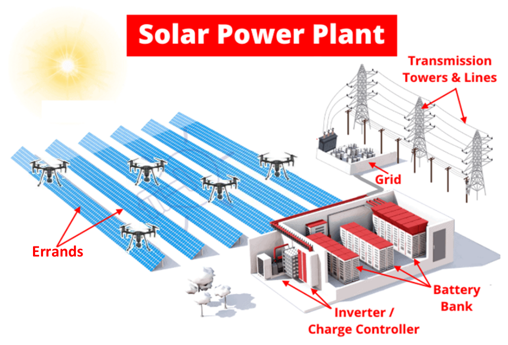


Fig. 2. Solar plant system monitored by drones.

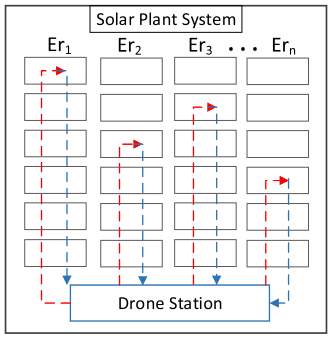


Fig. 3. Drone controlling system.

II. Problem Description

The studied problem is based on controlling the drones' scheduling system of the solar power plant monitoring system. These drones fly over the plant extent, make monitoring round and return to the drone station. Each drone can fly for a certain distance, specified by the amount of available power that the battery can provide. This distance is known as the distance to empty that is represented in this context by time to empty. Time to empty values is nondeterministic due to batteries' various degradation rates and other factors explained previously. The general overview of the solar plant system and the monitoring drones are illustrated in the overview depicted in Figure 2. The park of the solar power plant is divided into several errands. This division facilitates the control and monitoring of the park. Each drone can visit one errand at one time and return to the drone station. When the drone finishes the assigned tasks, it must return to the drone station even if it has sufficient flying time to visit other errands. The constraints of the presented problem state that a single drone can perform some tasks for one errand at a time. When the drone returns to its station, the control unit updates the scheduler and reassigns the existing tasks to the drones to start a new round of flying.

The drone controlling system is shown in Figure 3. It is worth noting here that the generated errands shown in Figures 2 and 3 are of the same length, but the drones' traveled distances over each errand are not equal because they are specified by the time to empty values as shown in Figure 3.

When the errands are of the same length, any drone can travel to any errand. The number of drones that will visit each errand depends on the number of tasks to be performed for that errand, which is the case in this paper. While for the case when errands are of various lengths, algorithms must be modified to check the errands length and the required flying

TABLE I
TIME TO EMPTY-DRONES ASSIGNMENT FOR EXAMPLE 1

j	1	2	3	4	5	6	7
te_j	20	15	40	35	50	45	60

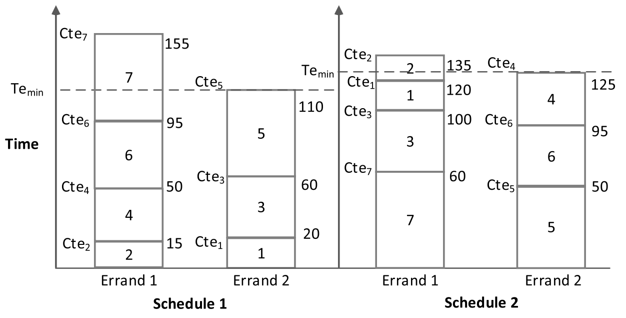


Fig. 4. Time to empty-drones schedules for Example 1.

time before assigning the drones to any errands. If the drone's flying time was much greater than the errand required time, then this drone will be assigned to another errand. So that drones with short flying times will be assigned to short errands and drones with long flying times will be assigned to longer errands. The rest of the assumed work will be the same.

The assignment problem is based on maximizing the minimum total time to empty drone batteries. In this paper, we will denote by S_{dr} the set of drones, n_{dr} the number of drones, n_e the number of errands, i the errand index, and Te_i the total time to empty for the errand E_i. The time to empty for the drone Dr_j is represented by the variable te_j where j is any drone. Whenever a drone Dr_j is assigned to any errand, it will be assigned to one errand only. Also, it is assumed that all drones are available immediately when required. The studied problem corresponds to maximizing the minimum flying time of drones. This minimum time is denoted by Te_{min}. While the cumulative time to empty of drone Dr_j will be denoted by Cte_j. The gap between total flying time for each errand scheduling is given in Equation (1)

$$GJ = \sum_{i=1}^{n_e}(Te_i - Te_{min}). \quad (1)$$

Minimizing GJ is the primary goal of this paper.

The following example explains the fundamental role of the scheduling problem.

Example 1: Let $n_{dr} = 7$ and $n_e = 2$. Table I represents the time to empty te_j for each drone Dr_j.

Considering the flying times described in Table I, we will apply a random algorithm to assign the corresponding drones to the given two errands ($n_e = 2$). The resulted schedule by using this algorithm is shown in Figure 4. It is observed that for schedule 1, drones $\{2, 4, 6, 7\}$ are assigned to errand 1, while drones $\{1, 3, 5\}$ are assigned to errand 2. For schedule 2, drones $\{1, 2, 3, 7\}$ are assigned to errand 1, while drones $\{4, 5, 6\}$ are assigned to errand 2.

For schedule 1, the total time to empty for errand 1 is 155, as shown in Figure 4. However, the total time to empty for errand 2 is 110. Thus, the total time to empty 155 is the maximum operating time is $Te_{max} = 155$. While the minimum total time to empty is $Te_{min} = 110$. The gap between the operating lifetimes is $GJ = Te_{max} - Te_{min} = 155 - 110 = 45$.

This work aims to minimize GJ, the gap between the errands. Thus, an enhanced algorithm that returned a minimum gap of less than 45 is required. Schedule 2 can represent this enhanced algorithm. Indeed, the total time to empty for errand 1 is 135, as shown in Figure 4. However, the total time to empty for errand 2 is 125. Thus, the total time to empty $Te_{max} = 135$ and the minimum total time to empty $Te_{min} = 125$. The gap between the operating lifetimes is $GJ = Te_{max} - Te_{min} = 135 - 125 = 10$. Minimizing the gap between the different errands means that at any time, more drones will be flying over a certain errand, thus increasing the number of performed tasks or missions, which increases the availability and the performance of the monitoring system, thus increasing the performance of the solar plant system.

Proposition 1: The studied problem is an NP-hard in the strong sense.

Proof: $\sum_{i=1}^{n_e} [Te_i - Te_{min}] = \sum_{i=1}^{n_e} T_i - \sum_{i=1}^{n_e} Te_{min}$. On other hand we have $\sum_{i=1}^{n_e} = Te_i = \sum_{j=1}^{n_{dr}} te_j$. It is clear to see that $\sum_{j=1}^{n_{dr}} te_j$ is not changed when the algorithm of scheduling is changed. This conduct to $\sum_{i=1}^{n_e} \big[Te_i - Te_{min}\big] = \sum_{j=1}^{n_{dr}} te_j - \sum_{i=1}^{n_e} Te_{min}$. Thus, Minimize $\sum_{i=1}^{n_e} \big[Te_i - Te_{min}\big]$ = Minimize $\sum_{j=1}^{n_{dr}} te_j - \sum_{i=1}^{n_e} Te_{min} = \sum_{j=1}^{n_{dr}} te_j$−Minimize $\sum_{i=1}^{n_e} Te_{min}$. The -Minimize $\sum_{i=1}^{n_e} Te_{min}$ can be considered as: Maximize Te_{min}. The maximization of Te_{min} is corresponding to parallel machine problem to maximize the C_{min} which is proved to be NP-hard in several works in literature. This correspondence is as follows: the errands are corresponding to the machines and the drones corresponding to the jobs.

III. Proposed Algorithms

In this paper, several lower-bounds algorithms are developed. These algorithms are based on four categories. The first category is based on the dispatching rules method. The second category is based on the swapping and the mixing method. The third category is based on the probabilistic method. While the fourth category is based on the randomization method.

A. Non-Increasing Time to Empty Order Algorithm (NIT)

In this algorithm, all drones are arranged in a non-increasing order based on the time to empty, then the drone with the longest time to empty will be assigned to the errand that has the minimum Te_i. The complexity of NIT is $O(nlogn)$.

B. Non-Increasing-Decreasing Swapping Time to Empty Algorithm ($IDST$)

This algorithm starts by selecting δ of drones with the longest time to empty, then selecting another δ of drones with the shortest time to empty, and so on, until the selection of all drones is finished. In the practice, δ is in $\{1, \cdots, n_{dr} - 1\}$. The function that returns the drone with the longest time to empty is $longer(P_s)$ where P_s is the list of the available drones. In addition, the function that returns the drones with the shortest time to empty among P_s will be denoted by $shorter(P_s)$.

The iteration of Algorithm 1 for $n_{dr} - 1$ times with the maximum value returned; will derive a new algorithm which is Algorithm 2. The complexity of $IDST$ is $O(n^2logn)$.

Algorithm 1 δ-Time to Empty Swapping Procedure: $IDST^{\delta}()$

```
1: Set $r = 1$; $P_s = S_{Dr}$
2: while ($r \leq n_{dr}$) do
3:     for ($l = 1$ to $l = \delta$) do
4:         $j_s = longer(P_s)$
5:         $Store(j_s)$
6:         $P_s = P_s \setminus j_s$
7:         $r++$
8:         if ($r > n_{dr}$) then
9:             Break;
10:        end if
11:    end for
12:    if ($r \leq n_{dr}$) then
13:        for ($l = 1$ to $l = \delta$) do
14:            $j_s = shorter(P_s)$
15:            $Store(j_s)$
16:            $P_s = P_s \setminus j_s$
17:            $r++$
18:            if ($r > n_{dr}$) then
19:                Break;
20:            end if
21:        end for
22:    end if
23: end while
```

Algorithm 2 Non-Increasing-Decreasing Swapping Time to Empty Algorithm ($IDST$)

```
1: for ($l = 2$ to $l = n_{dr} - 1$) do
2:     $GJ^l = IDST^l(S_{dr})$
3: end for
4: Calculate $GJ = \min_{2 \leq l \leq n_{dr}-1} (GJ^l)$
5: Return $GJ$.
```

C. Multi-Restart Probabilistic Algorithm (MPV)

For this algorithm, available drones are arranged according to the decreasing order of their time to empty. After that, choose between the two drones with the largest time to empty, applying a probability α. Next, iterate the algorithm several times (In the performed experiments, the number of iterations is chosen to be $it = 1000$). The last step is to choose the best solution among the returned values. The complexity of MPV is $O(nlogn)$.

D. Multi-Restart Probabilistic Variation Algorithm (MPV^{α})

For this algorithm, arrange all drones according to the non-increasing order of their time to empty. After that, probability α will guarantee to choose one of the two longest times to empty, while the value of β will choose between the first or the second-longest time to empty, where β is the value returned by the function $rand(1, 100)$, as explained in (Algorithm 3). Finally, change the probability value several times to derive different minimum times to empty values, then choose the best-derived value.

Algorithm 3 can be improved by applying a fixed number of iterations with $it = 1000$. This improvement will be obtained after calling the Algorithm 3 several times. For each call new β value will be returned, and a new solution will be derived. The best solution will be selected. These steps will generate Algorithm 4. The complexity of MPV^{α} is $O(n^2logn)$.

Algorithm 3 Probabilistic Variation Algorithm (PV^{α})

```
1: $Decr(S_{Dr})$
2: Set $\alpha = 0.1$
3: while ($\alpha \leq 0.9$) do
4:     $k = 1$ and $P_s = S_{Dr}$
5:     while ($k \leq n_{dr}$) do
6:         $\beta = rand(1, 100)$
7:         if ($\beta \in [1, \alpha \times 100]$) then
8:             Select the first drone.
9:             $Assign(1)$
10:            $P_s = P_s \setminus \{1\}$
11:        else
12:            Select the second drone.
13:            $Assign(2)$
14:            $P_s = P_s \setminus \{2\}$
15:        end if
16:        $k++$
17:    end while
18:    Determine $GJ^{\alpha}$
19:    $\alpha = \alpha + 0.05$
20: end while
21: Calculate $GJ = \min_{0.1 \leq \alpha \leq 0.9} (GJ^{\alpha})$
```

Algorithm 4 Probabilistic Variation Algorithm (MPV^{α})

```
1: Set iter
2: for ($k = 1$ to $k = iter$) do
3:     Calculate $GJ^k = PV^{\alpha}$
4: end for
5: Calculate $GJ = \min_{1 \leq k \leq iter} (GJ^k)$
6: Return $GJ$.
```

E. Randomized Errand-Drone Based Algorithms RED

For this algorithm, start by choosing drones with the longest time to empty to be scheduled on the most available errand with probabilistic variants. The focus is on probabilistic choice from the k drones having the largest time to empty with $k \in \{2, 3, 4, 5\}$. The longest time to empty is chosen randomly between 1 and k, where k represents the number of the longest time to empty drones, which are considered to be selected first. The best obtained value will be picked and forwarded to the algorithm.

The selected drones are chosen from the k that has the longest time to empty with probability α. This probability is fixed as follows:

- select a random number r in the interval $[1, k]$. Then choose the drone that has the largest r^{th} unscheduled drone, then assign the selected drone to the most available errand.
- The number of unscheduled drones is denoted by UN_{dr}. If $UN_{dr} < k$, then r will be selected randomly between $[1, UN_{dr}]$.

For a fixed k, choose an iteration number ni to be $ni = 1000$. Hereafter, denote by $Assign(te_j)$, the procedure that schedules drone j to the most available errand. To apply the proposed algorithm, the first step will be to arrange the drones in a non-increasing order based on their time to empty. The algorithm of the randomized errand-drone is described below (see Algorithm 5). The complexity of RED is $O(nlogn)$.

Algorithm 5 Randomized Errand-Drone Algorithm (RED)

```
1: for $k = 2$ to 5 do.
2:     Set $l = 1$.
3:     while ($l \leq ni$) do
4:         Set $S_k = S_{dr}$.
5:         while ($S_k \neq \emptyset$) do
6:             Select $r$ in $[1-k]$.
7:             $Assign(te_r)$.
8:             $S_k = S_k \setminus r$.
9:         end while
10:        Evaluate $GJ^l$
11:        $l++$.
12:    end while.
13:    $RED(k) = \min_{1 \leq l \leq ni} GJ^l$.
14: end for
15: Calculate $GJ = \min_{2 \leq k \leq 5} RED(k)$.
16: Return $GJ$.
```

F. Minimum Two-Selected Based Algorithms M2S

This algorithm is based on the RED algorithm and $IDST$ algorithm. In fact, after running these two latter algorithms the best solution will be selected. Indeed, $M2S = min(RED, IDST)$.

IV. Experimental Results

This section presents the obtained results, the analysis, and the discussion of the obtained results after the implementation of the developed algorithms. Microsoft Visual C++ (Version 2013) was used to implement the developed algorithms. The implemented algorithms were executed on an Intel(R) Xeon(R) CPU E5-2687W v4 @3.00 GHz with RAM of 64 GB, 64 bits windows 10 workstations. The developed algorithms were tested by a set of instances. The time to empty te_j was generated according to different probability distributions. In fact, $U[x, y]$ is the uniform distribution between $[x, y]$ and $N[x, y]$ is the normal distribution. Each one of the probability distributions represents a class. These classes are:

- Class 1: te_j in $U[1, 100]$.
- Class 2: te_j in $U[20, 100]$.
- Class 3: te_j in $U[50, 100]$.
- Class 4: te_j in $N[50, 100]$.
- Class 5: te_j in $N[20, 100]$.

The generated instances were obtained by choosing n_{dr}, n_e and $Class$. For each triple $(n_{dr}, n_e, Class)$, ten instances are tested.

The pair (n_{dr}, n_e) has many possibilities and was populated by a set of values as shown in Table II.

As shown by the Table II the total of instances is $(3 \times 3 + 4 \times 5 + 5 \times 7) \times 5 \times 10 = 3200$.

TABLE II
CHOICE OF THE PAIR (n_{dr}, n_e)

n_{dr}	n_e
10,20,25	2,3,5
50,100,150,200	2,3,5,10,15
250,300,350,400,500	2,3,5,10,15,20,30

TABLE III
OVERALL RESULTS

	NIT	$IDST$	MPV	MPV^α	RED	$M2S$
$Perc$	40.03%	61.22%	63.53%	74.13%	79.59%	95.94%
$Time$	0.0001	0.0022	0.0177	0.3059	0.0722	0.0744

We denoted by:

- H_* the best (minimum) value of GJ saved after execution of all algorithms.
- H the developed algorithm value of GJ.
- $GAP = \frac{H-H_*}{H}$ and $GAP = 0$ if $H = 0$. The symbol "*" is used when the GAP variations are less than 0.001.
- $Time$ algorithm elapsed time for the corresponding instances. Time is given in seconds and is denoted by "-" if the time value is less than 0.0001 seconds.
- $Perc$ the percentage of the (3200) instances that has $H = H_*$.

Choosing the number of iterations to be 1000 was selected experimentally, as we noticed that a smaller number of iterations negatively affect the results while increasing the number of iterations to more than 1000 requires more time and will not lead to improved results. Choosing the number of instances to be 3200 was made to clarify the possibilities and to ensure the expansion of the choices to include all possible scenarios.

Table III presents the overall results of all instances. This table shows that the best algorithm is $M2S$ with a percentage of 95.94 % and an average running time of 0.0744 s. The second best algorithm is RED with a percentage of 79.59 %, while the third-best algorithm was MPV^α with a percentage rate of 74.13 %. Although the $M2S$ algorithm percentage was the best, the execution time is acceptable and less than the running time for MPV^α. The worst algorithm based on the percentage rate was the NIT algorithm with a percentage rate of 40.03 %.

Table IV shows GAP variations according to the number of drones for all algorithms. It can be noticed that when $n_{dr} \geq 350$, the performance of $IDST$ began to improve significantly. For this reason, we choose to develop $M2S$ based on the maximum between $IDST$ and RED algorithms. It is worthily noting that for $IDST$ when $n_{dr} \geq 25$, the performance increases remarkably when n_{dr} increases. The given results show that the developed algorithms performed well even with a high number of drones $n_{dr} = 500$, for a low number of drones $n_{dr} = 10$, most algorithms managed the system with high-performance values. The indication of these results is the availability and reliability that will be provided for the monitoring systems.

Table V presents GAP variation according to the number of errands for all algorithms. The performance of the algorithm $M2S$ has the best performance for all n_e values. The most common criteria among all solar plants are the wide extent of

TABLE IV
GAP VARIATION ACCORDING TO THE NUMBER OF DRONES FOR ALL ALGORITHMS

n_{dr}	NIT	$IDST$	MPV	MPV^{α}	RED	$M2S$
10	0.509	0.402	0.086	0.081	*	*
20	0.815	0.602	0.078	0.036	0.019	0.019
25	0.822	0.632	0.166	0.081	0.023	0.023
50	0.565	0.353	0.120	0.030	0.036	0.023
100	0.520	0.253	0.208	0.116	0.077	0.011
150	0.380	0.247	0.082	0.029	0.029	0.012
200	0.404	0.193	0.195	0.080	0.074	0.009
250	0.370	0.153	0.209	0.172	0.144	0.009
300	0.294	0.167	0.139	0.057	0.063	0.010
350	0.329	0.108	0.211	0.146	0.137	0.003
400	0.330	0.142	0.208	0.155	0.122	0.004
500	0.297	0.090	0.197	0.147	0.128	0.005

TABLE V
GAP VARIATION ACCORDING TO THE NUMBER OF ERRANDS FOR ALL ALGORITHMS

n_e	NIT	$IDST$	MPV	MPV^{α}	RED	$M2S$
2	0.325	0.164	0.013	0.004	*	*
3	0.602	0.317	0.274	0.142	0.048	0.001
5	0.405	0.372	0.067	0.018	0.016	0.016
10	0.410	0.178	0.160	0.081	0.083	0.018
15	0.341	0.162	0.221	0.167	0.158	0.012
20	0.349	0.107	0.246	0.186	0.182	0.005
30	0.379	0.113	0.297	0.247	0.254	0.017

TABLE VI
GAP VARIATION ACCORDING TO $Class$ FOR ALL ALGORITHMS

$Class$	NIT	$IDST$	MPV	MPV^{α}	RED	$M2S$
1	0.266	0.226	0.049	0.010	0.010	0.010
2	0.409	0.234	0.149	0.075	0.059	0.010
3	0.383	0.130	0.249	0.200	0.169	0.006
4	0.405	0.299	0.087	0.034	0.028	0.015
5	0.656	0.263	0.307	0.197	0.153	0.008

solar parks, the larger area of the solar parks in the power plant means a greater number of errands, which means more tasks to perform and more complex schedules to handle, here where the importance of the developed algorithms appears. The high number of errands $n_e = 30$ indicates that the developed algorithms will perform well, even for a vast extent of solar parks.

Table VI represents GAP variations according to $Class$ for all algorithms. It can be noticed that for the algorithm $M2S$ the value of the approximately average gap of 0.010 is almost constant for all classes. Algorithms NIT, MPV, MPV^{α}, and RED are more efficient for $Class$ 1. However, $IDST$ is more efficient for $Class$ 3.

Hereafter, the pair (n_{dr}, n_e) will be denoted by Pa. Examining n_{dr} and n_e values, we have in total 64 pairs in Pa. Figure 5 shows the variation of GAP for $M2S$ algorithm according to Pa. The given results show that GAP values were in the range of 0.000 to 0.040 for most of the Pa values. Figure 5 shows that there are five peaks of GAP values when Pa changes. These peaks are obtained when $Pa = \{6, 9, 13, 23, 41\}$ which corresponds to the pairs $(n_{dr}, n_e) = \{(20, 5), (25, 5), (50, 10), (150, 10), (300, 15)\}$, respectively. The GAP values of these pairs are respectively $\{0.057; 0.068; 0.062; 0.047; 0.048\}$.

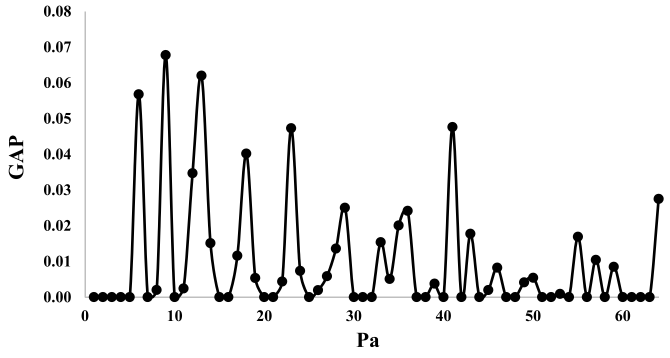


Fig. 5. The variation of GAP for $M2S$ algorithm according to Pa.

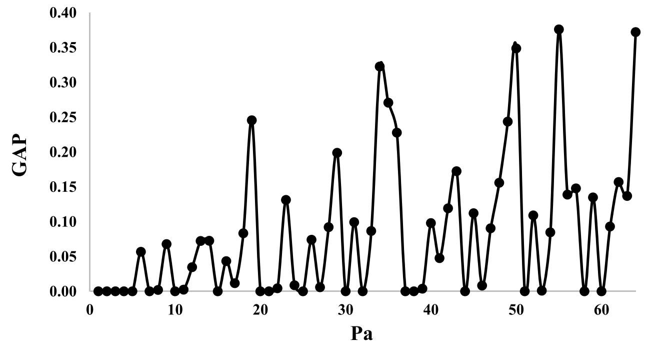


Fig. 6. The variation of GAP for RED algorithm according to Pa.

TABLE VII
$Time$ ACCORDING TO n_{dr}

n_{dr}	NIT	$IDST$	MPV	MPV^{α}	RED	$M2S$
10	-	-	0.0008	0.0074	0.0028	0.0028
20	-	-	0.0013	0.0143	0.0045	0.0045
25	-	-	0.0013	0.0184	0.0050	0.0050
50	0.0001	0.0001	0.0032	0.0495	0.0156	0.0158
100	0.0001	0.0005	0.0064	0.1079	0.0369	0.0374
150	0.0001	0.0006	0.0097	0.1607	0.0439	0.0445
200	0.0001	0.0010	0.0126	0.2127	0.0503	0.0513
250	-	0.0016	0.0182	0.3020	0.0748	0.0765
300	-	0.0023	0.0219	0.3655	0.0885	0.0909
350	0.0001	0.0034	0.0263	0.4362	0.1035	0.1068
400	0.0001	0.0041	0.0306	0.5864	0.1218	0.1259
500	0.0001	0.0071	0.0409	0.7103	0.1616	0.1687

Figure 6 shows the variation of GAP for RED algorithm according to Pa. The given results show that GAP values were in the range of 0.00 to 0.020 for most of the Pa values. The given results in Figure 6 show five peaks of GAP values when Pa changes. The highest peak of 0.38 is reached when $Pa = 55$ which corresponds to the pair $(n_{dr}, n_e) = \{400, 15\}$.

The corresponding running time for the results given in Tables IV, V, and VI are presented in Tables VII, VIII, and IX, respectively. As it can be seen from the tables, all the algorithms have good performance measures based on running time

TABLE VIII
$Time$ ACCORDING TO n_e

n_e	NIT	$IDST$	MPV	MPV^α	RED	$M2S$
2	0.0001	0.0004	0.0112	0.1916	0.0467	0.0471
3	-	0.0006	0.0116	0.2013	0.0468	0.0474
5	-	0.0009	0.0125	0.2216	0.0510	0.0519
10	-	0.0018	0.0175	0.2984	0.0708	0.0726
15	0.0001	0.0026	0.0187	0.3239	0.0773	0.0799
20	0.0001	0.0062	0.0336	0.5612	0.1333	0.1395
30	0.0001	0.0086	0.0365	0.6348	0.1480	0.1566

TABLE IX
$Time$ ACCORDING TO $Class$

$Class$	NIT	$IDST$	MPV	MPV^α	RED	$M2S$
1	-	0.0022	0.0180	0.3106	0.0730	0.0752
2	0.0001	0.0022	0.0179	0.3072	0.0732	0.0755
3	0.0001	0.0023	0.0173	0.2959	0.0709	0.0732
4	0.0001	0.0023	0.0180	0.3122	0.0725	0.0748
5	-	0.0020	0.0176	0.3035	0.0715	0.0735

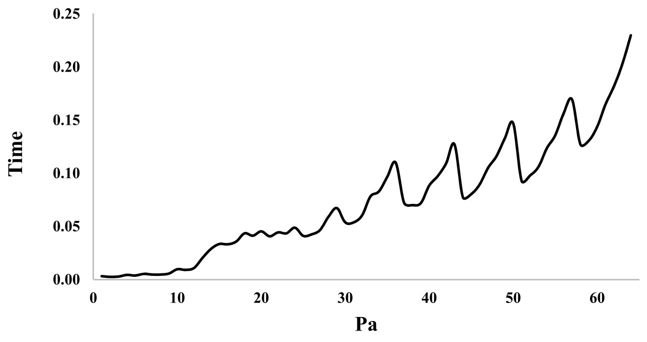


Fig. 7. The variation of $Time$ for $M2S$ algorithm according to Pa.

except for MPV^α, where the running time increases as n_{dr} values increases. Table VII shows that the highest running time of 0.7103 s is obtained for MPV^α when $n_{dr} = 500$. However, for NIT the running time is less than 0.0001 s for all values of n_{dr}.

Table VIII shows that the highest running time of 0.6348 s is obtained for MPV^α when $n_e = 30$. However, for NIT the running time is less than 0.0001 s for all values of n_e.

Table IX shows that the maximum running time of 0.3122 s is obtained for MPV^α when $Class = 4$.

Figure 7 shows the variation of $Time$ for $M2S$ algorithm according to Pa. This figure shows that the running time of $M2S$ almost increases when Pa increases.

V. CONCLUSION AND PROSPECTS

The problem of the battery-based-drone systems scheduling was presented in this research to maximize the minimum flying time of drones (i.e., lowest downtime and the most extended operation time). This problem was resolved by developing six algorithms to solve approximately the studied problem. The battery time to empty is considered to be of different values. The developed algorithms are based on four categories. The first category is based on the randomized method of flying time. The second category is based on a probabilistic algorithm. While the third category is the swapping and the mixing method between some fixed rules. The fourth category is based on a dispatching rules method. The six developed algorithms were compared based on several criteria, including the number of assigned drones, number of errands, class, gap values, and running time values. The experimental results showed that the $M2S$ algorithm had achieved the best performance in all experiments. The presented work assumes that all errands are equal in length and that any drone can visit only one errand during one task. Several possible extensions can be considered in future work. We plan to use meta-heuristics such as genetic algorithms and swarm optimization algorithms to provide an initial solution to the proposed problem. In addition, generating exact solutions based on branch and bound methods will be investigated. Moreover, adopting the proposed algorithms in other drone-based domains, such as forest fire surveillance systems, will be considered.

ACKNOWLEDGMENT

The authors would like to thank the Deanship of Scientific Research at Majmaah University for supporting this work under Project R-2022-199.

REFERENCES

[1] B. Wen, Y. Pan, Y. Zhang, J. Liu, and M. Xia, "Does the exhaustion of resources drive land use changes? Evidence from the influence of coal resources-exhaustion on coal resources–based industry land use changes," *Sustainability*, vol. 10, no. 8, p. 2698, Aug. 2018.

[2] Special Working Session WCED, "World commission on environment and development," *Our Common Future*, vol. 17, no. 1, pp. 1–91, 1987.

[3] H. Lund, "Renewable energy strategies for sustainable development," *Energy*, vol. 32, no. 6, pp. 912–919, 2017.

[4] R. Foster, M. Ghassemi, and A. Cota, *Solar Energy: Renewable Energy and the Environment*. Boca Raton, FL, USA: CRC Press, 2009.

[5] J. Li, P. Liu, and Z. Li, "Optimal design of a hybrid renewable energy system with grid connection and comparison of techno-economic performances with an off-grid system: A case study of West China," *Comput. Chem. Eng.*, vol. 159, Mar. 2022, Art. no. 107657.

[6] H. L. Zhang, J. Baeyens, J. Degrève, and G. Cacères, "Concentrated solar power plants: Review and design methodology," *Renew. Sustain. Energy Rev.*, vol. 22, pp. 466–481, Jun. 2013.

[7] R. Bravo and D. Friedrich, "Two-stage optimisation of hybrid solar power plants," *Sol. Energy*, vol. 164, pp. 187–199, Apr. 2018.

[8] A. Azouzoute et al., "Developing a cleaning strategy for hybrid solar plants PV/CSP: Case study for semi-arid climate," *Energy*, vol. 228, Aug. 2021, Art. no. 120565.

[9] M. Jemmali, L. K. B. Melhim, A. Alourani, and M. M. Alam, "Equity distribution of quality evaluation reports to doctors in health care organizations," *PeerJ Comput. Sci.*, vol. 8, p. e819, Jan. 2022.

[10] M. Jemmali, "Projects distribution algorithms for regional development," *Adv. Distrib. Comput. Artif. Intell. J.*, vol. 10, pp. 293–305, 2021.

[11] M. Jemmali, "An optimal solution for the budgets assignment problem," *RAIRO-Oper. Res.*, vol. 55, no. 2, pp. 873–897, Mar. 2021.

[12] M. Alharbi and M. Jemmali, "Algorithms for investment project distribution on regions," *Comput. Intell. Neurosci.*, vol. 2020, pp. 1–13, Aug. 2020.

[13] M. Jemmali, "Budgets balancing algorithms for the projects assignment," *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 11, pp. 574–578, 2019.

[14] M. Jemmali, M. M. Otoom, and F. Fayez, "Max-min probabilistic algorithms for parallel machines," in *Proc. Int. Conf. Ind. Eng. Ind. Manage.*, Jan. 2020, pp. 19–24.

[15] M. Jemmali and H. Alquhayz, "Equity data distribution algorithms on identical routers," in *Proc. Int. Conf. Innov. Comput. Commun.* Cham, Switzerland: Springer, 2020, pp. 297–305.

[16] M. Jemmali, L. K. B. Melhim, and M. Alharbi, "Randomized-variants lower bounds for gas turbines aircraft engines," in *Proc. World Congr. Global Optim.* Cham, Switzerland: Springer, 2019, pp. 949–956.

[17] G. Christodoulou, A. Kovács, and R. van Stee, "A truthful constant approximation for maximizing the minimum load on related machines," *Theor. Comput. Sci.*, vols. 489–490, pp. 88–98, Jun. 2013.

[18] Y. Jiang, P. Zhou, and W. Zhou, "MapReduce machine covering problem on a small number of machines," *J. Combinat. Optim.*, vol. 38, no. 4, pp. 1066–1076, Nov. 2019.

[19] Y. He and Z. Tan, "Ordinal on-line scheduling for maximizing the minimum machine completion time," *J. Combinat. Optim.*, vol. 6, no. 2, pp. 199–206, 2002.

[20] H. Alquhayz and M. Jemmali, "Max-min processors scheduling," *Inf. Technol. Control*, vol. 50, no. 1, pp. 5–12, Mar. 2021.

[21] M. Alkhelaiwi, W. Boulila, J. Ahmad, A. Koubaa, and M. Driss, "An efficient approach based on privacy-preserving deep learning for satellite image classification," *Remote Sens.*, vol. 13, no. 11, p. 2221, Jun. 2021.

[22] S. Latif et al., "Deep learning for the industrial Internet of Things (IIoT): A comprehensive survey of techniques, implementation frameworks, potential applications, and future directions," *Sensors*, vol. 21, no. 22, p. 7518, 2021.

[23] F. A. Ghaleb et al., "Misbehavior-aware on-demand collaborative intrusion detection system using distributed ensemble learning for VANET," *Electronics*, vol. 9, no. 9, p. 1411, 2020.

[24] C. Feng et al., "Efficient and secure data sharing for 5G flying drones: A blockchain-enabled approach," *IEEE Netw.*, vol. 35, no. 1, pp. 130–137, Jan. 2021.

[25] S. B. Prathiba, G. Raja, A. K. Bashir, A. A. AlZubi, and B. Gupta, "SDN-assisted safety message dissemination framework for vehicular critical energy infrastructure," *IEEE Trans. Ind. Informat.*, vol. 18, no. 5, pp. 3510–3518, May 2022.

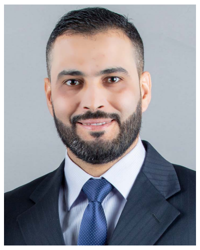
[26] M. Abdel-Basset, R. Mohamed, M. Elhoseny, A. K. Bashir, A. Jolfaei, and N. Kumar, "Energy-aware marine predators algorithm for task scheduling in IoT-based fog computing applications," *IEEE Trans. Ind. Informat.*, vol. 17, no. 7, pp. 5068–5076, Jul. 2021.

[27] R. H. Jhaveri, S. V. Ramani, G. Srivastava, T. R. Gadekallu, and V. Aggarwal, "Fault-resilience for bandwidth management in industrial software-defined networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 4, pp. 3129–3139, Oct. 2021.

[28] M. Jemmali, M. Denden, W. Boulila, G. Srivastava, R. H. Jhaveri, and T. R. Gadekallu, "A novel model based on window-pass preferences for data emergency aware scheduling in computer networks," *IEEE Trans. Ind. Informat.*, vol. 18, no. 11, pp. 7880–7888, Nov. 2022.

Mahdi Jemmali was born in Monastir, Tunisia. He received the Engineering degree from the National Computer Science School of Tunisia in 2003, the M.S. degree in mathematics engineering from the Polytechnic School of Tunisia in 2005, and the Ph.D. degree in computer science from the University of Tunis in 2011. He is currently an Associate Professor at the Department of Computer Science and Information, Majmaah University. His research interests include algorithms, scheduling, big data, smart city, operations management, machine learning, and artificial intelligence. He is attracted by the application of several scheduling techniques to solve and implement real-life hard problems from different domains.

Ali Kashif Bashir (Senior Member, IEEE) received the Ph.D. degree in computer science and engineering from Korea University, South Korea. He is currently a Senior Lecturer at the Department of Computing and Mathematics, Manchester Metropolitan University, U.K. He is also an Adjunct Professor at the National University of Science and Technology, Pakistan. His past assignments include an Associate Professor of ICT at the University of the Faroe Islands, Denmark; Osaka University, Japan; the Nara National College of Technology, Japan; the National Fusion Research Institute, South Korea; Southern Power Company Ltd., South Korea; and the Seoul Metropolitan Government, South Korea. He worked on several research and industrial projects of South Korean, Japanese, and European agencies and government ministries. He is also advising several start-ups in the field of STEM-based education, block chain, robotics, and smart homes. He has authored over 100 research articles and is supervising/co-supervising several graduate (M.S. and Ph.D.) students. His research interests include the Internet of Things, wireless networks, distributed systems, network/cyber security, and cloud/network function virtualization. He is an Invited Member of the IEEE Industrial Electronic Society, a member of ACM, and a Distinguished Speaker of ACM. He is serving as the Editor-in-Chief for the IEEE Future Directions Newsletter.



Wadii Boulila (Senior Member, IEEE) received the B.Eng. degree (Hons.) in computer science from the Aviation School of Borj El Amri in 2005, the M.Sc. degree in computer science from the National School of Computer Science (ENSI), University of Manouba, Tunisia, in 2007, and the Ph.D. degree in computer science conjointly from the ENSI and Telecom-Bretagne, University of Rennes 1, France, in 2012. He is currently an Associate Professor of computer science with Prince Sultan University, Saudi Arabia. He is also a Senior Researcher with the RIOTU Laboratory, Prince Sultan University; a Senior Researcher with the RIADI Laboratory, University of Manouba, and previously a Senior Research Fellow with the ITI Department, University of Rennes 1, France. He participated in many research and industrial-funded projects. His research interests include data science, big data analytics, deep learning, cybersecurity, artificial intelligence, uncertainty modeling, and image analysis and interpretation. He has served as the chair, a reviewer, and a TPC member for many leading international conferences and journals. He is an ACM Member and a Senior Fellow of the Higher Education Academy (SFHEA), U.K. He received the Award of the Young Researcher in computer science in Tunisia in 2021 from Beit El-Hikma and the Award of Best Researcher from the University of Manouba in 2021.

Loai Kayed B. Melhim, (Senior Member, IEEE) photograph and biography not available at the time of publication.

Rutvij H. Jhaveri (Senior Member, IEEE) received the Ph.D. degree in computer engineering in 2016. He is an Experienced Educator and a Researcher working with the Department of Computer Science and Engineering, Pandit Deendayal Energy University, Gandhinagar, India. He conducted his Postdoctoral Research at Delta-NTU Corporate Laboratory for Cyber-Physical Systems, Nanyang Technological University, Singapore. He was ranked among top 2% scientists around the world in 2021 and 2022. Currently, he is co-investigating a funded project from GUJCOST. Moreover, he has several national and international patents and, copyrights to his name. He has more than 2300 Google Scholar citations with H-index 25. His research interests include cyber security, the IoT systems, SDN, and smart healthcare. He authored more than 130 articles, including the IEEE/ACM TRANSACTIONS and flagship IEEE/ACM conferences. He also possesses memberships of various technical bodies, such as ACM, CSI, ISTE, and others. He is a member of the Advisory Board of the Symbiosis Institute of Digital and Telecom Management, and other reputed universities since 2022. He is an editorial board member in several Springer and Hindawi journals. He also served as a Committee Member in "Smart Village Project"-Government of Gujarat, at the district level during the year 2017. In 2017, he was awarded with the Prestigious Pedagogical Innovation Award by Gujarat Technological University. Apart from serving as an editor/guest editor for various journals of repute, he also serves as a reviewer for several international journals and also as an advisory/TPC member for renowned international conferences.

Jawad Ahmad (Senior Member, IEEE) is currently an Experienced Researcher with more than ten years of cutting-edge research and teaching experience in prestigious institutes, including Edinburgh Napier University, U.K.; Glasgow Caledonian University, U.K.; Hongik University, South Korea; and HITEC University, Taxila, Pakistan. He has coauthored more than 100 research papers, in international journals and peer-reviewed international conference proceedings. He has taught various courses both at undergraduate (UG) and postgraduate (PG) levels during his career. He regularly organizes timely special sessions and workshops for several flagship IEEE conferences. His research interests include cybersecurity, multimedia encryption, and machine learning and applications. He is an invited reviewer for numerous world-leading high-impact journals.