Article

Enabling Heterogeneous IoT Networks over 5G Networks with Ultra-Dense Deployment—Using MEC/SDN

Abdelhamied A. Ateya ^{1,*} , Abeer D. Algarni ², Monia Hamdi ², Andrey Koucheryavy ³
and Naglaa. F. Soliman ^{1,2,*}

¹ Department of Electronics and Communications Engineering, Zagazig University, Zagazig, Sharqia 44519, Egypt

² Department of Information Technology, College of Computer and Information Sciences, Princess Nourah Bint Abdulrahman University, Riyadh 84428, Saudi Arabia; adalgarni@pnu.edu.sa (A.D.A.); mshamdi@pnu.edu.sa (M.H.)

³ Department of Telecommunication Networks and Data Transmission, The Bonch-Bruевич Saint-Petersburg State University of Telecommunications, 193232 Saint Petersburg, Russia; akouch@mail.ru

* Correspondence: a_ashraf@zu.edu.eg (A.A.A.); nfsoliman@pnu.edu.sa (N.F.S.); Tel.: +55-798-1845-4131 (A.A.A.); +20-1005-237-673 (N.F.S.)

Abstract: The Internet of things (IoT) is the third evolution of the traditional Internet that enables interaction and communication among machines. Many IoT platforms and networks have been developed, and recently, market sectors have started to develop specific IoT applications and services. Integrating heterogeneous IoT networks with the existing ones, mainly with the cellular networks, is a great demand. IoT represents one of the main use cases of the fifth-generation (5G) cellular system as announced by the 3rd Generation Partnership Project (3GPP) and the International Telecommunication Union (ITU). Integrating IoT networks with 5G networks face many challenges related to dense deployment and a massive number of expected connected devices. Thus, IoT network availability and scalability are the main requirements that should be achieved. To this end, this work provides a framework for integrating heterogeneous IoT networks with the 5G networks. The proposed system considers dense deployment and system scalability and availability requirements as announced by ITU and 3GPP. Our proposed structure deploys three main communication paradigms; mobile edge computing (MEC), device-to-device communications (D2D), and software-defined networking (SDN). Our proposed system is evaluated over a reliable environment for various deployment scenarios, and the results validate the proposed structure. The proposed IoT/5G reduces the percentage of blocked tasks by an average of 30% than other traditional IoT networks. This increases the overall system availability and scalability since IoT networks can have more devices and tasks than existing IoT networks. Furthermore, our proposed structure reduces the overall consumed energy by an average of 20% than existing IoT networks, which is an effective metric for IoT networks.

Keywords: Internet of things; 5G; mobile edge computing; software-defined networking; device-to-device; scalability

Citation: Ateya, A.A.; Algarni, A.D.; Hamdi, M.; Koucheryavy, A.; Soliman, N.F. Enabling Heterogeneous IoT Networks over 5G Networks with Ultra-Dense Deployment—Using MEC/SDN. *Electronics* **2021**, *10*, 910. <https://doi.org/10.3390/electronics10080910>

Academic Editor: Michail J. Beliatas

Received: 12 March 2021

Accepted: 5 April 2021

Published: 11 April 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The Internet of things (IoT) is a recent communication paradigm considered to be the third evolution of the traditional Internet [1]. IoT is the communication system that considers connecting heterogeneous wireless devices and provides the communication medium for the interaction between machines [2]. As well as human-to-human (H2H) communication, IoT enables machine-to-machine (M2M) communication. M2M is the communication paradigm that has been introduced to enable the communications and interactions between machines, e.g., sensors and actuators [3]. Many market sectors consider developing IoT systems and applications, have been launched, and massive IoT applications in many fields have been developed [4]. However, there are many design

challenges associated with IoT networks and the integration of such networks with the existing and future cellular networks, e.g., fourth-generation (4G) and fifth-generation (5G) [5].

IoT represents a main use case of the 5G cellular system as announced by the third-generation partner project (3GPP) and the international telecommunication union (ITU) [6]. Thus, integrating existing IoT networks and platforms with the 5G systems is a great demand. However, this integration and, moreover, the design of IoT with the announced 5G specifications represent great challenges. In the era of 5G, IoT networks should achieve ultra-high availability and reliability [7]. Furthermore, IoT networks are expected to support ultra-low latency applications and achieve high system scalability. With the release of 5G, it is expected that the number of connected devices will be of the order of billions, which puts high constraints on the communication systems that provide the coverage for such a massive number [8].

The design of IoT networks, in the era of 5G, should provide a high level of scalability to be able to support the expected massive number of devices. To achieve high-level system scalability and the other introduced requirements, IoT networks should deploy new technologies, such as mobile edge computing (MEC), device-to-device communication (D2D) and software-defined networking (SDN) [9–11]. Furthermore, these technologies are used to enable the integration of IoT with the 5G systems.

MEC is a recent communication paradigm that has been introduced to move the computing capabilities and energy resources to the edge of the radio access network (RAN). This enables the paradigm shift from the centralized computing paradigm to the distributed computing paradigm [9]. Each cellular base station, i.e., and, is connected to a MEC server to provide the computing resources, e.g., processing and storage, one communication hop away from cellular users. The introduction of MEC servers at the edge of RAN unloads the core network by providing a way for traffic offloading and thus, increases the system scalability. Furthermore, since MEC servers can access network parameters, new services can be developed based on the extracted data [10].

SDN is another communication paradigm that is recently introduced for the core networks. SDN breaks the control and data planes and provides a way for managing the whole networks via a software-based control scheme [12]. The core networks deploy either single or multiple SDN controllers to manage and control the data plane and the RAN. The other network devices, e.g., switches and middleboxes, act as packet forwarding devices. Deploying SDN for cellular networks turns them into programmable networks and, thus, makes network evolution and innovation much easier [13].

D2D is a recent communication technology that has been introduced for enabling direct communication between devices [14]. The introduction of D2D communication to cellular networks unloads the RAN and also the core networks since a part of traffic can be handled without the involvement of the base station. Furthermore, the deployment of D2D communication achieves higher performance in terms of energy consumption, system scalability, and spectrum utilization [15].

This work considers developing a framework for an IoT/5G system, in which heterogeneous IoT networks can be integrated with the 5G cellular system. The system mainly considers the dense deployment of IoT devices and even cellular supported devices in 5G networks. The proposed system considers MEC, D2D and SDN technologies for such integration. Energy and the latency-efficient offloading scheme are developed for the proposed structure to manage data offloading among network parts in a way that achieves the previously mentioned requirements. Our main contributions can be summarized in the following points:

1. Integrating heterogeneous IoT networks with the 5G networks;
2. Enabling dense deployment of IoT networks;
3. Enabling dense deployment of 5G cellular systems;
4. Developing a framework for IoT networks able to support ultra-low latency applications;

5. Developing an energy-efficient offloading scheme that offloads computing tasks in a way that preserve devices' energy without affecting the quality of service (QoS);
6. Achieving high system scalability of IoT and 5G networks.

Paper organization: The article is structured as follows: Section 2 introduces the current literature and compares the related works with the proposed IoT/5G system. The novelty of the proposed work is well defined in this section. Section 3 presents the proposed framework for the IoT/5G system. The system structure is introduced, and the main parts are clearly discussed. Section 4 introduces the developed energy-aware, and latency-aware offloading scheme for the proposed IoT/5G system. Section 5 is introduced for performance evaluation and result in a discussion of the proposed system. Section 6 is introduced for conclusions and future directions.

2. Related Works

With the enormous increase in wireless and IoT devices, dense deployment becomes a major requirement of IoT networks. IoT is announced as the main use case of 5G systems as per 3GPP and ITU, and thus, the integration between existing IoT networks and cellular networks, mainly 5G systems, is a demand. These two challenges, i.e., scalability and integration with cellular systems, besides other existing network challenges of IoT networks, e.g., energy and availability, need to be considered. To overcome these challenges, recent communication paradigms have been introduced for heterogeneous IoT networks and applications that include MEC, SDN, D2D and Blockchain [16].

The existing literature considers a part of these challenges; however, this is for specific IoT networks. This work considers developing a framework for integrating IoT networks with 5G systems using MEC and SDN technologies. Our main contributions are clearly defined in the previous section. However, this section is introduced to illustrate the novelty of the proposed IoT/5G framework. Few studies have considered the integration of IoT networks with the 5G systems. In this section, we compare our proposed framework with the existing works by considering integrating IoT networks with cellular systems. Furthermore, since the dense deployment of IoT networks and energy efficiency are the main contributions of our proposed IoT framework, all recent existing works considering these objectives are included in the comparison.

Table 1 presents a comparison between existing related works and our proposed IoT/5G system. The comparison introduces the used key enabling technology for the IoT networks and the considered IoT networks, e.g., low-power wide-area networks (LP-WAN), LoRaWAN, or the work suitable for any IoT network. Furthermore, the considered performance evaluation metrics are introduced for each work.

Table 1. Internet of things (IoT)/5G system compared to existing related works.

Ref.	Key Enabling Technology			IoT Network	Application	Performance Metrics	Cellular Integration
	Fog/MEC	D2D	SDN				
[17]	✓	X	✓	Any	General	No evaluation	✓
[18]	✓	✓	X	Any	General	Latency Scalability	X
[15]	X	✓	X	Any	General	Energy	X
[19]	✓	X	X	Any	General	Offloading	X
[20]	✓	X	X	Any	Vehicle-to-grid	Data classification	✓
[21]	✓	X	X	Any	General	Completion time Energy	X
[22]	✓	X	X	Any	General	Energy	X
[23]	✓	X	X	Any	Smart home	Temporal delay	X

Table 1. Cont.

Ref.	Key Enabling Technology			IoT Network	Application	Performance Metrics	Cellular Integration
	Fog/MEC	D2D	SDN				
[24]	✓	X	✓	Any	General	Reliability Latency	X
[25]	✓	X	X	Any	General	Data analysis	X
[26]	X	✓	X	Any	General	Energy Security	X
[27]	✓	X	X	Any	Autonomous vehicles	Latency	X
[28]	✓	X	X	Any	Time-critical apps.	Latency	X
[29]	✓	X	X	Any	Medical	Energy	X
[30]	✓	X	X	Any	General	Latency Energy	X
[31]	✓	X	✓	Any	General	Security	X
[32]	✓	X	X	LoRaWAN	Low latency apps	Energy	X
[33]	✓	X	X	Any	General	Spectrum Data management	X
[34]	✓	X	X	LPWAN	General	Resource provisioning	X
[35]	✓	X	X	WAN-IoT	Time-sensitive	Resources	X
IoT/5G	✓	✓	✓	Any	General	Scalability Energy	✓

3. IoT/5G System Structure

The proposed system deploys D2D communications, MEC and SDN technologies to integrate heterogeneous IoT networks with 5G networks. Figure 1 provides an end-to-end system structure of the proposed IoT/5G. The system can be viewed into two main parts; radio access networks, RAN, and core networks, CN. RAN comprises three subsystems; heterogeneous IoT networks, D2D-based networks, and RAN of the cellular networks. However, core networks deploy an SDN network with a multi-controller scheme.

3.1. Radio Access Network (RAN)

3.1.1. Heterogeneous IoT Networks

Heterogeneous IoT networks, e.g., low-power wide-area networks (LPWAN), can be connected to the 5G cellular system via MEC servers, as presented in Figure 2. Each IoT gateway is connected to a MEC server via a fiber-wireless, FiWi, connection. The MEC server, deployed for IoT networks, is referred to as the I-MEC server. The introduction of MEC technology to the edge of the access networks of IoT networks achieves various benefits [16]. MEC servers, connected to IoT gateways, provide computing resources, e.g., processing and storage, to IoT devices; one communication hops away. This reduces the communication latency, increases the system availability and reduces the load on the core networks. Furthermore, the introduction of the I-MEC servers achieves higher efficiency in terms of energy, scalability and reliability. Since I-MEC servers provide computing resources near to end-devices, this reduces the load on the end devices, which reduces the consumed energy.

Figure 1. System structure of (IoT)/5G.

Figure 2. Heterogeneous IoT networks connected to 5G edge servers.

MEC servers provide resources for existing and upcoming devices and thus increase the overall system scalability. Furthermore, the edge server can be scaled in terms of resources, e.g., storage and processing, to handle more traffic of new devices.

Our main contribution is to integrate the heterogeneous IoT networks with the 5G cellular system, which is achieved efficiently by the proposed MEC-based structure introduced in Figure 2. The introduction of edge computing, i.e., I-MEC, to the access networks of the IoT networks provides an interface to the cellular networks since the I-MEC server is connected to nearby MEC servers of the cellular networks, which makes the IoT data available at the edge of the RAN of cellular networks.

For managing the data offloading process, we develop an energy-aware offloading scheme for the proposed MEC-based IoT structure. The proposed offloading scheme is based on our developed energy-aware offloading algorithm for multilevel MEC-based 5G systems, presented in [36]. The proposed offloading algorithm controls data offloading between IoT devices and I-MEC servers in a way that achieves higher energy efficiency and thus prolongs the lifetime of IoT devices. This comes with the announced 3GPP requirements of IoT as a use case of 5G [6]. The proposed offloading algorithm for IoT networks is presented in Section 4.1.

3.1.2. D2D Enabled Network

Heterogeneous sensors and other wireless devices may be deployed over 5G cellular networks. One way to enable these devices is to introduce an appropriate radio communication interface, besides the existing cellular interface, to unload the cellular networks. Furthermore, some of these devices, i.e., sensors and actuators, may not support the cellular interface, and thus, the need for another radio interface becomes a must. D2D is one of the best solutions to provide coverage for the distributed sensors and wireless devices [18]. This enables the dense deployment scenario required for 5G systems. Figure 3 illustrates the main structure of the D2D enabled networks. Distributed sensors and wireless devices that support D2D communications use the nearby devices that support cellular interface to forward their data to the cellular base station with the connected edge server.

Figure 3. Device-to-Device communication (D2D) based network.

Devices deployed in the proposed D2D networks are categorized into three main levels based on the device capabilities, e.g., processing and storage, and the radio communication interfaces supported by the device. The first level represents devices that support only D2D communication and does not support the cellular interface. These devices are referred to as low-level devices (LL). LL devices have limited resources, including energy, processing and storage. These devices include sensors and actuators. The second level represents devices that support either D2D communication interfaces only or both cellular and D2D interfaces. These devices are wireless devices with acceptable computing resources that are higher than that of LL devices. These devices are referred to as gateway devices (GW). GW devices can handle some computing tasks to the nearby LL devices based on the currently available resources and the current level of energy of GW devices. Furthermore, if the available resources are not enough to handle the offloaded tasks, GW devices act as a relay and forward the received tasks from LL to the higher level.

The last level of devices is the high-level devices (HL), which represents devices with powerful computing resources and energy resources. These devices must support both cellular and D2D interfaces. Such examples of these devices include smartphones, tablets and notebooks. HL devices receive offloading requests from the distributed GW devices and nearby LL devices and provide the appropriate response upon the available resources. If the currently available resources of the HL device are enough to handle the received offloading requests, the HL device accepts the offloading request and handles the computing task for GW or LL device. When the current resources of the HL device are not enough to handle the offloading request, the HL device sends the request to the MEC server connected to the cellular base station. The previously developed network setup algorithm presented in [19], is used for the proposed D2D-based networks. The required offloading algorithm for managing task offloading between devices in D2D-based networks is presented in Section 4.2. The algorithm is energy and latency-aware to achieve a required QoS latency for computing tasks and to achieve high-energy efficiency.

3.1.3. Cellular Network

This part represents the heterogeneous 5G cellular cells with the connected MEC servers. Each eNB is connected to an edge server to provide computing resources to the distributed devices in the cell. The proposed system deploys homogeneous MEC servers with equal resources. The MEC server deployed for a cellular call is referred to as C-MEC. I-MEC servers are connected directly to nearby C-MEC servers via high-speed fiber connections. C-MEC servers handle the offloaded tasks from the connected I-MEC servers, cellular devices and corresponding D2D networks. Based on the currently available resources and the QoS latency of the computing task, the C-MEC server handles the computing task or offloads it to the core networks cloud. The proposed offloading algorithm, introduced in Section 4.3, provides the main steps required to perform the energy-aware and latency-aware offloading.

3.2. Core Network (CN)

The core network, CN, deploys the SDN technology, where the control plane and the data plane are separated. The proposed system deploys multiple SDN controllers since a physical SDN controller handles only a limited number of requests at a time, and as previously introduced, an SDN network with a single controller is efficient only for small-scale networks. SDN controllers communicate and interact via OpenFlow protocol [37]. Distributed OpenFlow switches connect to the appropriated SDN controller and receive their flow tables over the appropriate OpenFlow version deployed for the networks. A core networks cloud with powerful computing capabilities is deployed at the core networks and connected to the control plane of SDN networks. Our previously proposed controller placement algorithm, introduced in [38], is deployed to dynamically define the optimum number of SDN controllers required for the networks based on the current status of net-

work traffic. Furthermore, the algorithm provides the optimum connections of OpenFlow switches with SDN controllers in a way that achieves the highest latency performance.

4. Energy-Aware and Latency-Aware Offloading Scheme

In this section, the proposed offloading algorithms for subnetworks are introduced. The proposed algorithms are built based on our previously developed energy-aware and latency-aware offloading algorithm developed for multilevel MEC structure, presented in [36]. Table 2 represents the notation of all used parameters and variables. This section is divided into three subsections. The first subsection introduces the developed offloading algorithm for IoT networks, while the second subsection presents the offloading scheme developed for D2D-based networks. The last subsection provides the offloading algorithm for the MEC-based cellular system.

Table 2. Key annotation.

Notation	Description
Z	Total size of computing task (in bits)
$N_{CPU-CYC-Bit}$	CPU cycles required for bit processing
$N_{CPU-CYC}$	Total number of processing cycles, i.e., CPU cycles, required to process a computing task
T_{QoS}	Quality of service latency of computing task, i.e., maximum allowable latency to handle a computing task
R_{IoT}	Resources of the IoT device allocated for a computing task
R_{LL}	Resources of the LL device allocated for a computing task
R_{GW}	Resources of the GW device allocated for a computing task
R_{HL}	Resources of the HL device allocated for a computing task
R_{I-MEC}	Resources of the I-MEC server allocated for a computing task
R_{C-MEC}	Resources of the C-MEC server allocated for a computing task
f_{IoT}	Total resources of the IoT device
f_{LL}	Total resources of the LL device
f_{GW}	Total resources of the GW device
f_{HL}	Total resources of the HL device
f_{I-MEC}	Total resources of the I-MEC server
f_{C-MEC}	Total resources of the C-MEC server
δ_{IoT}	Energy consumed per CPU cycle of the IoT device
δ_{LL}	Energy consumed per CPU cycle of the LL device
δ_{GW}	Energy consumed per CPU cycle of the GW device
δ_{HL}	Energy consumed per CPU cycle of the HL device
δ_{I-MEC}	Energy consumed per CPU cycle of the I-MEC server
δ_{C-MEC}	Energy consumed per CPU cycle of the C-MEC server
$E_{thr-IoT}$	Energy threshold level of the IoT device
E_{thr-LL}	Energy threshold level of the LL device
E_{thr-GW}	Energy threshold level of the GW device
E_{thr-HL}	Energy threshold level of the HL device
$E_{thr-I-MEC}$	Energy threshold level of the I-MEC server
$E_{thr-C-MEC}$	Energy threshold level of the C-MEC server
E_{C-IoT}	Energy of the IoT device after local execution
E_{C-LL}	Energy of the LL device after local execution
E_{C-GW}	Energy of the GW device after handling the requested task
E_{C-HL}	Energy of the HL device after handling the requested task
$E_{C-I-MEC}$	Energy of the I-MEC server after handling the requested task
$E_{C-C-MEC}$	Energy of the C-MEC server after handling the requested task
$E_{L-exc-IoT}$	Total energy required for local execution of current computing task at the IoT device
$E_{L-exc-LL}$	Total energy required for local execution of current computing task at the LL device
$E_{L-exc-GW}$	Total energy required to execute a computing task at the GW device
$E_{L-exc-HL}$	Total energy required to execute a computing task at the HL device
$E_{L-exc-I-MEC}$	Total energy required to execute a computing task at the I-MEC server
$E_{L-exc-C-MEC}$	Total energy required to execute a computing task at the C-MEC server
E_{IoT}	Energy of the IoT device before local execution
E_{LL}	Energy of the LL device before local execution

Table 2. Cont.

Notation	Description
E_{GW}	Energy of the GW device before handling the requested task
E_{HL}	Energy of the HL device before handling the requested task
E_{I-MEC}	Energy of the I-MEC server before handling the requested task
E_{C-MEC}	Energy of the C-MEC server before handling the requested task
E_{h-GW}	Total energy required to handle a computing task at the GW device
E_{h-HL}	Total energy required to handle a computing task at the HL device
$E_{h-I-MEC}$	Total energy required to handle a computing task at the I-MEC server
$E_{h-C-MEC}$	Total energy required to handle the computing task at the C-MEC server
$T_{exc-IoT}$	Total execution time of computing task at the IoT device
T_{exc-LL}	Total time required to execute a computing task at the LL device
T_{exc-GW}	Total time required to execute a computing task at the GW device
T_{exc-HL}	Total time required to execute a computing task at the HL device
$T_{exc-I-MEC}$	Total time required to execute the requested task at the I-MEC server
$T_{exc-C-MEC}$	Total time required to execute the requested task at the C-MEC server
T_{h-GW}	Total time required to handle the computing task at the GW device
T_{h-HL}	Total time required to handle the computing task at the HL device
$T_{h-I-MEC}$	Total time required to handle the computing task at the I-MEC server
$T_{h-C-MEC}$	Total time required to handle the computing task at the C-MEC server
T_{tx}	Total transmission time of input data of the computing task
T_{comm}	Total communication latency between cellular base station and end device
T_{rx}	Total feedback time of computation results
I	Mode Indicator Variable
$D_{Off-E-IoT}$	Binary energy decision of offloading; decided by the IoT device
$D_{Off-T-IoT}$	Binary time decision of offloading; decided by the IoT device
$D_{off-IoT}$	Binary decision of offloading, decided by the IoT device
$D_{Off-E-I-MEC}$	Binary energy decision of offloading; decided by the I-MEC server
$D_{Off-T-I-MEC}$	Binary time decision of offloading; decided by the I-MEC server
$D_{Off-I-MEC}$	Binary decision of offloading, decided by the I-MEC server
$D_{Off-E-C-MEC}$	Binary energy decision of offloading; decided by the C-MEC server
$D_{Off-T-C-MEC}$	Binary time decision of offloading; decided by the C-MEC server
$D_{Off-C-MEC}$	Binary decision of offloading, decided by the C-MEC server
$D_{Off-E-LL}$	Binary energy decision of offloading; decided by the LL device
$D_{Off-T-LL}$	Binary time decision of offloading; decided by the LL device
D_{Off-LL}	Binary decision of offloading, decided by the LL device
$D_{Off-E-GW}$	Binary energy decision of offloading; decided by the GW device
$D_{Off-T-GW}$	Binary time decision of offloading; decided by the GW device
D_{Off-GW}	Binary decision of offloading, decided by the GW device
$D_{Off-E-HL}$	Binary energy decision of offloading; decided by the HL device
$D_{Off-T-HL}$	Binary time decision of offloading; decided by the HL device
D_{Off-HL}	Binary decision of offloading, decided by the HL device
P_{IoT}	Transmitting power of the IoT device
P_S	Transmitting power of the IoT gateway connected to the I-MEC server
P_{LL}	Transmitting power of the LL device
P_{GW}	Transmitting power of the GW device
P_{HL}	Transmitting power of the HL device
η_{IoT}	Channel efficiency of the IoT network
η_S	Channel efficiency of the IoT-cellular
η_{D2D}	Channel efficiency of D2D network
η_C	Channel efficiency of cellular interface

4.1. Energy-Aware and Latency-Aware Offloading Algorithm for IoT Networks

The proposed offloading algorithm consists of three offloading levels besides the local execution, as presented in Figure 4. The zero-level offloading, which corresponds to the local execution. This represents tasks that can be executed by the end devices without any need for the offloading process. This depends on the currently available resources of end-devices and the required QoS of the task. The first level of offloading represents

the I-MEC servers. Tasks that cannot be executed by the end-devices are offloaded via the appropriate communication link to the I-MEC server connected to the IoT gateway. The I-MEC server accepts the offloading process or rejects it based on the currently available resources at the I-MEC server and the required QoS of the offloaded task. If the currently available resources of the I-MEC server are not enough to handle the offloaded task, the task is then offloaded to the higher level, i.e., the second level of offloading.

Figure 4. Main offloading levels of heterogeneous IoT networks.

The second offloading level is represented by the cellular MEC server, i.e., C-MEC. I-MEC server offloads the unhandled tasks to the connected C-MEC server. C-MEC server accepts the offloaded task or rejects it based on the currently available resources of end-devices and the required QoS of the task. Tasks that cannot be handled by the C-MEC server are offloaded to the final offloading level represented by the core network cloud. Algorithm 1 presents the developed energy-aware, a latency-aware offloading algorithm for MEC-based IoT networks.

End devices decide the main offloading decision based on the currently available resources, e.g., storage, processing and energy, at the device and the required QoS latency of the current task. Each device uses the program profile to calculate the size of the computing task, Z , and the required number of processing cycles, $N_{CPU-CYC}$ to handle the task. The decision engine of the end device is the hardware that implements our proposed offloading algorithm. It receives the information from the program profiler, i.e., Z and $N_{CPU-CYC}$, and the information of the currently available resources from the resources scheduler. Furthermore, the decision engine receives the required QoS latency, T_{QoS} , of the current computing task, which represents the maximum allowable latency to handle the computing task. The decision engine uses these parameters to decide either the local execution or offloading.

Algorithm 1 Latency-aware and energy-aware offloading algorithm for MEC-based IoT networks.

1:	Initialize $T_{QoS}, E_{thr-IoT}, E_{thr-I-MEC}$	
2:	Calculate $Z, N_{CPU-CYC-Bit}$	
3:	Calculate $T_{exc-IoT}$	
4:	If $(T_{exc-IoT} \leq T_{QoS})$	
5:		$D_{Off-T-IoT} = 0$
6:		Calculate $E_{L-exc-IoT}, E_{C-IoT}$
7:		If $(E_{C-IoT} > E_{thr-IoT})$
8:		$D_{Off-E-IoT} = 0$
9:		else
10:		$D_{Off-E-IoT} = 1$

```

11:                                     end if
12:     else
13:                                      $D_{Off-T-IoT} = 1$ 
14:     end if
15:      $D_{off-IoT} = D_{Off-T-IoT} \wedge D_{Off-E-IoT}$ 
16:     If ( $D_{off-IoT} == 0$ )
17:         Handle task locally
18:     else
19:         Offload Task to I-MEC server
20:     end if
21:     Calculate  $T_{exc-I-MEC}, T_{h-I-MEC}$ 
22:     If ( $T_{h-I-MEC} \leq T_{QoS}$ )
23:                                      $D_{Off-T-I-MEC} = 0$ 
24:                                     Calculate  $E_{l-exc-I-MEC}, E_{h-I-MEC}, E_{C-I-MEC}$ 
25:                                     If ( $E_{C-I-MEC} > E_{thr-I-MEC}$ )
26:                                      $D_{Off-E-I-MEC} = 0$ 
27:                                     else
28:                                      $D_{Off-E-I-MEC} = 1$ 
29:                                     end if
30:     else
31:                                      $D_{Off-T-I-MEC} = 1$ 
32:     end if
33:      $D_{Off-I-MEC} = D_{Off-T-I-MEC} \wedge D_{Off-E-I-MEC}$ 
34:     If ( $D_{Off-I-MEC} == 0$ )
35:         I-MEC server accepts offloading request, and handles the computing task.
36:     else
37:         Offload Task to C-MEC server
38:     end if

```

At first, the decision engine calculates the total time required to execute the task locally, $T_{exc-IoT}$, at the IoT device. This time is calculated based on the currently available resources at the IoT device according to (1). Then, the decision engine makes the energy and time decisions of offloading, $D_{Off-E-IoT}$ and $D_{Off-T-IoT}$. The energy decision is calculated by comparing the residual energy of IoT device after task execution, E_{C-IoT} , with the threshold energy, $E_{thr-IoT}$, of IoT device. The threshold energy is predefined for IoT devices in a way to prolong the lifetime of IoT devices. If the residual energy of the IoT device after the local execution is higher than the threshold level of energy, the energy offloading decision is set to zero, and the local execution is recommended from the energy perspective. For a positive offloading decision, the task is directly offloaded to the I-MEC server. For a negative energy offloading decision, the time offloading decision is calculated. The time decision of offloading, $D_{Off-T-IoT}$, is calculated by comparing the execution time, $T_{exc-IoT}$, with the QoS latency, T_{QoS} , as in (6).

$$T_{exc-IoT} = \frac{N_{CPU-CYC}}{R_{IoT}}, \quad R_{IoT} \in f_{IoT}, \quad (1)$$

$$N_{CPU-CYC} = Z \cdot N_{CPU-CYC-Bit}, \quad (2)$$

$$E_{L-exc-IoT} = N_{CPU-CYC} \delta_{IoT}, \quad (3)$$

$$E_{C-IoT} = E_{IoT} - E_{L-exc-IoT}, \quad (4)$$

$$D_{Off-E-IoT} = I(E_{C-IoT}, E_{thr-IoT}) = \begin{cases} 1 & \text{IF } (E_{C-IoT} \leq E_{thr-IoT}) \\ 0 & \text{IF } (E_{C-IoT} > E_{thr-IoT}) \end{cases}, \quad (5)$$

$$D_{Off-T-IoT} = I(T_{exc-IoT}, T_{QoS}) = \begin{cases} 0 & \text{IF } (T_{exc-IoT} \leq T_{QoS}) \\ 1 & \text{IF } (T_{exc-IoT} > T_{QoS}) \end{cases}, \quad (6)$$

$$D_{Off-IoT} = D_{Off-E-IoT} \vee D_{Off-T-IoT}, \quad (7)$$

If the decision engine decides to offload, i.e., $D_{Off-IoT} = 1$, the computing task is offloaded to the I-MEC server. The I-MEC server deploys the same server structure developed in [36]. The decision engine of the I-MEC server receives the offloading request from the IoT device and decides either to handle the task at the I-MEC server or to offload the task to the higher offloading level, i.e., C-MEC server. The I-MEC server's decision engine calculates the execution time required to execute the requested task, $T_{exc-I-MEC}$, and then the total time required to handle it, $T_{h-I-MEC}$ [36].

$$T_{exc-I-MEC} = \frac{N_{CPU-CYC}}{R_{I-MEC}}, \quad R_{I-MEC} \in f_{I-MEC}, \quad (8)$$

$$T_{h-I-MEC} = T_{exc-I-MEC} + T_{tx} + T_{rx}, \quad (9)$$

The decision engine of the I-MEC server calculates the total energy consumed to handle the requested task at the I-MEC server, $E_{h-I-MEC}$.

$$E_{L-exc-I-MEC} = N_{CPU-CYC} \delta_{I-MEC}, \quad (10)$$

$$E_{h-I-MEC} = E_{L-exc-I-MEC} + T_{tx} P_{IoT} \eta_{IoT}, \quad (11)$$

$$E_{C-I-MEC} = E_{I-MEC} - E_{h-I-MEC}, \quad (12)$$

Then, the I-MEC server's decision engine calculates the binary energy and time decision values of offloading, $D_{Off-E-I-MEC}$ and $D_{Off-T-I-MEC}$. The energy decision is calculated by comparing the residual energy, after handling the requested task, with the threshold level of energy, $E_{thr-I-MEC}$, as presented in (13). However, the time decision is calculated by comparing the total time required to handle the requested task with the required QoS latency, T_{QoS} [36,39].

$$D_{Off-E-I-MEC} = I(E_{C-I-MEC}, E_{thr-I-MEC}) = \begin{cases} 1 & \text{IF } (E_{C-I-MEC} \leq E_{thr-I-MEC}) \\ 0 & \text{IF } (E_{C-I-MEC} > E_{thr-I-MEC}) \end{cases}, \quad (13)$$

$$D_{Off-T-I-MEC} = I(T_{h-I-MEC}, T_{QoS}) = \begin{cases} 0 & \text{IF } (T_{h-I-MEC} \leq T_{QoS}) \\ 1 & \text{IF } (T_{h-I-MEC} > T_{QoS}) \end{cases}, \quad (14)$$

$$D_{Off-I-MEC} = D_{Off-E-I-MEC} \vee D_{Off-T-I-MEC}, \quad (15)$$

For a negative offloading decision, i.e., $D_{Off-I-MEC} = 0$, the task is handled at the I-MEC server. However, for a positive offloading decision, i.e., $D_{Off-I-MEC} = 1$, the task is offloaded to the higher level, i.e., C-MEC server [36,40].

Once the C-MEC server receives an offloading request, the decision engine of the server performs similar calculations as the I-MEC server. At first, the total time and energy required to handle the requested task, $T_{h-C-MEC}$ and $E_{h-C-MEC}$, is calculated.

$$T_{exc-C-MEC} = \frac{N_{CPU-CYC}}{R_{C-MEC}}, \quad R_{C-MEC} \in f_{C-MEC}, \quad (16)$$

$$T_{h-C-MEC} = T_{exc-C-MEC} + T_{tx} + T_{rx}, \quad (17)$$

$$E_{L-exc-C-MEC} = N_{CPU-CYC} \delta_{C-MEC}, \quad (18)$$

$$E_{h-C-MEC} = E_{L-exc-C-MEC} + T_{tx} P_S \eta_S, \quad (19)$$

$$E_{C-C-MEC} = E_{C-MEC} - E_{h-C-MEC}, \quad (20)$$

The C-MEC server's decision engine makes the energy and time decisions of offloading, $D_{Off-E-C-MEC}$ and $D_{Off-T-C-MEC}$, according to (21, 22).

$$D_{Off-E-C-MEC} = I(E_{C-C-MEC}, E_{thr-C-MEC}) = \begin{cases} 1 & \text{IF } (E_{C-C-MEC} \leq E_{thr-C-MEC}) \\ 0 & \text{IF } (E_{C-C-MEC} > E_{thr-C-MEC}) \end{cases}, \quad (21)$$

$$D_{\text{Off-T-C-MEC}} = I(T_{h-C-MEC}, T_{QoS}) = \begin{cases} 0 & \text{IF } (T_{h-C-MEC} \leq T_{QoS}) \\ 1 & \text{IF } (T_{h-C-MEC} > T_{QoS}) \end{cases}, \quad (22)$$

$$D_{\text{Off-C-MEC}} = D_{\text{Off-E-C-MEC}} \vee D_{\text{Off-T-C-MEC}}, \quad (23)$$

For a negative offloading decision, the task is handled at the edge server of cellular networks, i.e., C-MEC. Otherwise, the task is offloaded to the core networks cloud.

4.2. Energy-Aware and Latency-Aware Offloading Algorithm for D2D Networks

Algorithm 2 presents the developed energy-aware and latency-aware offloading algorithm for D2D networks deployed over a 5G cellular systems. The offloading algorithm introduces four levels of offloading besides local execution. Figure 5 illustrates the main offloading levels for D2D-based networks. The zero levels represent the local execution, where computing tasks are executed locally at the task generated device, e.g., LL, GW and HL. If the task-generated device has enough resources to handle the generated computing task within the announced QoS latency, the task is executed locally without any need for task offloading.

Figure 5. Main offloading levels of device-to-device communications (D2D) networks.

However, if there are no available resources for local execution, the task is offloaded to the higher offloading level. There are four available offloading levels. Each of them corresponds to an executing device. The first level represents the offloading to the GW device, while the second offloading level represents the offloading to the HL device. The third offloading level is the edge server, i.e., C-MEC server, while the last level is the core networks cloud. Figure 6 illustrates heterogenous task handling and offloading over a D2D network, using the proposed offloading scheme [18].

Algorithm 2 Latency-aware and energy-aware offloading algorithm for D2D-based networks.

```

1:   Initialize  $T_{QoS}$ ,  $E_{thr-LL}$ ,  $E_{thr-GW}$ ,  $E_{thr-HL}$ ,  $E_{thr-C-MEC}$ 
2:   Calculate  $Z$ ,  $N_{CPU-CYC-Bit}$ 
3:   Calculate  $T_{exc-LL}$ 
4:   If ( $T_{exc-LL} \leq T_{QoS}$ )
5:        $D_{Off-T-LL} = 0$ 
6:       Calculate  $E_{L-exc-LL}$ ,  $E_{C-LL}$ 
7:       If ( $E_{C-LL} > E_{thr-LL}$ )
8:            $D_{Off-E-LL} = 0$ 
9:       else
10:           $D_{Off-E-LL} = 1$ 
11:      end if
12:  else
13:       $D_{Off-T-LL} = 1$ 
14:  end if
15:   $D_{off-LL} = D_{Off-T-LL} \wedge D_{Off-E-LL}$ 
16:  If ( $D_{off-LL} == 0$ )
17:      Handle task locally
18:  else
19:      Offload Task to nearby GW device
20:  end if
21:  Calculate  $T_{exc-GW}$ ,  $T_{h-GW}$ 
22:  If ( $T_{h-GW} \leq T_{QoS}$ )
23:       $D_{Off-T-GW} = 0$ 
24:      Calculate  $E_{L-exc-GW}$ ,  $E_{h-GW}$ ,  $E_{C-GW}$ 
25:      If ( $E_{C-GW} > E_{thr-GW}$ )
26:           $D_{Off-E-GW} = 0$ 
27:      else
28:           $D_{Off-E-GW} = 1$ 
29:      end if
30:  else
31:       $D_{Off-T-GW} = 1$ 
32:  end if
33:   $D_{Off-GW} = D_{Off-T-GW} \wedge D_{Off-E-GW}$ 
34:  If ( $D_{Off-GW} == 0$ )
35:      GW device accepts offloading request, and handles the computing task.
36:  else
37:      Offload Task to nearby HL device
38:  end if
39:  Calculate  $T_{exc-HL}$ ,  $T_{h-HL}$ 
40:  If ( $T_{h-HL} \leq T_{QoS}$ )
41:       $D_{Off-T-HL} = 0$ 
42:      Calculate  $E_{L-exc-HL}$ ,  $E_{h-HL}$ ,  $E_{C-HL}$ 
43:      If ( $E_{C-HL} > E_{thr-HL}$ )
44:           $D_{Off-E-HL} = 0$ 
45:      else
46:           $D_{Off-E-HL} = 1$ 
47:      end if
48:  else
49:       $D_{Off-T-HL} = 1$ 
50:  end if
51:   $D_{Off-HL} = D_{Off-T-HL} \wedge D_{Off-E-HL}$ 
52:  If ( $D_{Off-HL} == 0$ )
53:      HL device accepts offloading request, and handles the computing task.
54:  else
55:      Offload Task to C-MEC server
56:  end if

```

Figure 6. Heterogenous task handling and offloading over a D2D network.

The offloading algorithm for D2D-based networks has similar steps as that of IoT networks. The end device, e.g., the LL device, decides either local execution or offloading. The decision engine of the end device calculates the required time and energy for handling the task locally, T_{exc-LL} and $E_{L-exc-LL}$, based on the currently available resources.

$$T_{exc-LL} = \frac{N_{CPU-CYC}}{R_{LL}}, \quad R_{LL} \in f_{LL}, \quad (24)$$

$$E_{L-exc-LL} = N_{CPU-CYC} \delta_{LL}, \quad (25)$$

$$E_{C-LL} = E_{LL} - E_{L-exc-LL}, \quad (26)$$

$$D_{Off-E-LL} = I(E_{C-LL}, E_{thr-LL}) = \begin{cases} 1 & \text{IF } (E_{C-LL} \leq E_{thr-LL}) \\ 0 & \text{IF } (E_{C-LL} > E_{thr-LL}) \end{cases}, \quad (27)$$

$$D_{Off-T-LL} = I(T_{exc-LL}, T_{QoS}) = \begin{cases} 0 & \text{IF } (T_{exc-LL} \leq T_{QoS}) \\ 1 & \text{IF } (T_{exc-LL} > T_{QoS}) \end{cases}, \quad (28)$$

$$D_{Off-LL} = D_{Off-E-LL} \vee D_{Off-T-LL}, \quad (29)$$

For a positive offloading decision, the LL device offloads the computing task to the first offloading level, i.e., GW device. The GW device receives the offloading request and processes it. The decision engine of the GW device decides either to handle the offloaded task at the GW device or to offload it to the higher level, i.e., the HL device. GW performs similar steps as LL device to make the decision according to (30–37)

$$T_{exc-GW} = \frac{N_{CPU-CYC}}{R_{GW}}, \quad R_{GW} \in f_{GW}, \quad (30)$$

$$T_{h-GW} = T_{exc-GW} + T_{tx} + T_{rx}, \quad (31)$$

$$E_{L-exc-GW} = N_{CPU-CYC} \delta_{GW}, \quad (32)$$

$$E_{h-GW} = E_{L-exc-GW} + T_{tx} P_{LL} \eta_{D2D}, \quad (33)$$

$$E_{C-GW} = E_{C-GW} - E_{h-GW}, \quad (34)$$

$$D_{Off-E-GW} = I(E_{C-GW}, E_{thr-GW}) = \begin{cases} 1 & \text{IF } (E_{C-GW} \leq E_{thr-GW}) \\ 0 & \text{IF } (E_{C-GW} > E_{thr-GW}) \end{cases}, \quad (35)$$

$$D_{Off-T-LL} = I(T_{exc-GW}, T_{QoS}) = \begin{cases} 0 & \text{IF } (T_{exc-GW} \leq T_{QoS}) \\ 1 & \text{IF } (T_{exc-GW} > T_{QoS}) \end{cases}, \quad (36)$$

$$D_{Off-GW} = D_{Off-E-GW} \bigvee D_{Off-T-GW}, \quad (37)$$

If the GW device fails to handle the requested task, it offloads the task to the nearby HL device. HL device receives the offloading request and processes it to either handle the requested task or offload it to the C-MEC server. The decision engine performs similar steps as LL and GW devices to make the energy and time decisions of offloading.

$$T_{exc-HL} = \frac{N_{CPU-CYC}}{R_{HL}}, \quad R_{HL} \in f_{HL}, \quad (38)$$

$$T_{h-HL} = T_{exc-HL} + T_{tx} + T_{rx}, \quad (39)$$

$$E_{L-exc-HL} = N_{CPU-CYC} \delta_{HL}, \quad (40)$$

$$E_{h-HL} = E_{L-exc-HL} + T_{tx} P_{GW} \eta_{D2D}, \quad (41)$$

$$E_{C-HL} = E_{C-HL} - E_{h-HL}, \quad (42)$$

$$D_{Off-E-HL} = I(E_{C-HL}, E_{thr-HL}) = \begin{cases} 1 & \text{IF } (E_{C-HL} \leq E_{thr-HL}) \\ 0 & \text{IF } (E_{C-HL} > E_{thr-HL}) \end{cases}, \quad (43)$$

$$D_{Off-T-HL} = I(T_{exc-HL}, T_{QoS}) = \begin{cases} 0 & \text{IF } (T_{exc-HL} \leq T_{QoS}) \\ 1 & \text{IF } (T_{exc-HL} > T_{QoS}) \end{cases}, \quad (44)$$

$$D_{Off-HL} = D_{Off-E-HL} \bigvee D_{Off-T-HL}, \quad (45)$$

For a positive offloading decision of the HL device, i.e., $D_{Off-HL} = 1$, the task is offloaded to the C-MEC server at the edge of the RAN.

4.3. Energy-Aware and Latency-Aware Offloading Algorithm for Cellular Network

Once a C-MEC server receives an offloading request from any connected devices, it makes certain processes based on Algorithm 3 to decide whether to handle or offload the received computing task. The C-MEC server has the same structure proposed in [36]. At first, the profiler extracts the task data from the received request. These data include the QoS latency, T_{QoS} , the task size in bits, Z , and the communication latency, T_{comm} , defined by the connected base station. The profiler passes these data to the C-MEC controller, which also receives the information about the current energy level and the available resources of the C-MEC from the resource scheduler. The controller takes its decision of handling or offloading the requested task based on the received data. The C-MEC controller calculates the total energy consumption for handling the task, $E_{h-C-MEC}$, and then calculates the energy level of the C-MEC, $E_{C-C-MEC}$, if the task is handled. If the remaining energy, $E_{C-C-MEC}$, after handling the task, is higher than the predefined threshold level of energy of C-MEC, $E_{thr-C-MEC}$, the controller then checks the latency to make the time decision of offloading. However, if the remaining energy is less than the pre-announced threshold level, the controller decides to offload the computing task to the core network cloud.

$$T_{exc-C-MEC} = \frac{N_{CPU-CYC}}{R_{C-MEC}}, \quad R_{C-MEC} \in f_{C-MEC}, \quad (46)$$

$$T_{h-C-MEC} = T_{exc-C-MEC} + T_{tx} + T_{rx}, \quad (47)$$

$$E_{L-exc-C-MEC} = N_{CPU-CYC} \delta_{C-MEC}, \quad (48)$$

$$E_{h-C-MEC} = E_{L-exc-C-MEC} + T_{tx} P_{HL} \eta_C, \quad (49)$$

$$E_{C-C-MEC} = E_{C-C-MEC} - E_{h-C-MEC}, \quad (50)$$

$$D_{Off-E-C-MEC} = I(E_{C-C-MEC}, E_{thr-C-MEC}) = \begin{cases} 1 & \text{IF } (E_{C-C-MEC} \leq E_{thr-C-MEC}) \\ 0 & \text{IF } (E_{C-C-MEC} > E_{thr-C-MEC}) \end{cases}, \quad (51)$$

$$D_{Off-T-C-MEC} = I(T_{exc-C-MEC}, T_{QoS}) = \begin{cases} 0 & \text{IF } (T_{exc-C-MEC} \leq T_{QoS}) \\ 1 & \text{IF } (T_{exc-C-MEC} > T_{QoS}) \end{cases}, \quad (52)$$

$$D_{Off-C-MEC} = D_{Off-E-C-MEC} \vee D_{Off-T-C-MEC}, \quad (53)$$

Algorithm 3 Latency-aware and energy-aware offloading algorithm for mobile edge computing (MEC)-based cellular networks.

```

1:   Initialize  $E_{thr-C-MEC}$ 
2:   Calculate  $T_{exc-C-MEC}$ ,  $T_{h-C-MEC}$ 
3:   If  $(T_{h-C-MEC} \leq T_{QoS})$ 
4:        $D_{Off-T-C-MEC} = 0$ 
5:       Calculate  $E_{l-exc-C-MEC}$ ,  $E_{h-C-MEC}$ ,  $E_{C-C-MEC}$ 
6:       If  $(E_{C-C-MEC} > E_{thr-C-MEC})$ 
7:            $D_{Off-E-C-MEC} = 0$ 
8:       else
9:            $D_{Off-E-C-MEC} = 1$ 
10:      end if
11:  else
12:       $D_{Off-T-C-MEC} = 1$ 
13:  end if
14:   $D_{Off-C-MEC} = D_{Off-T-C-MEC} \wedge D_{Off-E-C-MEC}$ 
15:  If  $(D_{Off-C-MEC} == 0)$ 
16:      C-MEC server accepts offloading request, and handles the computing task.
17:  else
18:      Offload Task to CN
19:  end if

```

5. Performance Evaluation

In this part, the proposed system and the developed algorithms are evaluated over reliable environments. Various scenarios are considered to indicate the performance. The section is divided into two subsections; simulation setup and simulation results. In the first subsection, the main steps of the simulation setup process are introduced, and the simulation parameters are defined. The considered simulation scenarios are well defined. The second subsection discusses the obtained results and the considered performance metrics.

5.1. Simulation Setup

For performance evaluation of the proposed system and the comprised algorithms, an event-driven simulator is developed based on [13,18]. The simulator is implemented using Java. The system is simulated for the topology illustrated in Figure 7. The IoT network considered for the simulation purposes is the LoRaWAN, which represents the most common and efficient LPWAN network [40]. LoRaWAN is a wide range IoT network that has been used widely for many applications [41]. The main reason for considering LoRaWAN as the IoT network in the simulation is the market use since LoRa is the most used IoT network [42]. The gateway of the LoRaWAN networks is connected to an I-MEC server to enable the edge services. The main characteristics of the considered LoRaWAN and the connected I-MEC server are indicated in Table 3.

Figure 7. Topology of the simulated network.

Table 3. Simulation parameters of LoRaWAN networks.

Parameter	Value
Area of the network	10 Km
Number of IoT devices	300
Number of IoT GWs	5
f_{IoT}	$\in [0.1, 5.0]$ GHz
f_{I-MEC}	$\in [2.0, 5.0]$ GHz
δ_{IoT}	1.2 J/GHz
δ_{C-MEC}	1.2 J/GHz
P_{IoT}	22 dBm
P_S	22 dBm
I-MEC Storage resources	16 GB
$E_{thr-IoT}$	35% $E_{Full-Battery-IoT}$
$E_{thr-I-MEC}$	25% $E_{Full-Battery-I-MEC}$
$E_{Full-Battery-IoT}$	5 KJ
$E_{Full-Battery-I-MEC}$	25 KJ

A cellular network of one cell is considered for simulation purposes, with a base station centered at the cell. The base station is connected to the C-MEC edge server to provide computing capabilities to the end-users of the cellular cell. Heterogeneous devices, e.g., smartphones and tablets, are distributed randomly among the cell with the specifications introduced in Table 4. All these devices are assumed to support a D2D interface, wi-fi direct. Wi-fi direct is selected because it is the most common D2D interface with many advantages [43].

The full battery energy of heterogeneous devices of D2D networks and 5G cellular networks is selected as that of real existing devices. HL devices are assumed to be powerful smartphones, e.g., iPhone x and tablets. These devices have a lithium-ion battery (Li-ion) with a capacity of around 3000 mAh at a rate of 3.80 volt-DC. Thus, the full battery energy of such devices is around 20 KJ, and similar steps are performed for the better assumption of the full battery energy of GW and LL devices. The channel parameters are set as in [13].

Table 4. Simulation parameters of the radio access network (RAN) of a cellular network.

Parameter	Value
Cell radius	1 km
Number of LL devices	30
Number of GW devices	10
Number of HL devices	5
f_{LL}	$\in [0.1, 5.0]$ GHz
f_{GW}	$\in [0.5, 1.0]$ GHz
f_{HL}	$\in [1.0, 2.0]$ GHz
f_{C-MEC}	$\in [4.0, 10.0]$ GHz
$f_{Core-networks}$	$\in [10.0, 30.0]$ GHz
δ_{LL}	1.2 J/GHz
δ_{GW}	1.15 J/GHz
δ_{HL}	1.1 J/GHz
δ_{C-MEC}	1.0 J/GHz
P_{LL}	22 dBm
P_{GW}	22 dBm
P_{HL}	22 dBm
D2D distance	20–80 m
C-MEC Storage resources	64 Gb
E_{thr-LL}	30% $E_{Full-Battery-LL}$
E_{thr-GW}	30% $E_{Full-Battery-GW}$
E_{thr-HL}	30% $E_{Full-Battery-HL}$
$E_{thr-C-MEC}$	20% $E_{Full-Battery-C-MEC}$
$E_{Full-Battery-LL}$	8 KJ
$E_{Full-Battery-GW}$	14 KJ
$E_{Full-Battery-HL}$	20 KJ
$E_{Full-Battery-C-MEC}$	100 KJ

The core network deploys a core network cloud with the SDN networks. The SDN networks are deployed with the same specification as in [13]. The considered OpenFlow version in the simulation process is OpenFlow 1.4 [44,45].

For IoT networks, each IoT device is assumed to have a computing task from the defined computing tasks, illustrated in Table 5. The assumed tasks are heterogeneous and selected to cover a wide range of IoT applications. Table 6 illustrates the specifications of LL devices' tasks. Each LL device is assumed to have one of the computing tasks introduced in Table 6. Tables 7 and 8 introduce task specifications of GW and HL devices, respectively. Each GW device is assumed to have a computing task from the tasks presented in Table 7, while HL devices have one task for each device; from the tasks presented in Table 8. All considered computing tasks are equivalent to the workloads of real tasks, such as processing web pages.

Table 5. Task specifications of IoT networks.

Task	Task (1)	Task (2)	Task (3)	Task (4)	Task (5)
Z (KB)	0.5	0.6	0.7	0.8	0.9
T_{QoS1} (ms)	1.0	1.0	1.1	1.1	1.2
T_{QoS2} (ms)	1.5	1.6	1.7	1.8	1.9
T_{QoS3} (ms)	2.0	2.2	2.4	2.6	2.8
Task	Task (6)	Task (7)	Task (8)	Task (9)	Task (10)
K (MB)	1.0	1.1	1.2	1.3	1.4
T_{QoS1} (ms)	1.3	1.4	1.4	1.5	1.5
T_{QoS2} (ms)	2.0	2.2	2.4	2.6	2.8
T_{QoS3} (ms)	3.0	3.1	3.2	3.3	3.4

Table 6. Task specifications of low-level devices (LL).

Task	Task (1)	Task (2)	Task (3)	Task (4)	Task (5)
Z (KB)	1.0	1.1	1.2	1.3	1.4
T_{QoS1} (ms)	1.1	1.1	1.2	1.2	1.3
T_{QoS2} (ms)	1.3	1.4	1.5	1.6	1.7
T_{QoS3} (ms)	2.0	2.2	2.4	2.6	2.8
Task	Task (6)	Task (7)	Task (8)	Task (9)	Task (10)
K (MB)	1.5	1.7	1.9	2.1	2.3
T_{QoS1} (ms)	1.4	1.4	1.5	1.5	1.6
T_{QoS2} (ms)	1.8	1.8	1.9	1.9	2.0
T_{QoS3} (ms)	3.0	3.1	3.2	3.3	3.4

Table 7. Task specifications of gateway devices (GW).

Task	Task (1)	Task (2)	Task (3)	Task (4)	Task (5)
Z (KB)	2.0	2.2	2.4	2.6	2.8
T_{QoS1} (ms)	1.0	1.1	1.2	1.3	1.4
T_{QoS2} (ms)	1.4	1.5	1.6	1.7	1.8
T_{QoS3} (ms)	2.2	2.4	2.6	2.8	3.0
Task	Task (6)	Task (7)	Task (8)	Task (9)	Task (10)
K (MB)	3.0	3.1	3.2	3.3	3.4
T_{QoS1} (ms)	1.5	1.6	1.6	1.7	1.7
T_{QoS2} (ms)	1.9	2.0	2.1	2.2	2.3
T_{QoS3} (ms)	3.2	3.4	3.6	3.8	4.0

Table 8. Task specifications of high-level devices (HL).

Task	Task (1)	Task (2)	Task (3)	Task (4)	Task (5)
Z (KB)	3.2	3.4	3.6	3.8	4.0
T_{QoS1} (ms)	1.1	1.2	1.3	1.4	1.5
T_{QoS2} (ms)	1.7	1.8	1.9	2.0	2.1
T_{QoS3} (ms)	2.2	2.5	2.9	3.3	3.8

Two main scenarios are considered for the performance evaluation. In the first scenario, the effect of QoS latency, T_{QoS} , is checked. Three cases are considered for this case; in each case, a certain value of QoS latency is assumed. The first case, case 1, represents the minimum latency, which corresponds to the ultra-low latency applications. In the second case, case 2, the QoS latency is assumed to be higher than that of case 1. This facilitates local executions, even if the resources are limited. The last case, case 3, is set for the highest value of QoS latency. These cases are deployed for both IoT, and D2D enabled 5G cellular cells.

The second scenario is introduced to evaluate the effect of introducing MEC servers to the edge of the IoT networks and also to the edge of the 5G cellular system. Two main cases are considered in this scenario; the first case, case I, is introduced to evaluate the effect of introducing I-MEC servers. In this case, the I-MEC servers are removed, and the percentage of blocked tasks is measured without deploying I-MEC servers. Furthermore, the percentage of blocked tasks is remeasured with the proposed MEC-based structure, where the I-MEC and C-MEC servers are deployed. The percentage of blocked tasks are compared in both systems to evaluate the effectiveness of introducing I-MEC servers.

The second case, case II, is considered for evaluating the effectiveness of offloading computing tasks to GW and HL devices. Furthermore, the effect of adding C-MEC server for D2D-based 5G system. Four systems are considered in this case. The first, system A, is the proposed system structure, with GW, HL and C-MEC. The second system, system B, is the same as the proposed structure, without deploying GW devices. GW devices are considered LL devices. The third system, system C, removes HL devices and all other devices and structures remain the same. For this system, all GW and HL devices are

considered LL devices. In the last system, system D, C-MEC server is removed, and all devices are considered to be LL devices.

5.2. Simulation Results and Discussion

The results of the first scenario are introduced in Figures 8–13. Figure 8 presents the time required to handle tasks for IoT devices. Each device is assumed to have one task from tasks indicated in Table 5. These tasks are handled either locally, i.e., at IoT device, or offloaded to edge servers, I-MEC and C-MEC. The average time required to handle each task is presented in Figure 8 for the three considered cases. As the QoS latency increases, i.e., moving from case 1 to case 2 and then case 3, the probability of offloading increases. This is to save the IoT devices' energy and prolong the lifetime of devices. Using the proposed structure, with MEC and SDN, all tasks are handled without any blocking, even with the existence of QoS latency constraints.

Figure 8. Cont.

Figure 8. Average latency of handling IoT tasks for the three considered cases of the first scenario. (a) Average latency for case (1), (b) Average latency for case (2), and (C) Average latency for case (3).

Figure 9. Cont.

Figure 9. Average latency of handling LL devices' tasks for the three considered cases of the first scenario. (a) Average latency for case (1), (b) Average latency for case (2), and (C) Average latency for case (3).

Figure 10. Average latency of handling GW devices' tasks for the three considered cases of the first scenario.

Figure 11. Average latency of handling HL devices' tasks for the three considered cases of the first scenario.

Figure 12. Total number of handled tasks of heterogeneous deployed devices and servers for the three considered cases of the first scenario.

Figure 13. Total number of local executed tasks and offloaded tasks of the first scenario.

Figure 9 provides the average latency for handling tasks of LL devices. Each LL device is assumed to have a computing task with the workload indicated in Table 6. A part of these tasks is executed locally at LL devices, while the rest of these tasks are offloaded to nearby GW and HL devices. The offloaded tasks are due to either energy constraints or

lake of local computing resources. Figures 10 and 11 present the average latency of GW and HL devices' handling tasks for the three considered cases.

For better discussing the results of the first scenario, the total number of handled tasks at each type of device is calculated. Figure 12 indicates the total number of handled tasks, at IoT devices, LLs, GWs, HLs, I-MEC servers and C-MEC servers, for the three considered cases. In the third case, the QoS latency increase, which provides a time for offloading and thus, the probability of offloading increases. This reduces the total number of local executed tasks in case 2 and case 3 compared to case 1. For the I-MEC and C-MEC, the number of handled tasks increases for cases 2 and 3 than that in case 1. This is because the number of offloaded tasks is increased due to decreasing latency constraints. Furthermore, the total number of local executed tasks and offloaded tasks are calculated to better indicate the effectiveness of the proposed structure. Figure 13 illustrates the total number of the local executed tasks and offloaded tasks for the three cases of the first simulation scenario. The number of offloaded tasks increases as the QoS latency increases, i.e., moving from case 1 to case 2 and then to case 3. This unloads end-devices in heterogeneous considered networks and efficiently makes use of network resources.

Results for the first case, case I, of the second scenario are introduced in Tables 9 and 10. Table 9 presents the percentage of blocked tasks of the considered LoRaWAN networks with and without edge computing servers, i.e., I-MEC and C-MEC. This is for the three considered values of QoS latency. Results indicate that the percentage of blocked tasks falls from 30% for LoRaWAN without edge computing servers to zero with the deployment of edge computing servers and offloading algorithm. Furthermore, the introduction of edge servers not only improves the percentage of blocked tasks but also improves the overall energy efficiency of LoRaWAN networks through the deployment of the developed offloading algorithm implemented with the developed edge structure. This is reasonable since IoT devices make use of edge computing resources. Table 10 presents the percentage of the average remaining energy, compared to start energy, for LoRaWAN networks with and without edge system with the offloading algorithm; this is after handling the considered tasks in the first scenario for the three considered values of QoS latency. This implies an average improvement of energy efficiency of 24% after deploying edge computing servers with the developed offloading algorithm.

Table 9. Percentage of blocked tasks of IoT networks in case (II) of the second scenario.

Case	Case (1)	Case (2)	Case (3)
IoT network (without edge)	38%	28%	24%
IoT network (with edge)	0	0	0

Table 10. Percentage of average remaining energy of IoT networks, after task execution, in case (II) of the second scenario.

Case	Case (1)	Case (2)	Case (3)
IoT network (without edge)	72%	67%	61%
IoT network (with edge)	84%	93%	95%

Figure 14 presents results for the second case, case II, of the second scenario. The number of blocked tasks for each considered system is recorded, as illustrated in Figure 14. Table 11 provides the percentage of blocked tasks for each system in each case. As results indicate, the percentage of blocked tasks of the proposed system, system A, is zero, and it increases as the edge devices are cut. System D achieves the worst results in terms of task blocking since the C-MEC server is removed and edge devices of D2D networks are considered as LL devices that send to the base station. For each system, the percentage of blocked tasks decreases as the QoS latency increases since this gives more time for task handling either by offloading or by limited resources.

Figure 14. Total number of blocked tasks for the four considered systems in case (II) of the second simulation scenario.

Table 11. Percentage of blocked tasks for the four considered systems in case (II) of the second scenario.

Case	Case (1)	Case (2)	Case (3)
System (B)	17.78%	15.56%	11.11%
System (C)	37.78%	26.67%	13.33%
System (D)	46.67%	42.22%	31.11%

In summary, results indicate that the introduction of MEC, D2D and SDN to IoT networks facilitates integrating IoT networks with 5G systems. Furthermore, the proposed framework can be used for dense deployment scenarios with higher network scalability. This fit great and efficient for many applications, especially the announced dense deployment applications for next-generation networks. These applications include smart city applications, smart homes and industrial 4.0. All these applications have a challenge of dense deployment and a massive amount of traffic. Integrating IoT with a 5G system provides a way for realizing such applications with the required QoS and scalability requirements.

6. Conclusions and Future Work

Heterogeneous IoT networks can be integrated with 5G systems via MEC servers, using the proposed system structure. Connecting IoT gateways with MEC servers, I-MEC servers provide paths for 5G edge servers, C-MEC servers. Furthermore, the introduction of edge computing servers to the access networks of heterogeneous IoT networks provides computing resources to IoT devices, which unloads the core networks. This achieves many benefits in terms of network scalability, availability and reliability. The article also introduces D2D communications over cellular cells to unloads the core networks and increases the system availability and scalability. Computing tasks can be managed, over the proposed IoT/5G networks, using the proposed energy-aware and latency-aware offloading scheme. With the proposed IoT/5G structure, the proposed offloading scheme, achieves 30% improvement in blocked tasks and 20% improvement in energy efficiency. Thus, the proposed IoT/5G structure, with the proposed offloading scheme, achieves IoT's main requirements over 5G systems announced by the 3GPP and ITU.

The proposed IoT/5G framework is limited by the capabilities of the I-MEC and C-MEC servers. Higher server capabilities, e.g., energy, processing and storage, achieve higher performance in terms of scalability, availability and energy efficiency; however, this comes on the overall system cost. Thus, our future goals for this work include developing an optimization algorithm for optimizing and allocating resources dynamically and presents the edge intelligence by developing machine-learning algorithms at the edge servers to improve the overall network performance and provide novel services. Moreover, further measures will be examined for key performance indicators (KPIs), such as overall system reliability.

Author Contributions: Conceptualization, M.H., A.A.A. and N.F.S.; methodology, A.D.A., A.K. and A.A.A.; software, N.F.S. and A.A.A.; validation, A.A.A., M.H. and A.D.A.; formal analysis, A.K., M.H. and A.A.A.; investigation, A.D.A. and A.A.A.; resources, A.A.A. and A.K.; data curation, M.H.; writing—original draft preparation, A.A.A.; writing—review and editing, A.A.A., N.F.S. and M.H. and N.F.S.; visualization, A.A.A. and A.D.A.; supervision, A.K. and N.F.S.; project administration, M.H.; funding acquisition, M.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Deanship of Scientific Research at Princess Nourah bint Abdulrahman University through the Research Funding Program (grant no. FRP-1440-24).

Conflicts of Interest: The authors declare no conflict of interest.

References

- Li, S.; Da Xu, L.; Zhao, S. 5G Internet of Things: A survey. *J. Ind. Inf. Integr.* **2018**, *10*, 1–9. [\[CrossRef\]](#)
- Byabazaire, J.; O'Hare, G.; Delaney, D. Data Quality and Trust: Review of Challenges and Opportunities for Data Sharing in IoT. *Electronics* **2020**, *9*, 2083. [\[CrossRef\]](#)
- Iraji, S.; Mogensen, P.; Ratasuk, R. Recent advances in M2M communications and Internet of Things (IoT). *Int. J. Wirel. Inf. Netw.* **2017**, *24*, 240–242. [\[CrossRef\]](#)
- Escribano, C.P.; Theologou, N.; Likar, M.; Tryferidis, A.; Tzovaras, D. Business Models and Use Cases for the IoT. In *IoT Platforms, Use Cases, Privacy, and Business Models*; Springer: Cham, Switzerland, 2021; pp. 51–80.
- Glaroudis, D.; Iossifides, A.; Chatzimisios, P. Survey, comparison and research challenges of IoT application protocols for smart farming. *Comput. Netw.* **2020**, *168*, 107037. [\[CrossRef\]](#)
- Henry, S.; Alshaily, A.; Sousa, E.S. 5G is real: Evaluating the compliance of the 3GPP 5G new radio system with the ITU IMT-2020 requirements. *IEEE Access* **2020**, *8*, 42828–42840. [\[CrossRef\]](#)
- Shafique, K.; Khawaja, B.A.; Sabir, F.; Qazi, S.; Mustaqim, M. Internet of things (IoT) for next-generation smart systems: A review of current challenges, future trends and prospects for emerging 5G-IoT scenarios. *IEEE Access* **2020**, *8*, 23022–23040. [\[CrossRef\]](#)
- Ateya, A.; Al-Bahri, M.; Muthanna, A.; Koucheryavy, A. End-to-end system structure for latency sensitive applications of 5G. *Электросвязь* **2018**, *6*, 56–61.
- Pham, Q.V.; Fang, F.; Ha, V.N.; Piran, M.J.; Le, M.; Le, L.B.; Hwang, W.J.; Ding, Z. A survey of multi-access edge computing in 5G and beyond: Fundamentals, technology integration, and state-of-the-art. *IEEE Access* **2020**, *8*, 116974–117017. [\[CrossRef\]](#)
- Contreras, L.M.; Bernardos, C.J. Overview of Architectural Alternatives for the Integration of ETSI MEC Environments from Different Administrative Domains. *Electronics* **2020**, *9*, 1392. [\[CrossRef\]](#)
- Choumas, K.; Giatsios, D.; Flegkas, P.; Korakis, T. SDN Controller Placement and Switch Assignment for Low Power IoT. *Electronics* **2020**, *9*, 325. [\[CrossRef\]](#)
- Ahmad, S.; Mir, A.H. Scalability, Consistency, Reliability and Security in SDN Controllers: A Survey of Diverse SDN Controllers. *J. Netw. Syst. Manag.* **2021**, *29*, 1–59. [\[CrossRef\]](#)
- Ateya, A.A.; Muthanna, A.; Gudkova, I.; Abuarqoub, A.; Vybornova, A.; Koucheryavy, A. Development of intelligent core network for tactile internet and future smart systems. *J. Sens. Actuator Netw.* **2018**, *7*, 1. [\[CrossRef\]](#)
- Alquhali, A.H.; Roslee, M.; Alias, M.Y.; Mohamed, K.S. D2D communication for spectral efficiency improvement and interference reduction: A survey. *Bull. Electr. Eng. Inform.* **2020**, *9*, 1085–1094. [\[CrossRef\]](#)
- Basak, S.; Acharya, T. On energy efficient secure routing in multi-hop underlay D2D communications for IoT applications. *Ad Hoc Netw.* **2020**, *108*, 102275. [\[CrossRef\]](#)
- Mehta, P.; Gupta, R.; Tanwar, S. Blockchain envisioned UAV networks: Challenges, solutions, and comparisons. *Comput. Commun.* **2020**, *151*, 518–538. [\[CrossRef\]](#)
- Vilalta, R.; López, V.; Giorgetti, A.; Peng, S.; Orsini, V.; Velasco, L.; Serral-Gracia, R.; Morris, D.; De Fina, S.; Cugini, F.; et al. TelcoFog: A unified flexible fog and cloud computing architecture for 5G networks. *IEEE Commun. Mag.* **2017**, *55*, 36–43. [\[CrossRef\]](#)
- Ateya, A.A.; Muthanna, A.; Koucheryavy, A. 5G framework based on multi-level edge computing with D2D enabled communication. In Proceedings of the 2018 20th International Conference on Advanced Communication Technology (ICACT), Chuncheon, Korea, 11–14 February 2018; pp. 507–512.
- Apostolopoulos, P.A.; Tsiropoulou, E.E.; Papavassiliou, S. Cognitive data offloading in mobile edge computing for internet of things. *IEEE Access* **2020**, *8*, 55736–55749. [\[CrossRef\]](#)
- Shen, Y.; Fang, W.; Ye, F.; Kadoch, M.E.V. Charging Behavior Analysis Using Hybrid Intelligence for 5G Smart Grid. *Electronics* **2020**, *9*, 80. [\[CrossRef\]](#)
- Wang, Q.; Chen, S. Latency-minimum offloading decision and resource allocation for fog-enabled Internet of Things networks. *Trans. Emerg. Telecommun. Technol.* **2020**, *31*, 3880. [\[CrossRef\]](#)
- Shahryari, O.K.; Pedram, H.; Khajehvand, V.; TakhtFooladi, M.D. Energy-Efficient and delay-guaranteed computation offloading for fog-based IoT networks. *Comput. Netw.* **2020**, *182*, 107511. [\[CrossRef\]](#)

23. Ahanger, T.A.; Tariq, U.; Ibrahim, A.; Ullah, I.; Bouteraa, Y. IoT-Inspired Framework of Intruder Detection for Smart Home Security Systems. *Electronics* **2020**, *9*, 1361. [\[CrossRef\]](#)
24. Akbar, A.; Ibrar, M.; Jan, M.A.; Bashir, A.K.; Wang, L. SDN-enabled Adaptive and Reliable Communication in IoT-Fog Environment Using Machine Learning and Multi-Objective Optimization. *IEEE Internet Things J.* **2020**, *8*, 3057–3065. [\[CrossRef\]](#)
25. Ahmed, M.; Mumtaz, R.; Zaidi, S.M.H.; Hafeez, M.; Zaidi, S.A.R.; Ahmad, M. Distributed Fog Computing for Internet of Things (IoT) Based Ambient Data Processing and Analysis. *Electronics* **2020**, *9*, 1756. [\[CrossRef\]](#)
26. Zahed, M.I.A.; Ahmad, I.; Habibi, D.; Phung, Q.V. Green and secure computation offloading for cache-enabled IoT networks. *IEEE Access* **2020**, *8*, 63840–63855. [\[CrossRef\]](#)
27. Lee, J.; Lee, K.; Yoo, A.; Moon, C. Design and Implementation of Edge-Fog-Cloud System through HD Map Generation from LiDAR Data of Autonomous Vehicles. *Electronics* **2020**, *9*, 2084. [\[CrossRef\]](#)
28. Saba, U.K.; ul Islam, S.; Ijaz, H.; Rodrigues, J.J.; Gani, A.; Munir, K. Planning Fog networks for time-critical IoT requests. *Comput. Commun.* **2021**, *172*, 75–83. [\[CrossRef\]](#)
29. Lakhan, A.; Qurat-Ul-Ain, M.; Mohamed, E.; Muhammad, S.M.; Mazin, A.M. Deep neural network-based application partitioning and scheduling for hospitals and medical enterprises using IoT assisted mobile fog cloud. *Enterp. Inf. Syst.* **2021**, 1–23. [\[CrossRef\]](#)
30. Abbasi, M.; Mohammadi-Pasand, E.; Khosravi, M.R. Intelligent workload allocation in IoT-Fog-cloud architecture towards mobile edge computing. *Comput. Commun.* **2021**, *169*, 71–80. [\[CrossRef\]](#)
31. Alamer, A. Security and Privacy-Awareness in a Software-Defined Fog Computing Network for the Internet of Things. *Opt. Switch. Netw.* **2021**, *41*, 100616. [\[CrossRef\]](#)
32. Losada, M.; Cortés, A.; Irizar, A.; Cejudo, J.; Pérez, A. A Flexible Fog Computing Design for Low-Power Consumption and Low Latency Applications. *Electronics* **2021**, *10*, 57. [\[CrossRef\]](#)
33. Bolettieri, S.; Bruno, R.; Mingozzi, E. Application-aware resource allocation and data management for MEC-assisted IoT service providers. *J. Netw. Comput. Appl.* **2021**, *181*, 103020. [\[CrossRef\]](#)
34. Santos, J.; Wauters, T.; Volckaert, B.; De Turck, F. Towards end-to-end resource provisioning in Fog Computing over Low Power Wide Area Networks. *J. Netw. Comput. Appl.* **2021**, *175*, 102915. [\[CrossRef\]](#)
35. Liu, C.; Feng, W.; Tao, X.; Ge, N. MEC-Empowered Non-Terrestrial Network for 6G Wide-Area Time-Sensitive Internet of Things. *arXiv* **2021**, arXiv:2103.11907.
36. Ateya, A.A.; Muthanna, A.; Vybornova, A.; Darya, P.; Koucheryavy, A. Energy-aware offloading algorithm for multi-level cloud based 5G system. In *Internet of Things, Smart Spaces, and Next Generation Networks and Systems*; Springer: Cham, Switzerland, 2018; pp. 355–370.
37. Al Breiki, M.S.; Zhou, S.; Luo, Y.R. Development of OpenFlow Native Capabilities to Optimize QoS. In Proceedings of the 2020 Seventh International Conference on Software Defined Systems (SDS), Paris, France, 20–23 April 2020; pp. 67–74.
38. Ateya, A.A.; Muthanna, A.; Vybornova, A.; Algarni, A.D.; Abuarqoub, A.; Koucheryavy, Y.; Koucheryavy, A. Chaotic salp swarm algorithm for SDN multi-controller networks. *Eng. Sci. Technol. Int. J.* **2019**, *22*, 1001–1012. [\[CrossRef\]](#)
39. Ateya, A.A.; Muthanna, A.; Kirichek, R.; Hammoudeh, M.; Koucheryavy, A. Energy-and latency-aware hybrid offloading algorithm for UAVs. *IEEE Access* **2019**, *7*, 37587–37600. [\[CrossRef\]](#)
40. Marais, J.M.; Abu-Mahfouz, A.M.; Hancke, G.P. A Survey on the Viability of Confirmed Traffic in a LoRaWAN. *IEEE Access* **2020**, *8*, 9296–9311. [\[CrossRef\]](#)
41. Ballerini, M.; Polonelli, T.; Brunelli, D.; Magno, M.; Benini, L. NB-IoT Versus LoRaWAN: An Experimental Evaluation for Industrial Applications. *IEEE Trans. Ind. Inform.* **2020**, *16*, 7802–7811. [\[CrossRef\]](#)
42. Kulkarni, P.; Pradeep, B.; Raza, U.; Alsaedi, S.; Almadhaani, R. LoRaWAN in Licensed Access Spectrum? A Techno-Economic Perspective. *IEEE Internet Things Mag.* **2020**, *3*, 70–75. [\[CrossRef\]](#)
43. Khan, M.A.; Cherif, W.; Filali, F.; Hamila, R. Wi-Fi Direct Research-Current Status and Future Perspectives. *J. Netw. Comput. Appl.* **2017**, *93*, 245–258. [\[CrossRef\]](#)
44. Fernandes, E.L.; Rojas, E.; Alvarez-Horcajo, J.; Kis, Z.L.; Sanvito, D.; Bonelli, N.; Cascone, C.; Rothenberg, C.E. The road to BOFUSS: The basic OpenFlow userspace software switch. *J. Netw. Comput. Appl.* **2020**, *165*, 102685. [\[CrossRef\]](#)
45. Semong, T.; Maupong, T.; Anokye, S.; Kehulakae, K.; Dimakatso, S.; Boipelo, G.; Sarefo, S. Intelligent Load Balancing Techniques in Software Defined Networks: A Survey. *Electronics* **2020**, *9*, 1091. [\[CrossRef\]](#)