Computer Networks 209 (2022) 108900

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Joint computing, communication and cost-aware task offloading in D2D-enabled Het-MEC

Nadine Abbas [a,*], Sanaa Sharafeddine [a], Azzam Mourad [a,b], Chadi Abou-Rjeily [c], Wissam Fawaz [c]

[a] Department of Computer Science and Mathematics, Lebanese American University, Beirut, Lebanon
[b] Division of Science, New York University, Abu Dhabi, United Arab Emirates
[c] Department of Electrical and Computer Engineering, Lebanese American University, Byblos, Lebanon

ARTICLE INFO

Keywords:
Mobile edge computing
Cloud computing
Partial offloading
Computation resource allocation
Radio resource allocation
D2D communication
Multi-RAT

ABSTRACT

Due to the exploding traffic demands and the diversity of novel applications requiring extensive computation and radio resources, research has been active to devise mechanisms for responding to these challenges. Mobile edge computing (MEC) and device-to-device (D2D) computation task offloading are expected to play a major role in serving devices with limited capabilities, and thus enhance system performance. In this work, we propose a joint computing, communication and cost-aware task offloading optimization problem aiming at maximizing the number of completed tasks, while minimizing energy consumption and monetary cost in D2D-enabled heterogeneous MEC networks. Our proposed scheme allows partial offloading where a requester mobile terminal offloads different parts of its data task simultaneously to multiple peer mobile terminals (MTs), edge servers and cloud. We formulate and solve the optimal allocation strategy then decompose the problem into two sub-problems in an attempt to reduce its complexity. Furthermore, we propose a low-complexity algorithm that generates high performance results and can be applied for large-scale networks. Compared to conventional and state-of-the-art system models, results show the effectiveness of the proposed schemes and provide useful insights into the tradeoffs between the number of completed tasks, energy consumption and monetary cost.

1. Introduction

As technology is advancing, applications are becoming so diverse and demanding as many global trends have arisen. Applications, such as video surveillance, feature and facial recognition, healthcare monitoring, automatic driving, in addition to virtual and augmented reality, are expected to increase 12-fold between 2017 and 2022 [1]. These novel applications require extensive computation and radio resources, which may exceed the computing capabilities of the devices and thus creating the need for efficient computation offloading and faster means of communication. In support of this direction, the future wireless networks are expected to provide the ability to accommodate massive connections and high loads with ultra-fast speeds [2].

Mobile edge computing is considered as a key design of the future wireless networks providing cloud computing capabilities such as heavy computation tasks offloading at mobile edge network devices or servers, which are located in close proximity to battery-powered user equipment for ultra-reliable and low-latency services [3,4]. The area of mobile edge computing has been gaining a lot of attention in the literature recently [5,6]. D2D communication was integrated with MEC to further enhance system performance in terms of capacity, energy consumption and delay [7,8]. The authors in [9,10] considered binary offloading where a task can be executed locally or remotely. The authors in [11–14] adopted partial offloading where a task can be partitioned into different sub-tasks to be executed simultaneously locally and remotely. Partial offloading proved to be efficient and suitable for low-latency and data partitioned oriented applications, such as virus scan, file/figure compression, recognition, and vision applications [8,11]. In such applications, the input data is bit-wise independent and can be arbitrarily partitioned for parallel processing.

In general, previous works considered D2D offloading while limiting the connection to one device and the number of chunks to maximum three [12,13]. Moreover, previous studies focused mainly on one or two objective functions including reducing latency in [7,11,15,16], energy consumption in [8,17], increasing the number of completed tasks in [12,13] with delay and energy constraints, minimizing monetary cost while reducing energy consumption [18] or latency in [19,20], and providing a trade-off between energy and latency in [14]. However, these objectives are interdependent, hence, a trade-off between

[*] Corresponding author.
E-mail addresses: nadine.abbas@lau.edu.lb (N. Abbas), sanaa.sharafeddin@lau.edu.lb (S. Sharafeddine), azzam.mourad@lau.edu.lb (A. Mourad), chadi.abourjeily@lau.edu.lb (C. Abou-Rjeily), wissam.fawaz@lau.edu.lb (W. Fawaz).

https://doi.org/10.1016/j.comnet.2022.108900
Received 29 September 2021; Received in revised form 11 January 2022; Accepted 9 March 2022
Available online 23 March 2022

multiple objectives should be considered to enhance task offloading performance. In summary, existing research work on resource management in D2D-enabled heterogeneous MEC (Het-MEC) networks is still limited in terms of scalability and performance. Considering partial offloading to multiple MTs, edge servers and cloud, simultaneously, becomes a must to provide substantial gains for tasks offloading. Lastly, fast and high performance solutions should be provided in real-time dense D2D-enabled Het-MEC.

Complementing the existing literature, we address joint computing, communication and cost-aware task offloading multi-objective optimization problem. We present a comprehensive multi-layer D2D-enabled heterogeneous MEC network where a network operator takes advantage of all the available computation and communication resources to serve the maximum number of tasks within their deadline while minimizing its operational cost. The main contributions of this work can be summarized as follows:

1. The adoption of multi-layer D2D-enabled Het-MEC networks where multiple edge servers, cloud, and peer mobile terminals cooperate to enhance system performance. We allow partial offloading where the computation task data is bit-wise independent and can be divided into multiple subtasks to be simultaneously executed locally and remotely over multiple nodes.
2. The adoption of heterogeneous networks (Het-Nets) and D2D communication where the requester mobile terminal (MTR) communicates simultaneously over multiple wireless technologies. MTR can then offload data to multiple peer MTs using short range wireless technologies such as Bluetooth, to multiple edge servers through access points (APs) using WiFi and to the cloud through the base stations (BSs) over long range technologies such as cellular technologies.
3. The development of a framework achieving optimized joint offloading decision, radio and computation resource allocation aiming at completing the maximum number of computation tasks while simultaneously reducing the mobile terminals energy consumption and the operational monetary cost of the network operator including incentives paid for the peer mobile terminals to contribute their resources, as well as, the cost of the edge and cloud services.
4. Problem decomposition into two optimization sub-problems and proposing a hierarchical offloading approach to reduce the complexity of solving the formulated optimization problem. We first solve the optimal allocation while maximizing the number of completed tasks which determines the MTRs served. We then provide optimized solutions for serving the determined MTRs with minimum energy consumption and monetary cost. Solving these sub-problems consecutively reduces the number of MTRs to the number of served MTRs, which has large impact on the number of decision variables and system constraints. However, it may not have high impact on reducing the execution time in scenarios where all MTRs can be served.
5. The development of low-complexity algorithm allowing for real-time iterative task offloading providing fast sub-optimal solutions in large-scale Het-MEC networks. The proposed iterative approach is scalable and can operate under real-time conditions while considering dynamic system parameters.

This paper is organized as follows. Existing studies are surveyed in Section 2. The system model is presented in Section 3. The optimization allocation problem is detailed in Section 4. The proposed hierarchical and iterative approaches are presented in Sections 5 and 6, respectively. Performance results are discussed in Section 7. Finally, conclusions are drawn in Section 8.

2. Related work

Recently, computation offloading to edge servers and cloud has become a hotspot in research [5,6]. Many work addressed binary offloading [15,21–24] and partial offloading [16,17,25,26] in MEC networks. The authors in [15] aimed at jointly optimizing the task offloading, the users' and servers' transmit power, communication and computation resource allocation for multi-user non-orthogonal multiple access (NOMA)-based multi-access MEC systems while minimizing the overall users' tasks delay. In [16], the authors addressed cellular-assisted MEC with NOMA aiming at jointly optimizing the edge users' computation offloading, the offloading duration, and the edge server computation resource allocation while minimizing the overall tasks completion latency. The authors in [17] jointly optimize computation offloading, data compression and resource allocation in a multi-user MEC system aiming at minimizing the energy consumption subject to latency and computation capacity constraints.

As the delay-sensitive applications are becoming more diverse, D2D communication was integrated with MEC to further enhance system performance. In this section we survey the open literature considering D2D cooperation for computation task offloading. The work in [7,8] considered D2D cooperation to create a cloud environment and offload tasks to peer MTs, without the use of edge servers and cloud. The authors in [8] addressed partial D2D computation offloading to nearby cooperating devices while minimizing energy consumption and execution time.

In the following studies, D2D offloading was adopted with MEC and mobile cloud computing (MCC) offloading to achieve performance gains in terms of capacity, energy consumption, latency and monetary cost. The authors in [9,10] considered binary offloading in D2D-enabled MEC systems. The authors in [11–14] adopted partial offloading in D2D-enabled Het-MEC systems. The authors in [11] addressed resource allocation and interference management aiming at minimizing the task execution latency subject to energy and delay constraints. They adopted partial offloading, where a task can be divided into two parts. One part is processed locally, and the rest is offloaded for remote execution on the device or a smart base station. The authors in [12,13] adopted partial offloading where a user's task is partitioned intro three parts to be executed locally, at one edge server, and at one peer MT, simultaneously. They addressed task offloading, computation resource and power allocation, aiming at maximizing the number of supported devices while meeting delay and power constraints. In [14], the authors addressed the joint selection of computation modes, computation resources and bandwidth allocation while minimizing latency and energy consumption. The scheme in [14] allowed the tasks to be executed simultaneously locally, offloaded to one MEC, or to one peer device. Accordingly, they adopted 3 computation modes: local execution, complete or partial D2D offloading with local execution, and complete or partial MEC offloading with local execution. Price-aware offloading strategies were proposed while considering binary offloading in [19] and partial offloading in [18,20] in Het-MEC networks without D2D cooperation.

The system models considered in the existing D2D-enabled Het-MEC are still limited. They mainly used binary offloading while some used partial offloading with very limited number of cooperating nodes. Moreover, previous studies focused mainly on one or two objective functions ignoring the need to provide a trade-off between these objectives. In our work, we propose a comprehensive D2D-enabled Het-MEC system model that addresses the joint problem of (1) partial offloading of computation tasks to multiple nodes (MTs, MECs, and cloud), (2) computation resource allocation reserved by a MT, MEC or cloud to execute the offloaded portion of the task, and (3) radio resource allocation where we allocate channels over Bluetooth, WiFi and cellular networks for D2D, MEC and MCC offloading, respectively. We formulate the problem as a multi-objective optimization problem and present low-complexity sub-optimal approaches achieving close-to-optimal results in scenarios with limited number of devices, which are then applied for large-scale Het-MEC networks.

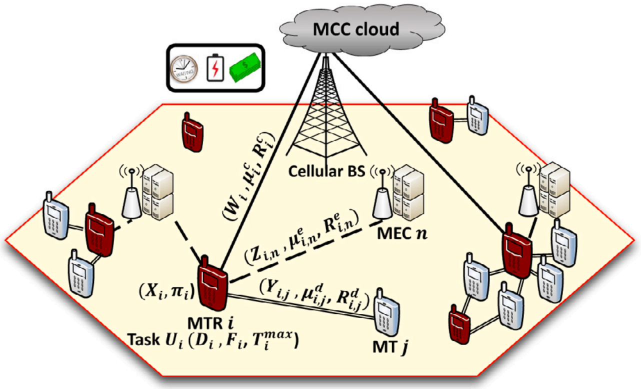

MCC cloud
WAITING
$
Cellular BS
(W_i, μ_i^c, R_i^c)
$(Z_{i,n}, \mu_{i,n}^e, R_{i,n}^e)$
MEC n
(X_i, π_i)
$(Y_{i,j}, \mu_{i,j}^d, R_{i,j}^d)$
MTR i
Task U_i (D_i, F_i, T_i^{max})
MT j

Fig. 1. Heterogeneous MEC network composed of mobile terminals, edge servers and cloud.

3. System model

In our work, we address D2D-enabled Het-MEC network where a network operator offers computation services to its subscriber MTRs having computation tasks to be executed. As shown in the sample scenario depicted in Fig. 1, a Het-MEC network is composed of multiple nodes having different computation capabilities, ranging from MTs to highly powerful cloud-computing servers. We address modern devices equipped with multiple wireless interfaces and take advantage of the heterogeneous networks for parallel computation offloading. A requester MTR can then offload to multiple MTs using short range (SR) wireless technologies (such as LTE-Direct, WiFi-Direct, or Bluetooth), multiple edge servers and cloud through the APs and the BSs over long range (LR) technologies (such as WiFi, LTE, or 5G). We assume our network operator is subscribed with cloud services to support its own computational services, deploys edge servers accessible through WiFi APs, and pays the peer MTs incentives to share their resources over SR connectivity. The mobile terminals are subscribed for the computation service offered by the network operator and provide the needed information over control channels. MTs may choose to contribute their computational resources for a monetary incentive. We assume a cellular BS is equipped with a controller server that handles the offloading decisions, computation and communication resource allocation. The network controller can then make computation offloading decisions and broadcast them to the mobile terminals over a control channel. Our problem aims at maximizing the benefits of the network operator to serve the maximum number of subscribers while minimizing its operational expenses (OPEX). The operational expenses of the network operator addressed in this paper include the usage fees of the cloud, edge and incentives. To do so, the operator encourages more cost effective task offloading through utilizing MEC and D2D offloading whenever feasible. In addition, the operational cost includes the incentives paid for the peer MTs to contribute their resources. Aiming at improving the quality of experience of its subscribers, it is of the network operator's interest to minimize the energy consumption of using this service whether being a requester or a peer mobile terminal.

3.1. Basic parameters

Our network is composed of N MEC servers/APs, a single MCC/BS, in addition to K mobile terminals out of which A are requester terminals and J are cooperating terminals. A MTR i has a computation task U_i described by (D_i, F_i, T_i^{max}), where D_i (in bits) represents the amount of computation data required to accomplish task U_i, F_i (in cycles/bit) represents the number of CPU cycles required for computing 1-bit of data, and T_i^{max} denotes the maximum delay tolerance for task U_i. The computation capabilities at MTR i may be limited and task U_i may not be executed locally within T_i^{max}, therefore, we adopt partial offloading, where each device can split its task into multiple parts for local and remote execution. The computation data task is then divided and assigned to be executed by the cooperating nodes including peer MTs, MEC servers and cloud having different computation capabilities. Let F_i^l (in Hz) denotes the computation resources of a requester MTR i, F_j^d, F_n^e and F^c the computation resources of a peer MT j, edge server MEC n and MCC, respectively. Accordingly, every cooperating node will allocate part of its computation resources to execute a chunk of the computation task U_i. Moreover, our system model is extended to accommodate for real-time scenarios with MTRs generating multiple tasks with a level of activity λ_{A}. Accordingly, at every time slot of duration T_s, the offloading decisions can be performed while considering the dynamic system variations. The main system parameters are summarized in Table 1.

3.2. Decision variables

As illustrated in Fig. 1, we decide on the amount of data to be executed locally and offloaded to every cooperating node, as well as the radio and computation resource allocated for every MTR. For offloading decisions, the main decision variables (see Table 1) are presented as follows: X_i the size of computation data executed locally at MTR i, $Y_{i,j}$, $Z_{i,n}$ and W_i the size of data offloaded from MTR i to MT j, MEC n and MCC, respectively. Accordingly, the total size of data processed can be represented by G_i as follows:

$$G_i = X_i + \sum_{j=1}^{J} Y_{i,j} + \sum_{n=1}^{N} Z_{i,n} + W_i \tag{1}$$

For computation resource allocation, the solution of the problem decides on $\mu_{i,j}^d$, $\mu_{i,n}^e$ and μ_i^c representing the fraction of computation capacity allocated by MT j, MEC n and MCC to execute task U_i generated by MTR i, respectively.

For communication resource allocation, we allocate radio channels for D2D, MEC and MCC offloading. We denote by $L_{i,j}^d$ a binary variable indicating whether a connection is established and a channel is allocated for computation task offloading from MTR i to peer MT j. A channel $L_{i,j}^d$ is only set to 1 when MTR i is offloading data to peer MT j and is located within its coverage range. We denote by $v_{i,j}^d$ an input binary variables indicating whether MTR i is within MT j coverage. Similarly, we denote by $L_{i,n}^e$ and L_i^c binary variables indicating whether a channel is allocated for computation offloading from MTR i to MEC n, and MCC, respectively. We denote by $v_{i,n}^e$ an input binary variables indicating whether MTR i is within MEC n coverage range. Moreover, we assume multiple subchannels allocation for MCC computation offloading over the cellular network. We denote by η_i^c the number of subchannels allocated to MTR i by the MCC.

We denote by π_i a binary variable indicating whether task U_i requested by MTR i is fully executed within the time limit T_i^{max}. Accordingly, π_i is set to one when the data assigned G_i is equal to the data size D_i of task U_i. Otherwise, if the task cannot be fully executed, we do not allow data nor resources to be assigned for MTR i; that is, when π_i is set to zero, X_i, $Y_{i,j}$, $Z_{i,n}$, W_i, the computation resources $\mu_{i,j}^d$, $\mu_{i,n}^e$ and μ_i^c, as well as the channel and radio resources $L_{i,j}^d$, $L_{i,n}^e$ and L_i^c are set to zero. Accordingly, the actual transmission rate, computation and channel allocation are nonzero only if the task can be completed and the data is offloaded to the corresponding node within its coverage range.

Accordingly, the decision variables are: $\mathbf{X}$, $\mathbf{W}$, π, μ^c, L^c, and η^c vectors of size A, $\mathbf{Y}$, μ^d, and L^d matrices of size $A \times J$, and $\mathbf{Z}$, μ^e, and L^e matrices of size $A \times N$.

Table 1
Main parameters and decision variables.

Parameters	
K	number of mobile terminals, where $K = A + J$
A	number of mobile terminals requesting tasks, and a requester is referred to as MTR i where $i = 1, \dots, A$
J	number of peer mobile terminals, and a peer terminal is referred to as MT j where $j = 1, \dots, J$
N	number of MEC servers/APs, where a MEC server is referred to as MEC n where $n = 1, \dots, N$
D_i	size of computation resource (in bits) of task U_i of MTR i
F_i	number of CPU cycles required for computing 1-bit data of U_i
T_i^{max}	maximum delay tolerance (in seconds) for a task U_i
ζ^d	maximum number of D2D connections allowed per MTR
ζ^e	maximum number of MEC connections allowed per MTR
ζ^c	maximum number of cellular subchannels allocated per MTR
Ω^d	maximum number of MTRs served by a peer MT
Ω^e	maximum number of MTRs served by a MEC
Ω^c	maximum number of MTRs served by the MCC
Decision variables	
X_i	integer variable indicating the amount of computation tasks executed locally at the requester MTR i
$Y_{i,j}$	integer variable indicating the amount of computation tasks offloaded from MTR i to MT j
$Z_{i,n}$	integer variable indicating the amount of computation tasks offloaded from MTR i to MEC n
W_i	integer variable indicating the amount of computation tasks offloaded from MTR i to the MCC
$\mu_{i,j}^d$	real variable varying between 0 and 1, indicating the fraction of computation resources allocated by MT j to execute MTR i task
$\mu_{i,n}^e$	real variable varying between 0 and 1, indicating the fraction of computation resources allocated by MEC n to execute MTR i task
μ_i^c	real variable varying between 0 and 1, indicating the fraction of computation resources allocated by MCC to execute MTR i task
π_i	binary variable indicating if MTR i is served; i.e., task U_i is fully executed within T_i^{max}
$L_{i,j}^d$	binary variable indicating whether a channel is allocated for computation task offloading from MTR i to MT j
$L_{i,n}^e$	binary variable indicating whether a channel is allocated for computation task offloading from MTR i to MEC n
L_i^c	binary variable indicating whether a channel is allocated for computation task offloading from MTR i to MCC
η_i^c	integer variable indicating the number of cellular subchannels allocated to MTR i by the MCC BS

3.3. Communication models

In our work, we consider a heterogeneous network deployment where the MTR can offload computation tasks to the cloud through cellular network, to the edge servers through WiFi APs and to other MTs through Bluetooth. As in any wireless system, dedicated channels are used to exchange control information. The data size of the control messages is typically minimal leading to a negligible overhead. Accordingly, we target transmission data rate and energy consumption in our problem formulation which vary based on the channel conditions and the different wireless technologies characteristics such as transmit power, bandwidth and channel allocation presented below and in Section 7.2. Following from Shannon's channel capacity formulation, the achievable data rate can be estimated as follows:

$$R = B \cdot \log_2 \left(1 + \frac{p \cdot h}{\sigma^2} \right) \tag{2}$$

where B, p, and σ^2 represent the channel bandwidth, the transmit power and the noise power, respectively. We assume the wireless channel to be dominated by line-of-sight component, where the channel gain h can be expressed as follows:

$$h = \kappa \cdot \left(\frac{d_0}{d_{i,x}} \right)^{\alpha} \tag{3}$$

where κ is a pathloss constant, α is the pathloss exponent, d_0 is a reference distance (typically 1 or 10 meters in indoor or short range outdoor scenarios) while $d_{i,x}$ is the distance between MTR i and receiver node x, which can be a MT j, MEC n, or MCC [27].

3.3.1. Bluetooth for D2D offloading

For D2D cooperation, we assume that each D2D pair is assigned one orthogonal non-overlapping subchannel with bandwidth B^d over Bluetooth. For instance, Bluetooth Low Energy (BLE) 5.0 operates at 2.4 GHz and uses Frequency Hopping Spread Spectrum (FHSS) over 40 channels with 2 MHz channel spacing [28]. We assume offline network discovery where MTRs can discover other devices in their vicinity before the dynamic decisions are made. The MTR can then interconnect with up to seven active slave MTs and up to 255 inactive or parked peer slave MTs [29]. Accordingly, based on the dynamic computation offloading decisions, transmission physical channels will be allocated for computation offloading from the MTR to the active MTs. Following the Bluetooth Standards, we assume the availability of Ω^d orthogonal channels for active peer MTs and MTRs. We also limit the number of active D2D connections for a MTR i to ζ^d.

3.3.2. Wifi for MEC offloading

For MEC offloading, the MTR should be authenticated and associated with the WiFi AP before starting the computation offloading. The new generations of WiFi standards rely on request-to-send/clear-to-send procedure to avoid collisions in the case of multi-user scenarios [30]. Following WiFi Standards, we adopt the 5 GHz IEEE 802.11n which uses Orthogonal Division Multiplexing and provides three Unlicensed National Information Infrastructure (UNII) bands: UNII-1, UNII-2 and UNII-3 having 4, 15 and 4 number of orthogonal channels, respectively [31]. To limit the interference, we distribute the orthogonal channels to the N APs and assume the availability of Ω^e orthogonal channels per MEC. We also limit the number of connected MEC servers for a MTR i to ζ^e. Every MTR is assigned one channel at a time and thus no contention could occur in this case as the channel is reserved for this MTR. The MTRs can then directly access and fully use the channel allocated to them based on the computation offloading and resource allocation decisions.

3.3.3. Cellular network for cloud offloading

We assume that a MTR can offload its computation data task to MCC through cellular network. We adopt orthogonal frequency-division multiple access method for channel access. We further assume that the available bandwidth resource of the BS is divided into equal-band subchannels. Let B^c and Ω^c denote, respectively, the subchannel bandwidth and the total number of subchannels of the BS. Each MTR i is then allocated a number of orthogonal subchannels η_i^c. In our work, we limit the number of subchannels allocated to a MTR i by ζ^c.

3.4. Computing models

In our work, we adopt four computing models: (1) local computing, (2) D2D peer device computing, (3) edge computing, and (4) cloud computing. The computation delay and energy consumption are presented for every model as follows.

3.4.1. Local computing

The computation delay T_i^l of processing X_i data locally can be computed as follows:

$$T_i^l = \frac{X_i \cdot F_i}{F_i^l} \tag{4}$$

The energy consumed by MTR i to execute locally X_i bits can be expressed as follows:

$$E_i^l = \mathfrak{C}_i^l \cdot X_i \cdot (F_i^l)^2 \tag{5}$$

where $\mathfrak{C}_i^l$ is the local effective switched capacitance of MTR i, reflecting the energy consumption coefficient related to its CPU performance [14, 21].

3.4.2. D2D device computing

The total delay $T_{i,j}^d$ of offloading $Y_{i,j}$ to MT j includes computation and transmission delay as follows:

$$T_{i,j}^d = T_{i,j}^{dc} + T_{i,j}^{dt} \tag{6}$$

where $T_{i,j}^{dc}$ is the computation delay of device MT j to execute $Y_{i,j}$ requested by MTR i, and can be expressed as follows:

$$T_{i,j}^{dc} = \frac{Y_{i,j} \cdot F_i}{\mu_{i,j}^d \cdot F_j^d} \tag{7}$$

$T_{i,j}^{dt}$ represents the communication delay for transmitting $Y_{i,j}$ bits from MTR i to MT j can be expressed as follows:

$$T_{i,j}^{dt} = \frac{Y_{i,j}}{R_{i,j}^d} \tag{8}$$

where $R_{i,j}^d$ is the transmission rate of the D2D link. Note that the computation output data is normally much smaller than that of the input data. Accordingly, the downlink transmission time requires much lower transmission latency, and hence can be neglected [12].

In case of D2D communications, the total energy $E_{i,j}^d$ consumed by MTR i for offloading data to MT j includes communication (data transmission and reception), as well as, computation processing energy consumption as follows:

$$E_{i,j}^d = E_{i,j}^{dt} + E_{i,j}^{dr} + E_{i,j}^{dc} \tag{9}$$

$E_{i,j}^{dt}$ represents the energy consumed by MTR i to transmit $Y_{i,j}$ bits to MT j, and can be expressed as follows:

$$E_{i,j}^{dt} = P_i^{dt} \cdot T_{i,j}^{dt} = P_i^{dt} \cdot \frac{Y_{i,j}}{R_{i,j}^d} \tag{10}$$

where P_i^{dt} is the power consumed by MTR i to transmit data to MT j. $E_{i,j}^{dr}$ represents the energy consumed by MT j to receive data $Y_{i,j}$ from MTR i, and can be expressed as follows:

$$E_{i,j}^{dr} = P_j^{dr} \cdot T_{i,j}^{dt} = P_i^{dr} \cdot \frac{Y_{i,j}}{R_{i,j}^d} \tag{11}$$

where P_j^{dr} is the power consumed by MT j to receive data from MTR i. $E_{i,j}^{dc}$ is the energy consumed by MT j to execute $Y_{i,j}$, and can be expressed as follows:

$$E_{i,j}^{dc} = \mathfrak{C}_j^d \cdot Y_{i,j} \cdot (\mu_{i,j}^d \cdot F_j^d)^2 \tag{12}$$

where $\mathfrak{C}_j^d$ is the effective switched capacitance of MT j, reflecting the energy consumption coefficient related to its CPU performance [14,21].

3.4.3. Edge computing

The total delay $T_{i,n}^e$ for MEC offloading includes computation and transmission delay as follows:

$$T_{i,n}^e = T_{i,n}^{ec} + T_{i,n}^{et} \tag{13}$$

where $T_{i,n}^{ec}$ represents the computation delay of MEC n to execute $Z_{i,n}$ requested by MTR i, and expressed as follows:

$$T_{i,n}^{ec} = \frac{Z_{i,n} \cdot F_i}{\mu_{i,n}^e \cdot F_n^e} \tag{14}$$

The transmission delay $T_{i,n}^{et}$ represents the communication delay for transmitting $Z_{i,n}$ bits from MTR i to MEC n.

$$T_{i,n}^{et} = \frac{Z_{i,n}}{R_{i,n}^e} \tag{15}$$

where $R_{i,n}^e$ is the uplink transmission rate from MTR i to MEC n. In our work, we aim at minimizing the energy consumption of the mobile terminals. Accordingly, we do not consider in our formulation the energy consumed by the MEC and MCC for data processing. We then denote by $E_{i,n}^e$ the energy consumed by MTR i to transmit data to MEC n, expressed as follows:

$$E_{i,n}^e = P_i^{et} \cdot T_{i,n}^{et} = P_i^{et} \cdot \frac{Z_{i,n}}{R_{i,n}^e} \tag{16}$$

where P_i^{et} is the power consumed by MTR i to transmit $Z_{i,n}$ data bits to MEC n.

3.4.4. Cloud computing

The total MCC offloading delay T_i^c can be expressed as follows:

$$T_i^c = T_i^{cc} + T_i^{ct} \tag{17}$$

where the computation delay T_i^{cc} at MCC to execute W_i data from task U_i requested by MTR i can be expressed as follows:

$$T_i^{cc} = \frac{W_i \cdot F_i}{\mu_i^c \cdot F^c} \tag{18}$$

In our work, we assume the computation delay at the MCC is much smaller than the transmission delay due to the high computation capabilities of the cloud. Accordingly, we include in our problem formulation, a constraint guaranteeing the minimum cloud computation resources allocation to MTR i ensuring T_i^{cc} to be negligible (less than T^{Neg}). Hence, we consider $T_i^c = T_i^{ct}$ where T_i^{ct} represents the transmission delay of W_i bits to MCC.

$$T_i^{ct} = \frac{W_i}{R_i^c} \tag{19}$$

where R_i^c is the uplink transmission rate from MTR i to MCC. We consider E_i^c the data transmission energy consumption for MCC offloading that can be expressed as follows:

$$E_i^c = P_i^{ct} \cdot T_i^{ct} = P_i^{ct} \cdot \frac{W_i}{R_i^c} \tag{20}$$

where P_i^{ct} is the power consumed by MTR i to transmit data to MCC.

3.5. Pricing models

The prices included in the formulation incorporate the service cost to the operator. We consider the transmission and computation monetary cost for data task offloading. We assume the pricing model follows a usage-based pricing, which charges in proportion to the amount of data consumed. In general, some interfaces have much higher cost than others, e.g., cellular typically has higher cost than WiFi or Bluetooth. Note that the monetary cost is considered as an input variable to the proposed approaches and can be updated on a time slot basis in the case of dynamic pricing that may depend on various factors including peak hours, availability of resources, subscription type and market competition status.

3.5.1. *Pricing for D2D offloading*

In our work, we assume that the network operator pays the peer MTs incentives for sharing their radio and computation resources and allowing D2D offloading. We assume the peer MTs are paid $\phi^{dt}_{i,j}$ USD per bit as incentives for transmission over SR connectivity. We intend to make our formulation general and accommodate for any incentive and pricing strategy over any SR connectivity, which may also be free in some cases such as over Bluetooth. We denote by $\phi^{dc}_{i,j}$ the incentives in USD per Hz paid to MT j for sharing their computation resources. The D2D offloading cost can then be expressed as follows:

$$\phi^{d}_{i,j} = Y_{i,j} \cdot \phi^{dt}_{i,j} + \mu^{d}_{i,j} \cdot F^{d}_{j} \cdot \phi^{dc}_{i,j} \tag{21}$$

3.5.2. *Pricing for MEC offloading*

We denote by $\phi^{et}_{i,n}$ the amount of USD charged per bit for transmission over WiFi, and $\phi^{ec}_{i,n}$, the amount of USD charged per Hz for computation processing at MEC n. The MEC offloading cost can then be expressed as follows:

$$\phi^{e}_{i,n} = Z_{i,n} \cdot \phi^{et}_{i,n} + \mu^{e}_{i,n} \cdot F^{e}_{n} \cdot \phi^{ec}_{i,n} \tag{22}$$

3.5.3. *Pricing for cloud offloading*

Similarly, we denote by ϕ^{ct}_{i} the amount of USD charged per bit for transmission over the cellular network, and ϕ^{cc}_{i} the amount of USD charged per Hz for computation processing at MCC. The MCC offloading cost can then be expressed as follows:

$$\phi^{c}_{i} = W_{i} \cdot \phi^{ct}_{i} + \mu^{c}_{i} \cdot F^{c} \cdot \phi^{cc}_{i} \tag{23}$$

4. Optimal joint computing, communication and cost-aware task offloading

In this section, we formulate the joint computing, communication and cost-aware task offloading optimization problem. We present first the multi-objective function in Section 4.1. We then differentiate between two sets of constraints: (1) computation resource allocation constraints in Section 4.2, and (2) communication and radio resource allocation constraints in Section 4.3. The problem is a mixed-integer non-linear program, which we linearize in Section 4.4.

4.1. *Optimal joint computing, communication and cost-aware task offloading: Multi-objective function*

The objective function of our joint computing, communication and cost-aware offloading can be expressed as follows:

$$\underset{\substack{\mathbf{X},\mathbf{Y},\mathbf{Z},\mathbf{W},\boldsymbol{\pi}\\ \boldsymbol{\mu}^{d},\boldsymbol{\mu}^{e},\boldsymbol{\mu}^{c}\\ \boldsymbol{L}^{d},\boldsymbol{L}^{e},\boldsymbol{L}^{c},\boldsymbol{\eta}^{c}}}{\operatorname{argmax}} \quad \beta_1 \frac{\Pi}{A} - \beta_2 \frac{\Psi}{\Psi_{max}} - (1 - \beta_1 - \beta_2)\frac{\Phi}{\Phi_{max}} \tag{24}$$

The objective function in (24) represents the weighted sum of the three objectives aiming at maximizing the number of completed tasks $\Pi = \sum_{i=1}^{A} \pi_i$, while minimizing the energy consumption Ψ and monetary cost Φ. We used normalization in (24) to adjust the different parameters of different scales to a common scale ranging between 0 and 1. We assume Ψ_{max} and Φ_{max} are the maximum energy consumption and monetary cost, where all the MTRs are served with the highest cost. The minimum energy and monetary cost consumption is assumed to be 0 when no transmission occurs. The expressions for Ψ, Ψ_{max}, Φ and Φ_{max} are detailed in the next subsections. β_1 and β_2 are positive coefficients indicating the impact of maximizing the number of completed tasks and minimizing the energy consumption, respectively. β_1 and β_2 vary between 0 and 1 giving weights to the normalized values of tasks completed and energy consumed, respectively.

4.1.1. *Objective function — minimizing energy consumption*

The total energy consumption of the system is denoted by Ψ and can be expressed as follows:

$$\Psi = \sum_{i=1}^{A} E_i = \sum_{i=1}^{A} \left(E^{l}_{i} + \sum_{j=1}^{J} E^{d}_{i,j} + \sum_{n=1}^{N} E^{e}_{i,n} + E^{c}_{i} \right) \tag{25}$$

where E_i is the energy consumed by MTR i for local computing, D2D, MEC and MCC offloading. E^{l}_{i}, $E^{d}_{i,j}$, $E^{e}_{i,n}$ and E^{c}_{i} are expressed in (5), (9), (16) and (20), respectively. We assume Ψ_{max} to be the maximum energy consumed when all the MTRs are served using the maximum energy consuming interface. Ψ_{max} can be expressed as follows:

$$\Psi_{max} = \sum_{i=1}^{A} \psi^{max}_{i} \tag{26}$$

where ψ^{max}_{i} is the maximum energy consumed for processing all the data D_i of the task U_i requested by MTR i. ψ^{max}_{i} can be expressed as follows: $\psi^{max}_{i} = \max(\psi^{l}_{i}, \psi^{d}_{i}, \psi^{e}_{i}, \psi^{c}_{i})$. We denote by ψ^{l}_{i}, ψ^{d}_{i}, ψ^{e}_{i} and ψ^{c}_{i}, the maximum energy consumed for processing all the data D_i of the task U_i requested by MTR i using local computation, D2D, MEC and MCC offloading, respectively.

4.1.2. *Objective function — minimizing monetary cost*

We aim at minimizing the total monetary cost Φ of all the requester MTRs, which can be expressed as follows:

$$\Phi = \sum_{i=1}^{A} \phi_i = \sum_{i=1}^{A} \left(\sum_{j=1}^{J} \phi^{d}_{i,j} + \sum_{n=1}^{N} \phi^{e}_{i,n} + \phi^{c}_{i} \right) \tag{27}$$

where ϕ_i is the monetary cost for completing task U_i requested by MTR i. $\phi^{d}_{i,j}$, $\phi^{e}_{i,n}$ and ϕ^{c}_{i} are expressed in (21), (22) and (23), respectively. We assume Φ_{max} to be the maximum monetary cost consumed when all the MTRs are served using the highest interface cost. Φ_{max} can be expressed as follows:

$$\Phi_{max} = \sum_{i=1}^{A} \phi^{max}_{i} \tag{28}$$

where $\phi^{max}_{i} = \max(\Gamma^{d}_{i}, \Gamma^{e}_{i}, \Gamma^{c}_{i})$ denotes the maximum cost consumed for processing all the data D_i of the task U_i requested by MTR i. We denote by Γ^{d}_{i}, Γ^{e}_{i} and Γ^{c}_{i} the maximum monetary cost consumed for transmitting and processing all the data D_i of the task U_i requested by MTR i using D2D, MEC, MCC task offloading, respectively.

4.2. *Optimal joint computing, communication and cost-aware task offloading: Computation resources constraints*

The multi-objective optimization problem is subjected to computation resource allocation constraints and limitations as follows:

$$0 \le X_i \le D_i \cdot \pi_i, \forall i \tag{29}$$

$$0 \le Y_{i,j} \le D_i \cdot \pi_i \cdot \upsilon^{d}_{i,j}, \forall i, \forall j \tag{30}$$

$$0 \le Z_{i,n} \le D_i \cdot \pi_i \cdot \upsilon^{e}_{i,n}, \forall i, \forall n \tag{31}$$

$$0 \le W_i \le D_i \cdot \pi_i, \forall i \tag{32}$$

$$Y_{i,j} \cdot \epsilon \le \mu^{d}_{i,j} \le \pi_i \cdot \upsilon^{d}_{i,j}, \forall i, \forall j \tag{33}$$

$$Z_{i,j} \cdot \epsilon \le \mu^{e}_{i,n} \le \pi_i \cdot \upsilon^{e}_{i,n}, \forall i, \forall n \tag{34}$$

$$W_i \cdot \epsilon \le \mu^{c}_{i} \le \pi_i, \forall i \tag{35}$$

$$\sum_{i=1}^{A} \mu^{d}_{i,j} \le 1, \forall j \tag{36}$$

$$\sum_{i=1}^{A} \mu^{e}_{i,n} \le 1, \forall n \tag{37}$$

$$\sum_{i=1}^{A} \mu^{c}_{i} \le 1 \tag{38}$$

$$\mu^d_{i,j} \leq Y_{i,j}, \forall i, \forall j \quad (39)$$

$$\mu^e_{i,n} \leq Z_{i,n}, \forall i, \forall n \quad (40)$$

$$\mu^c_i \leq W_i, \forall i \quad (41)$$

$$T_i \leq T^{max}_i, \forall i \quad (42)$$

$$T^{cc}_i \leq T^{Neg}, \forall i \quad (43)$$

$$G_i = D_i \cdot \pi_i, \forall i \quad (44)$$

- Constraints (29) to (32) indicate the upper and lower bounds for the data size to be executed locally, or offloaded to a cooperating node. Moreover, the constraints ensure that no data is offloaded if the task cannot be completed, or the MTR is out of the coverage range of the cooperating node.
- Constraints (33) to (35) ensure that the computation resources are allocated to MTR i only if the task U_i will be completed, and data tasks are sent to the peer MTs, MEC or MCC within their corresponding coverage range.
- Constraints (36) to (38) ensure that the computation resources allocated to the MTRs is less than or equal to the computation capacity of every MT, MEC or MCC.
- Constraints (39) to (41) ensure that the computation resources of every MT, MEC or MCC are allocated only when data tasks are offloaded from the MTRs.
- Constraint (42) guarantees that the time for completing a task should be less than the maximum time limit T^{max}_i. In our work, we assume parallel processing where MTR i can locally process while transmitting and receiving data. Accordingly, the latency of a task U_i, denoted by T_i, is the maximum delay caused by the computation models adopted, and can be expressed as follows:

  $$T_i = \max(T^l_i, T^d_i, T^e_i, T^c_i) \quad (45)$$

  where T^d_i, T^e_i and T^c_i are the maximum latency caused by D2D offloading to any MT j, edge offloading to any MEC n and cloud offloading, respectively.
- Constraint (43) ensures that the computation time at the cloud is negligible, which enforces minimum computation resource μ^c_i allocation by the MCC to MTR i.
- Constraint (44) is used to indicate whether the task U_i requested by MTR i is considered completed, i.e.: the computation data task D_i is completely offloaded and executed within T^{max}_i. The decision variable π_i should be set to one, indicating the completion of the task U_i when the total amount of data offloaded G_i expressed in (1) is equal to the data size D_i of the task U_i requested by MTR i, and 0 otherwise. Accordingly, when the task cannot be completed due to the lack of computation or radio resources, π_i is set to zero; in addition, no data should be offloaded or allocated for processing, hence, setting G_i to zero.

4.3. *Optimal joint computing, communication and cost-aware task offloading: Communication resources constraints*

The optimization problem is also subjected to communication and radio resource allocation constraints and limitations as follows:

$$L^d_{i,j} \leq Y_{i,j}, \forall i, \forall j \quad (46)$$

$$L^d_{i,j} \geq \mu^d_{i,j}, \forall i, \forall j \quad (47)$$

$$L^d_{i,j} \leq \pi_i, \forall i, \forall j \quad (48)$$

$$L^e_{i,n} \leq Z_{i,n}, \forall i, \forall n \quad (49)$$

$$L^e_{i,n} \geq \mu^e_{i,n}, \forall i, \forall n \quad (50)$$

$$L^e_{i,n} \leq \pi_i, \forall i, \forall n \quad (51)$$

$$L^c_i \leq W_i, \forall i \quad (52)$$

$$L^c_i \geq \mu^c_i, \forall i \quad (53)$$

$$L^c_i \leq \pi_i, \forall i \quad (54)$$

$$\sum_{j=1}^{J} L^d_{i,j} \leq \zeta^d, \forall i \quad (55)$$

$$\sum_{n=1}^{N} L^e_{i,n} \leq \zeta^e, \forall i \quad (56)$$

$$\sum_{i=1}^{A} L^d_{i,j} \leq \Omega^d, \forall j \quad (57)$$

$$\sum_{i=1}^{A} L^e_{i,n} \leq \Omega^e, \forall n \quad (58)$$

$$\sum_{i=1}^{A} L^c_i \leq \Omega^c, \forall n \quad (59)$$

$$\mu^c_i \leq \eta^c_i \leq L^c_i \cdot \zeta^c, \forall i \quad (60)$$

$$\sum_{i=1}^{A} \eta^c_i \leq \Omega^c \quad (61)$$

- Constraints (46) to (48) guarantee that a D2D channel is reserved for the transmission between MTR i and MT j, in case MT j helps MTR i in executing part of its computation task. Accordingly, the decision variables $L^d_{i,j}$ are set to one indicating a channel reservation for the transmission between MTR i and MT j when (a) the data $Y_{i,j}$ is offloaded to MT j (Constraint (46)), (b) a fraction of the MT j computation resources is allocated to serve MTR i(47), and (c) the task will be completed (Constraint (48)). A connection is then established and a channel is allocated only if the task can be completed and computation data is offloaded to the corresponding node within its limited coverage range.
- Similar to (46) to (48), the constraints (49) to (51) and (52) to (54) guarantee that a WiFi and a cellular link are reserved for MTR i for MEC and MCC offloading, respectively.
- Constraints (55) and (56) limit number of D2D and WiFi connections of MTR i to ζ^d and ζ^e, respectively.
- Constraints (57), (58) and (59) ensure that the number of devices served by a peer MT j, MEC n and MCC is less than Ω^d, Ω^e and Ω^c, respectively.
- Constraint (60) ensures that the number of subchannels η^c_i allocated to MTR i for MCC offloading is less than ζ^c. Moreover, η^c_i is set to zero if no MCC offloading occurs.
- Constraint (61) ensures that the number of subchannels allocated to all the MTRs is less than the maximum number of cellular subchannels Ω^c.

4.4. *Problem linearization and complexity*

The problem is a mixed-integer non-linear program. The non-linearity comes from the objective function when energy consumption is considered, in addition to constraint (42). To reduce the complexity of the problem, we transform it into a mixed-integer linear program by assumption that the computation resources are assigned in terms of chunks; i.e.: the decision variables $\boldsymbol{\mu}^d$ and $\boldsymbol{\mu}^e$ representing the fraction of assigned computation resources are then transformed from continuous to discrete variables. This conversion leads to some discretization error that can be significantly reduced by increasing the number of discrete values at the expense of increased complexity. We then provide a corresponding transformation of the problem using several linearization techniques as follows [32].

4.4.1. *Constraints linearization*

Considering maximizing the number of completed tasks as the only objective with $\beta_1 = 1$ and $\beta_2 = 0$, the non-linearity of the problem arises solely from constraint (42), which guarantees that the time for completing a task should be less than the maximum time limit T_i^{max}. As presented in (45), we aim at ensuring that the local computing latency T_i^l, D2D offloading latency $T_{i,j}^d$ to MT j, MEC offloading latency $T_{i,n}^e$ to MEC n, and MCC offloading latency T_i^c, are all less than T_i^{max}. Accordingly, constraint (42) can first be replaced by the following constraints:

$$T_i^l \leq T_i^{max}, \forall i \tag{62}$$

$$T_{i,j}^d \leq T_i^{max}, \forall i, \forall j \tag{63}$$

$$T_{i,n}^e \leq T_i^{max}, \forall i, \forall n \tag{64}$$

$$T_i^c \leq T_i^{max}, \forall i \tag{65}$$

However, constraints (63) and (64) are non-linear due to the non-linearity of the D2D and MEC offloading processing latency, expressed in (7) and (14), respectively. To eliminate the non-linearity in (63), we replace the product of the two continuous variables $(Y_{i,j} \cdot \mu_{i,j}^d)$ by a new variable $\widehat{Y}_{i,j}$ as follows. We first assume the computation resources of a MT j are divided into C^d chunks of equal size S^c. We then transform the continuous variable $\mu_{i,j}^d$ to a discrete variable represented by a sum of binary variables $a_{i,(j,c)}^d$ indicating the allocation of the computation resources chunk c of MT j to MTR i. Accordingly, $\mu_{i,j}^d$ can be expressed in terms of the binary variables $a_{i,(j,c)}^d$ as follows: $\mu_{i,j}^d = \frac{1}{C^d} \sum_{c=1}^{C_j^d} a_{i,(j,c)}^d$. The product of $(Y_{i,j} \cdot \mu_{i,j}^d)$ can then be replaced by $\widehat{Y}_{i,j}$, expressed as follows:

$$\widehat{Y}_{i,j} = \sum_{c=1}^{C_j^d} \widehat{Y}_{i,(j,c)} \tag{66}$$

where $\widehat{Y}_{i,(j,c)}$ represents the product of $(Y_{i,j} \cdot a_{i,(j,c)}^d)$. Additional constraints are then needed to force $\widehat{Y}_{i,(j,c)}$ to take the value of the product of $(Y_{i,j} \cdot a_{i,(j,c)}^d)$, as follows:

$$\widehat{Y}_{i,(j,c)} \leq D_i \cdot a_{i,(j,c)}^d \tag{67}$$

$$\widehat{Y}_{i,(j,c)} \leq Y_{i,j} \tag{68}$$

$$\widehat{Y}_{i,(j,c)} \geq Y_{i,j} - D_i \cdot (1 - a_{i,(j,c)}^d) \tag{69}$$

$$\widehat{Y}_{i,(j,c)} \geq 0 \tag{70}$$

Similarly, constraint (64) can be linearized by introducing a new variable $\widehat{Z}_{i,n}$, representing the product of $(Z_{i,n} \cdot \mu_{i,n}^e)$ with $a_{i,(n,q)}^e$ and $\widehat{Z}_{i,(n,q)}$ representing the product of $(Z_{i,n} \cdot a_{i,(n,q)}^e)$ for $q = [1, \ldots, C^e]$, where C^e is the number of computation chunks resources, in addition to the needed constraints.

The task offloading problem aiming only at maximizing the number of completed tasks, $\Pi = \sum_{i=1}^{A} \pi_i$, is then transformed to a mixed-integer linear programming (MILP). In addition to Table 1, the decision variables are: a^d and $\widehat{Y}$ which are 3-dimensional matrices of size $J \times A \times C^d$, a^e and $\widehat{Z}$ which are 3-dimensional matrices of size $N \times A \times C^e$.

4.4.2. *Objective function linearization*

Minimizing energy consumption in addition to maximizing the number of completed tasks, make the problem MINLP. In this case, the objective function in (25) is non-linear due to the non-linear parameters $E_{i,j}^{dc}$ and E_i^c. To linearize the $E_{i,j}^{dc}$, we replace the product of $(\widehat{Y}_{i,j} \cdot \mu_{i,j}^d)$ by a new variable $\bar{Y}_{i,j}$, as well as introducing additional constraints as previously presented. To linearize E_i^c, we replace the continuous integer variable η_i representing the number of subchannels allocated to a requester MTR i by a sum of binary variables $a_{i,w}^c$ where $w \in [1, \ldots \zeta^c]$, $\eta_i = \sum_{w=1}^{\zeta^c} a_{i,w}^c$. We introduce $\widehat{\eta}_i$ and let $\widehat{\eta}_i = \frac{1}{\eta_i}$. Accordingly, $\widehat{\eta}_i$ will have discrete values varying from $[0, \frac{1}{\zeta^c}, \frac{1}{\zeta^c - 1}, \ldots, 1]$, which we represent by binary variables $\widehat{a}_{i,w}^c$. We then introduce a new variable $\widehat{W}_{i,w}$ representing the product of the two variables $\widehat{a}_{i,w}^c$ and W_i, as well as new constraints. Accordingly, new decision variables are needed in addition to the previous variables: $\widehat{\eta}$ a vector of length A, $\widehat{a}^c$, and $\widehat{W}$, matrices of size $A \times (\zeta^c + 1)$.

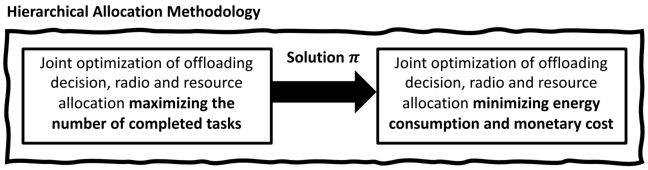

Hierarchical Allocation Methodology
Joint optimization of offloading decision, radio and resource allocation maximizing the number of completed tasks
Solution π
Joint optimization of offloading decision, radio and resource allocation minimizing energy consumption and monetary cost

Fig. 2. Hierarchical allocation for joint computing, communication and cost-aware task offloading.

4.4.3. *Complexity analysis*

The multi-objective optimization allocation (OA) problem aiming at jointly optimizing offloading decision, communication and computation resource allocation is NP-hard. In fact, the problem can be mapped to Generalized Assignment Problem (GAP) with multi-resources constraints, which is known to be NP-hard [33–36]. The complexity of the problem is heavily impacted by the number of possibilities a computation task can be divided into multiple parts and executed remotely at multiple bins, in our case nodes including peer MTs, edge servers and cloud. These nodes are considered as agents with limited radio and computation resources. They are assigned parts of the computation tasks to be performed without exceeding their capacity budget while maximizing the total profit of the assignment which is in our case the number of completed tasks, energy consumption and monetary cost while meeting latency, radio and computation constraints. Moreover, the problem involves both binary and real variables, and has a quite large size due the large number of nodes. The total number of variables involved is equal to $A\left(J\left(2+3C^d\right)+2N\left(1+C^e\right)+2\zeta^c+8\right)$. Hence, global optimal solution may be unfeasible and hard to obtain in reasonable time especially when the number of mobile terminals grows. Therefore, we propose two sub-optimal approaches to reduce the optimal problem time complexity.

5. Problem decomposition: Hierarchical allocation

The complexity of the multi-objective optimization allocation (OA) comes mainly from the large number of decision variables and constraints especially in large scale networks. Therefore, to reduce the complexity of the joint multi-objective optimization problem, we propose decomposing the problem into two optimization sub-problems with different objectives, which has large impact on reducing the number of decision variables and system constraints. Hence, as presented in Fig. 2, we decompose the problem into two optimization sub-problems solved consecutively as follows: (1) we solve the optimal allocation while maximizing the number of completed tasks only, (2) we use the solution of the problem π to identify the MTRs served, and (3) we then solve the problem of decision offloading, radio and computation resource allocation for the MTRs determined by π while minimizing the total energy consumption and monetary cost. In other words, given the tasks completed, we optimally offload the computation data and allocate radio and computation resources with minimum energy and monetary cost. In this case, π is no longer considered as a decision variable and the number of requesters is reduced to the number of MTRs served only. This allows the proposed HA approach to provide optimal number of completed tasks, with sub-optimal performance in terms of energy consumption and monetary cost, compared to those of the multi-objective OA but with less execution time. In scenarios where all MTRs can be served, the proposed HA approach may not have

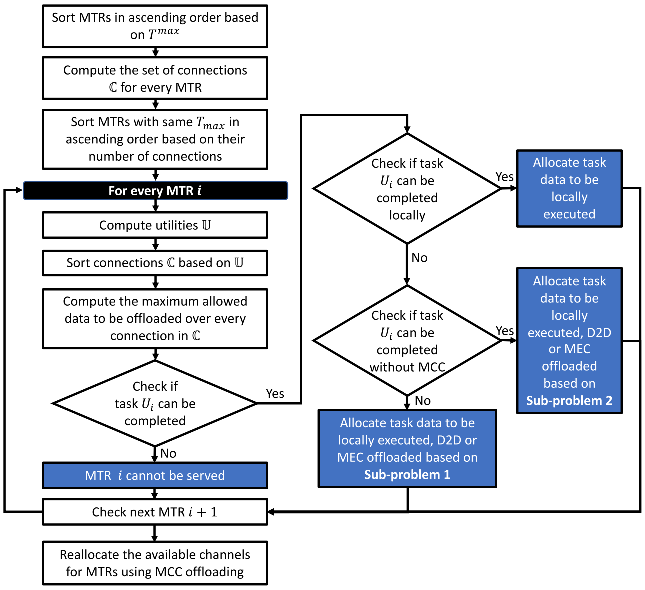

Sort MTRs in ascending order based on T^{max}
Compute the set of connections ℂ for every MTR
Sort MTRs with same T_{max} in ascending order based on their number of connections
For every MTR i
Compute utilities 𝕌
Sort connections ℂ based on 𝕌
Compute the maximum allowed data to be offloaded over every connection in ℂ
Check if task U_i can be completed
Yes
No
MTR i cannot be served
Check next MTR $i+1$
Reallocate the available channels for MTRs using MCC offloading
Check if task U_i can be completed locally
Yes
Allocate task data to be locally executed
No
Check if task U_i can be completed without MCC
Yes
Allocate task data to be locally executed, D2D or MEC offloaded based on Sub-problem 2
No
Allocate task data to be locally executed, D2D or MEC offloaded based on Sub-problem 1

Fig. 3. Iterative allocation for joint computing, communication and cost-aware task offloading.

high impact on reducing the execution time. Similarly to the multi-objective OA, the HA problem is NP-hard. Optimal solutions may not be achievable in real-time for dense D2D-enabled Het-MEC networks. This shows the importance of finding real-time fast sub-optimal solutions providing a balance between time complexity, number of completed tasks, energy consumption and monetary cost for large-scale networks with low latency applications.

6. Iterative allocation for joint computing, communication and cost-aware task offloading

In this section, we propose an iterative allocation (IA) approach addressing jointly offloading decision, radio and computation resource allocation while providing fast sub-optimal solutions with low time complexity. The MTRs data tasks are assigned sequentially to be offloaded to MTs, MECs and cloud, while reducing energy consumption and monetary cost. We design our proposed IA approach to complete the largest number of completed tasks by giving priority to MTRs with lower deadline first, restricting any computation offloading in case the task can be performed locally, selecting offloading strategies that provide the lowest energy consumption and monetary cost, and reducing the usage of external resources by reserving the cloud resources for serving more MTRs by favoring completion of the task without MCC offloading.

As presented in Fig. 3 and detailed in Algorithm 1, the MTRs are first sorted in ascending order to be served based on their task deadline T_i^{max}, and then based on the number of possible connections which can be established with the different nodes in their proximity (lines 1–4). The requester tasks are then processed to be offloaded sequentially (lines 5–23). Accordingly, for every MTR i, the performance of every connection l is evaluated based on a general utility 𝕌 reflecting the trade-off provided between monetary cost and energy consumption (lines 6–9). 𝕌 is composed of the different utilities $u_{i,l}$ for processing D^u bits using maximum computation capacity of node p over connection l. Utility $u_{i,l}$ can be computed as follows:

$$u_{i,l} = \beta u^e_{i,l} + (1-\beta) u^c_{i,l} \tag{71}$$

Algorithm 1: The proposed iterative allocation (IA) for joint computing, communication and cost-aware task offloading

```
Input  : System parameters presented in Table 1
Output: - Tasks completed: $\pi_i$
        - Data task offloading decision: $X_i$, $Y_{i,j}$, $Z_{i,n}$, $W_i$
        - Radio resource allocation: $L^d_{i,j}$, $L^e_{i,n}$, $L^c_i$, $\eta^c_i$
        - Computation resource allocation: $\mu^d_{i,j}$ ,$\mu^e_{i,n}$ ,$\mu^c_i$
1  Sort the MTRs in ascending order based on $T_i^{max}$
2  Check the nodes (MTs and MECs) within coverage range for every MTR
3  Create a list of possible connections ℂ for every MTR
4  Sort MTRs with same $T_i^{max}$ in ascending order based on the number of
     connections
5  Assign data task and allocate radio and computation resources for a MTR $i$
6      (a) Compute the general utilities 𝕌 for connections ℂ as follows:
7          • Check every possible connection $l$ if it is a MT $j$, MEC $n$ or MCC
8          • Estimate the transmission rate over $l$
9          • Compute utility $u_{i,l}$ for using connection $l$ based on (71)
10     (b) Sort connections ℂ based on 𝕌
11     (c) Select the maximum number of D2D and MEC connections allowed with
         lowest $u_{i,l}$
12     (d) Compute the maximum data $\mathbb{D}_{i,l}$ size that can be processed within $T_i^{max}$
         locally based on (4), and using connection $l$ based on (6) when $l$ is a MT,
         (13) when $l$ is a MEC, and (18) and (19) when $l$ is the MCC.
13     (e) If task $U_i$ of MTR $i$ cannot be completed
14             • Set $\pi_i \leftarrow 0$
15         else task $U_i$ can be completed
16             • Set $\pi_i \leftarrow 1$
17             • if task $U_i$ can be completed locally
18                 -Set $X_i \leftarrow D_i$
19             • elseif task $U_i$ cannot be completed without MCC offloading
20                 - Offload data based on sub-problem1
21             • else
22                 - Offload data to MTs and MECs based on sub-problem2
23     (f) Update the remaining available resources
24 Repeat process (lines 5–23) for all MTR $i+1$
25 Reallocate available channels left to MTRs using MCC offloading
```

Sub-problem 1: The proposed IA when task cannot be completed without MCC offloading in Algorithm 1- line 20

```
1  Allocate the maximum allowed data to be processed locally and over every
     connections $l$ within $T_i^{max}$ using all the remaining available computation
     resources $\mu^p_{i,l}$ of the node $p$ as follows:
2      If $l$ corresponds to a connection with MT $j$
3          • Set $Y_{i,j} \leftarrow \mathbb{D}_{i,l}$ and set $L^d_{i,j} \leftarrow 1$
4          • Set $\mu^p_{i,l} \leftarrow \left(1 - \sum_{i=1}^{A} \mu^d_{i,j}\right)$ and set $\mu^d_{i,j} \leftarrow \mu^p_{i,l}$
5      else $l$ corresponds to a connection with MEC $n$
6          • Set $Z_{i,n} \leftarrow \mathbb{D}_{i,l}$ and set $L^e_{i,n} \leftarrow 1$
7          • Set $\mu^p_{i,l} \leftarrow \left(1 - \sum_{i=1}^{A} \mu^e_{i,n}\right)$ and set $\mu^e_{i,n} \leftarrow \mu^p_{i,l}$
8  Offload the remaining data to the MCC with minimum number of cellular
     subchannels
9      • Set $W_i \leftarrow \left(D_i - X_i - \sum_{j=1}^{J} Y_{i,j} - \sum_{n=1}^{N} Z_{i,n}\right)$
10     • Set $L^c_i \leftarrow 1$,
11     • Allocate minimum computation resources to execute $W_i$ within a negligible
         computation time based on (18)
12     • Allocate minimum number of subchannels to execute $W_i$ within $T_i^{max}$
         based on (19)
```

where $u^e_{i,l} = E_{i,l}/E^{max}$ and $u^c_{i,l} = \phi_{i,l}/\phi^{max}$, $E_{i,l}$ and $\phi_{i,l}$ are the energy and cost consumed for processing D^u bits by the corresponding node, respectively. β weight factor indicating the impact of minimizing energy consumption. E^{max} and ϕ^{max} are the maximum energy and cost consumed for processing D^u for all MTRs. The connection l providing the minimum utility represents the most efficient connection with low energy and cost. Accordingly, the connections are sorted in ascending order based on their utilities (line 10). The maximum allowable number of D2D and MEC connections are selected based on their utilities and limited by ζ^d and ζ^e, respectively (line 11). The maximum allowed data to be offloaded using every connection considering transmission and computation time limitation is computed to check if the MTR can be served (line 12). If the amount of data allowed over all the interfaces including local execution, D2D, MEC and MCC offloading is less than D_i, the task cannot be completed; π_i is set to zero and MTR is considered not served (lines 13–14). Otherwise, MTR i can be served, and π_i is set to one (lines 15–22). First, the IA approach restricts any computation offloading in case the task can be performed locally to reduce the usage of external resources. In addition, it reserved the cloud resources for serving more MTRs by favoring completion of the task without MCC offloading. Accordingly, if task U_i cannot be performed without the use of the cloud, the minimum cloud resources to accomplish MTR i task are assigned based on sub-problem 1. Otherwise, task U_i can be performed without MCC; that is, the offloading decision and the resource allocation are performed based on sub-problem 2 while using D2D and MEC offloading only. When all the tasks are processed, the remaining cellular subchannels are distributed to the MTRs using MCC offloading to increase the transmission rate, hence, reducing the energy consumption (line 25).

As detailed in Sub-problem 1, if task U_i cannot be accomplished without MCC, the maximum allowed data to be offloaded using all the possible connections with peer MTs and MECs are first assigned (lines 1–7). The remaining data needed is assigned to the cloud with minimum number of subchannels and computation resources (lines 8–12).

If U_i can be accomplished without the MCC, Sub-problem 2 is used to offload data to peer MTs and MECs only. First, the local computation resources are fully assigned (line 1), then the connections are evaluated based on their performance in terms of energy and monetary cost (lines 3–7). A new list of connections $m \in \widehat{\mathbb{C}}$ is created based on duplicates of the connections $l \in \mathbb{C}$ using different chunk size of computation resources available at a node p. The connections are then sorted based on their utility $\hat{u}_{i,m}$ for processing D^u bits using specific computation resources $\hat{\mu}^p_{i,m}$. Every connection m is then processed until all the data of task U_i is assigned (lines 8–21). If a previous connection is established with the corresponding node p, larger amount of computation resources are then needed. Hence, the previous resource allocation with node p are released (lines 11–14). The system parameters are then updated based on the new assigned resources (line 15). The process is then repeated until all the data D_i is assigned.

6.1. Complexity analysis

Our proposed IA starts by sorting the MTRs based on their task deadlines and number of connections. Quicksort algorithm may be used with complexity of $O(nlogn)$. Then, the MTR are served sequentially until all the A requesters are served. For every MTR, the number of connections will be limited to $C_1 = \zeta^d + \zeta^e + 1$. Accordingly, we compute the utility function and maximum data to be assigned using maximum of C_1 connections. If the task cannot be completed locally and requires the use of cloud resources to complete the task, all the resources are fully utilized; i.e.: the decision variables $\boldsymbol{Y}$, $\boldsymbol{L^d}$ and $\boldsymbol{u^d}$ for maximum ζ^d peer MTs will be set, and $\boldsymbol{Z}$ and $\boldsymbol{u^e}$ for maximum ζ^e MECs will be set. The remaining data is computed assigned to be processed at the cloud. Hence, the complexity of Sub-problem 1 is very low where system parameters for maximum C_1 connections are directly set.

If the task can be completed without the use of the cloud, as presented in Sub-problem 2, $C_2 = C_1 \times C^d$ utilities should be computed to consider the different combinations of resource allocation, where the available resources are divided into a maximum of C^d chunks. These utilities are checked consecutively until all the data of the MTR is offloaded. The worst scenario is when the task size is large and requires almost all the available resources of the ζ^d peer MTs and ζ^e MECs, hence, all the C_2 utilities are checked. Note that the number of connections is limited in real-life scenario due the limitation of the number of orthogonal channels assigned to the MTRs and to the available cooperating peer MTs and MEC servers in proximity of the requester. For instance, in our work, we assume the resources are divided into $C^d = C^e$=10 equal chunk sizes and a MTR i can offload its computation data to up to $\zeta^d = 7$ peer MTs over Bluetooth, $\zeta^e = 5$ MEC servers over WiFi, simultaneously. Accordingly, 120 utilities will be considered for data offloading and resource allocation. Numerical evaluation of the time complexity of the proposed IA is presented in Table 2 and analyzed in Section 7.3.3.

Sub-problem 2: The proposed IA when task can be completed without MCC offloading in Algorithm 1- line 22

```
1  Allocate maximum data $X_i$ to be executed locally within $T_i^{max}$ based on (4)
2  Set the remaining data $\mathbb{L} \leftarrow D_i - X_i$
3  Create a new list of connections $\widehat{\mathbb{C}}$ based on duplicates of the connections $l \in \mathbb{C}$
     with different chunk size of computation resources allocation
4      (a) Check the available computation resources $\mu^p_{i,l}$ of node $p$ (MT or MEC)
         linked with connection $l$
5      (b) Create connections $m \in \widehat{\mathbb{C}}$, which are sub-connections of connection $l$ and
         node $p$ with different computation resource allocation
         $\hat{\mu}^p_{i,m} \in [S^c_p, 2S^c_p, ..., \mu^p_{i,l}]$, where $S^c_p$ is the minimum chunk size of
         computation resource allocated by node $p$
6  Compute $\hat{u}_{i,m}$ of processing $D^u$ data bits at the corresponding node $p$ using
     specific computation resources $\hat{\mu}^p_{i,m}$ based on (71)
7  Sort connections $\widehat{\mathbb{C}}$ in ascending order based on $\widehat{\mathbb{U}}$
8  While the remaining data $\mathbb{L} > 0$
9      (a) Identify the node $p$ as MT $j$ or MEC $n$
10     (b) Compute the maximum allowed data $\widehat{\mathbb{D}}_m$ to be transmitted and processed
         using connection $m$ at node $p$ using computation resources $\mu^p_{i,m}$ based on
         (6) if $p$ is MT $j$, and based on (13) if $p$ is MEC $n$
11     (c) If connection $m$ corresponds to a connection with MT $j$
12         • If previous connection was established with MT $j$; i.e.: $L^d_{i,j}$=1
13             - Larger amount of computation resources is needed
14             - Set $\mu^d_{i,j} \leftarrow 0$ and set $\mathbb{L} \leftarrow \mathbb{L} + Y_{i,j}$
15         • Set $Y_{i,j} \leftarrow \min(\mathbb{L}, \widehat{\mathbb{D}}_m)$, set $\mu^d_{i,j} \leftarrow \mu^p_{i,m}$, set $\mathbb{L} \leftarrow \mathbb{L}_i - Y_{i,j}$ and set $L^d_{i,j} \leftarrow 1$
16     else connection $m$ corresponds to a connection with MEC $n$
17         • If previous connection was established with MEC $n$; i.e.: $L^e_{i,n}$=1
18             - Larger amount of computation resources is needed
19             - Set $\mu^e_{i,n} \leftarrow 0$ and set $\mathbb{L} \leftarrow \mathbb{L} + Z_{i,n}$
20         • Set $Z_{i,n} = \min(\mathbb{L}, \widehat{\mathbb{D}}_m)$, set $\mu^e_{i,n} \leftarrow \mu^p_{i,m}$, set $\mathbb{L} \leftarrow \mathbb{L}_i - Z_{i,n}$ and set $L^e_{i,n} \leftarrow 1$
21     (d) Repeat process lines 9–20 until $D_i$ is completed and $\mathbb{L} = 0$
```

7. Performance results and analysis

In this section, we evaluate the performance of the proposed optimal and sub-optimal approaches. We first present the performance evaluation, simulation setup including case study topology, assumptions and system parameters. Then, to validate the proposed iterative allocation (IA) approach, we compare its performance to the multi-objective OA and sub-optimal HA approaches in terms of number of completed tasks, energy consumption, monetary cost and execution time. Lastly, we evaluate the performance of the proposed IA approach under different system parameters and models adopted from the literature.

7.1. Performance evaluation

In order to assess the performance effectiveness of the proposed IA approach, we generated results for the following different baseline strategies (1–4) and system models (5–8) adopted from the literature [8,12–14,37]:

1. *Complete Local Execution (CLE)*: the task is allowed to be only executed locally at the MTR without offloading.
2. *Binary Offloading- Complete D2D (BO-CD)*: the task is allowed to be fully offloaded to one peer MT.
3. *Binary Offloading- Complete MEC (BO-CM)*: the task is allowed to be fully offloaded to one MEC server .
4. *Binary Offloading- Complete MCC (BO-CC)*: the task is allowed to be fully offloaded to the cloud.
5. *Partial Offloading- Local and D2D offloading (PO-LD)*: the task is allowed to be partially executed locally and remotely using D2D offloading to one peer MT (PO-LD1) (system model adopted from [37]), or to multiple peer MTs (PO-LDM), simultaneously (system model adopted from [8]).
6. *Partial Offloading- Local, D2D and MCC offloading*: the task is allowed to be partially executed locally and remotely using D2D offloading to one peer MT, and MCC offloading, simultaneously (PO-LDC) (system model adopted from [12,13]).

7. *Partial Offloading- Local, D2D and MEC offloading mode selection (PO-MSDE)*: the task is allowed to be partially executed based on 5 modes: (1) local execution; D2D offloading to one peer MT including (2) complete D2D offloading, and (3) partial D2D offloading with local execution; and MEC offloading including (4) complete MEC offloading, and (5) partial MEC offloading with local execution (system model adopted from [14]).
8. *Partial Offloading- Local, D2D, MEC and MCC offloading mode selection (PO-MSDEC)*: we customized PO-MSDE adopted in [14] to consider two additional modes: (6) complete MCC offloading, and (7) partial MCC offloading with local execution.
9. *Partial offloading- Random Sequential offloading (PO-RSO)*: the task is allowed to be partially executed locally, or offloaded to peer MTs, MECs and MCC. The MTR selects randomly the nodes to offload its task data to and keeps assigning data sequentially to nodes until its data is completed.
10. *Partial offloading- IA- Maximum Offloading (IA-MO)*: we customized our IA approach and assigned data to the node providing maximum data execution first, considering the transmission and computation available capacity.

7.2. Simulation setup

As a case study, we consider A MTRs requesting computation task offloading and J peer MTs, which are randomly distributed in an area of 40 m × 40 m, where different wireless technologies exist. Our network is composed of four MECs and one MCC. The main system parameters are detailed below.

7.2.1. Computation demands

In our considered scenarios, we assume that a requester MTR i has a computation task U_i with size D_i varying uniformly from 1 kbit to 4 Mbits and computation requirement of $F_i = 1000$ CPU cycles per bit, with different delay tolerance T^{max}[13,14].

7.2.2. Computation resources

We assume that a MTR has limited local computation capacity F_i^l of 1 GHz [13]. The peer MTs contributing their computation resources are assumed to have higher capacity F_j^d of 2 GHz that can be assigned to serve multiple MTRs. The effective switched capacitance of a MT is assumed to be $\mathfrak{C}_i^l = \mathfrak{C}_j^d = 2 \cdot 10^{-26}$ reflecting the energy consumption coefficient related to the MT CPU performance [21]. The computation capacity of the MEC servers F_n^e and cloud F^c are assumed to be 10 GHz and 1 THz, respectively [38]. The MT and edge server resources are divided into $C^d = C^e = 10$ chunks of size S^c that is equal to $0.1 \cdot F_j^d$ and $0.1 \cdot F_n^e$, respectively. The computation cost may vary based on the server provider, capacity and performance. For instance, Hyve offers cloud services with 1 GB RAM and 4 × 3.0 GHz CPUs instance for 171 USD per month [39]. We assume a cost of 2 and 10 USD/month/terminal for 2 GHz and 10 GHz for a peer MT and MEC services, respectively [38]. Based on the previous example fees, the cost of the D2D, MEC, and MCC processing is computed to be $\phi_{i,j}^{dc} = 0.3858 \cdot 10^{-12}$ USD/Hz, $\phi_{i,n}^{ec} = 0.7716 \cdot 10^{-12}$ USD/Hz and $\phi_i^{cc} = 0.5466 \cdot 10^{-11}$ USD/Hz, respectively.

7.2.3. Radio resources

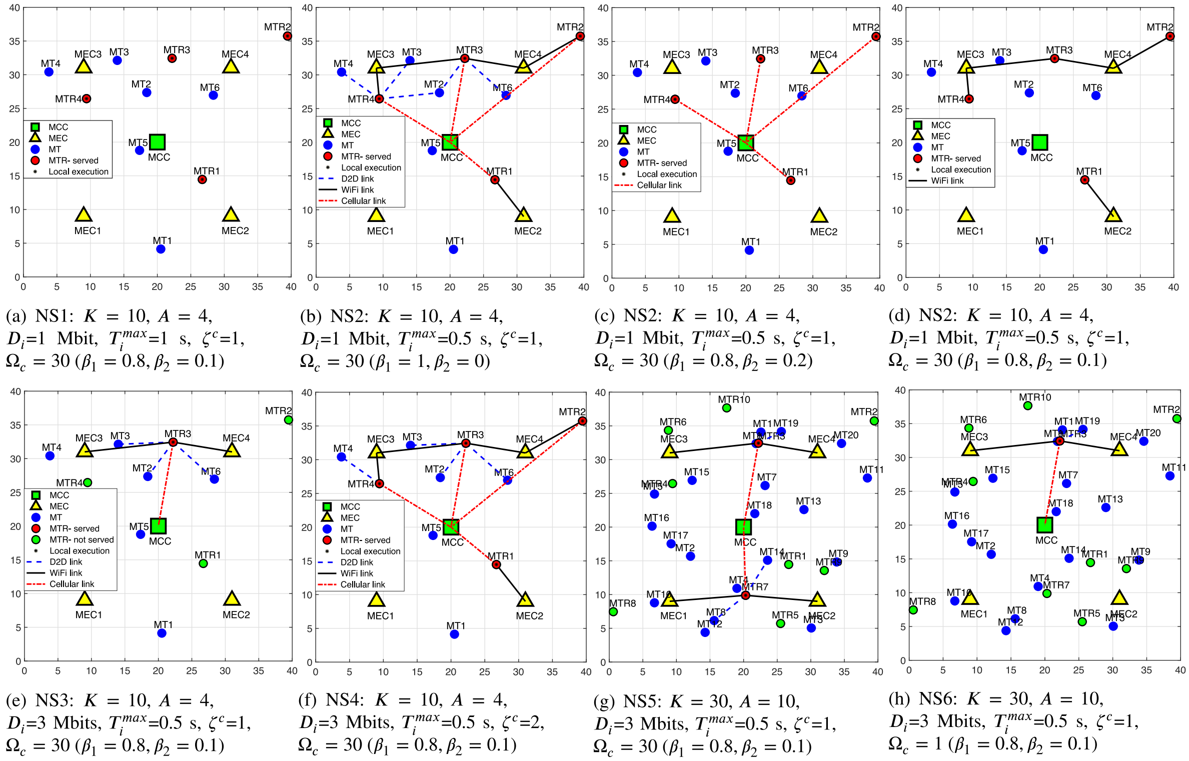
The coverage range of Bluetooth, WiFi, and cellular networks are set to 10, 15, and 50 m, respectively. The radio channel parameters α, κ, σ^2 and d_0 are set to 3.76, 127 dB, −75 dBm and 10 m, respectively, with 8-QAM modulation. The bandwidth of Bluetooth, WiFi channels and cellular subchannels are assumed to be 2, 5 and 10 MHz, respectively [14,28,31]. We assume the power P^{dt}, P^{et} and P^{ct} consumed by the MTR to transmit over Bluetooth, WiFi and cellular networks to be 0.5, 0.5, and 0.6 Watts, and the receive power P^{dr} over Bluetooth to be 0.2 Watts [40,41]. We limit the number of communication links as follows: a MTR i can offload its computation data to up to $\zeta^d = 7$ peer MTs over Bluetooth, $\zeta^e = 5$ MEC servers over WiFi, and to the cloud over the cellular network using multiple subchannels ζ^c, simultaneously. A peer MT j can serve up to $\Omega^d = 7$ MTRs over Bluetooth, simultaneously. Similarly, a MEC n and MCC can serve up to $\Omega^e = 15$ and $\Omega^c = 30$ MTRs over WiFi, and cellular networks, respectively [28,31]. We assume the transmission cost to be free over Bluetooth, 2×10^{-9} and 11×10^{-9} USD/kByte over WiFi and cellular networks, respectively [42,43].

7.3. Simulations results and analysis

7.3.1. Multi-objective optimal allocation performance evaluation

In this section, we illustrate solutions for the multi-objective optimal task offloading considering different network scenarios (NS) in Fig. 4. The performance evaluation in terms of number of MTRs served, energy consumption, monetary cost and execution time are detailed in Table 2. In the first four scenarios, we considered a network composed of 10 mobile terminals out of which 4 are MTRs and 6 are peer MTs, 4 MEC and a MCC, while varying system parameters such as data size D_i, delay requirement T_i^{max}, and number of allowed subchannels ζ^c. Considering the first NS, the task size D_i is assumed to be 1 Mbit, T_i^{max} is set to 1 s, and ζ_i^c to one, allowing only one cellular subchannel to every MTR. The optimal solution in Fig. 4(a) shows that all the users are served locally, without any data offloading. In the second scenario, T_i^{max} is reduced to 0.5 s, which forces the MTRs to offload parts of their computation task to other MTs, MEC and MCC to accomplish the task within the maximum allowed time. We illustrate three solutions for NS2 reflecting different objectives using different weights in (24) as follows: (1) maximizing the number of tasks only ($\beta_1 = 1, \beta_2 = 0$) in Fig. 4(b), (2) maximizing the number of completed tasks while minimizing energy consumption ($\beta_1 = 0.8$, $\beta_2 = 0.2$) in Fig. 4(c), (3) maximizing the number of completed tasks while minimizing energy consumption and monetary cost ($\beta_1 = 0.8$, $\beta_2 = 0.1$) in Fig. 4(d). The optimal allocation in Fig. 4(b) aiming at maximizing the number of completed tasks only is able to serve the 4 MTRs while consuming, in 0.5 s, a total of 2.12 Joules and $1.34 \cdot 10^{-4}$ USD in one time slot. Considering minimizing energy consumption in addition to maximizing the number of completed tasks ($\beta_1 = 0.8$, $\beta_2 = 0.2$), the MTRs tend to use local execution and MCC offloading as shown in Fig. 4(c). This is due to the fact that the local execution is less energy consuming and the cellular transmission rate is high, which reduces the transmission time between the MTR and the MCC and the MTRs energy consumption. In addition, the MCC processing time is very low due to its high computation capabilities. The solution is able to serve the 4 MTRs with a total energy of 0.53 Joules and $1.1 \cdot 10^{-3}$ USD per one timeslot, which represents a reduction of 74% in energy consumption, and 50% in monetary cost when compared to optimizing the number of completed tasks only. When aiming at maximizing the number of completed tasks while minimizing energy consumption and monetary cost, the MTRs tend to use the links with lower processing and transmission cost, accordingly tend to use local processing and MEC offloading as shown in Fig. 4(d). Compared to the solution of minimizing the energy consumption in Fig. 4(c), the multi-objective solution in Fig. 4(d) provides 98.27% reduction in monetary cost while consuming 37.57% more energy. The multi-objective OA is able to provide 59.52% and 99.1% reduction in energy consumption and monetary cost, respectively, compared to maximizing the number of completed tasks only. Increasing D_i to 3 Mbits in NS3 reduced the number of MTRs served due to time limitation. As presented in Fig. 4(e), the computation data tasks of MTRs 1, 2, and 4 cannot be transmitted and processed within a time limit of 0.5 s; only MTR 3 is able to complete its task by using D2D, MEC and MCC offloading. Allowing the allocation of two subchannels per MTR ($\zeta_i^c = 2$) in NS4 increases the cellular transmission rate, hence, allowing the completion of all the tasks within the time limit as presented in Fig. 4(f).

In Figs. 4(g) and (h), we presented solutions considering a larger network composed of 30 mobile terminals out of which 10 are MTRs requesting tasks with $D_i = 3$ Mbits and $T_i^{max} = 0.5$ s. We assume the



(a) NS1: $K = 10$, $A = 4$, D_i=1 Mbit, T_i^{max}=1 s, ζ^c=1, $\Omega_c = 30$ ($\beta_1 = 0.8, \beta_2 = 0.1$)

(b) NS2: $K = 10$, $A = 4$, D_i=1 Mbit, T_i^{max}=0.5 s, ζ^c=1, $\Omega_c = 30$ ($\beta_1 = 1, \beta_2 = 0$)

(c) NS2: $K = 10$, $A = 4$, D_i=1 Mbit, T_i^{max}=0.5 s, ζ^c=1, $\Omega_c = 30$ ($\beta_1 = 0.8, \beta_2 = 0.2$)

(d) NS2: $K = 10$, $A = 4$, D_i=1 Mbit, T_i^{max}=0.5 s, ζ^c=1, $\Omega_c = 30$ ($\beta_1 = 0.8, \beta_2 = 0.1$)

(e) NS3: $K = 10$, $A = 4$, D_i=3 Mbits, T_i^{max}=0.5 s, ζ^c=1, $\Omega_c = 30$ ($\beta_1 = 0.8, \beta_2 = 0.1$)

(f) NS4: $K = 10$, $A = 4$, D_i=3 Mbits, T_i^{max}=0.5 s, ζ^c=2, $\Omega_c = 30$ ($\beta_1 = 0.8, \beta_2 = 0.1$)

(g) NS5: $K = 30$, $A = 10$, D_i=3 Mbits, T_i^{max}=0.5 s, ζ^c=1, $\Omega_c = 30$ ($\beta_1 = 0.8, \beta_2 = 0.1$)

(h) NS6: $K = 30$, $A = 10$, D_i=3 Mbits, T_i^{max}=0.5 s, ζ^c=1, $\Omega_c = 1$ ($\beta_1 = 0.8, \beta_2 = 0.1$)

Fig. 4. Multi-objective optimal computation offloading solutions for different network scenarios.

Table 2
Performance evaluation of the multi-objective optimal allocation (OA), sub-optimal HA and IA approaches.

	Number of MTRs served			Energy (Joules per served MTR)			Cost (USD per served MTR)			Execution (seconds)		
NS	OA	HA	IA	OA	HA	IA	OA	HA	IA	OA	HA	IA
NS1	4	4	4	$8 \cdot 10^{-20}$	$8 \cdot 10^{-20}$	$8 \cdot 10^{-20}$	0	0	0	0.72	0.2	0.0458
NS2	4	4	4	0.2147	0.2150	0.2150	$4.75 \cdot 10^{-6}$	$4.56 \cdot 10^{-6}$	$4.56 \cdot 10^{-6}$	30 min	12.45	0.0511
NS3	1	1	1	1.5403	1.5412	1.5412	$5.23 \cdot 10^{-4}$	$5.21 \cdot 10^{-4}$	$5.21 \cdot 10^{-4}$	0.93	0.31	0.0501
NS4	4	4	3	0.7707	0.7707	0.7470	$9.45 \cdot 10^{-4}$	$9.46 \cdot 10^{-4}$	$9.87 \cdot 10^{-4}$	106 min	81.4 min	0.0465
NS5	2	2	2	1.5147	1.5185	1.5185	$5 \cdot 10^{-4}$	$5.01 \cdot 10^{-4}$	$5.01 \cdot 10^{-4}$	76.3 min	49.69	0.0739
NS6	1	1	1	1.5064	1.5269	1.5101	$5 \cdot 10^{-4}$	$5.1 \cdot 10^{-4}$	$5 \cdot 10^{-4}$	1.02	0.49	0.0635

MCC can connect to up to $\Omega_c = 30$ MTRs in NS5 represented in Fig. 4(g), while we limit the number of connections Ω_c to one MTR in NS6 represented in Fig. 4(h). Increasing the number of nodes from NS3 to NS5 affected the execution time as presented in Table 2 which increased from 1 s to 76 min (on a 3.4 GHz Core i7). Limiting the number of MTRs served to one in NS6, MTR3 providing simultaneously the minimum energy and cost was selected to be served by the multi-objective OA approach in Fig. 4(g).

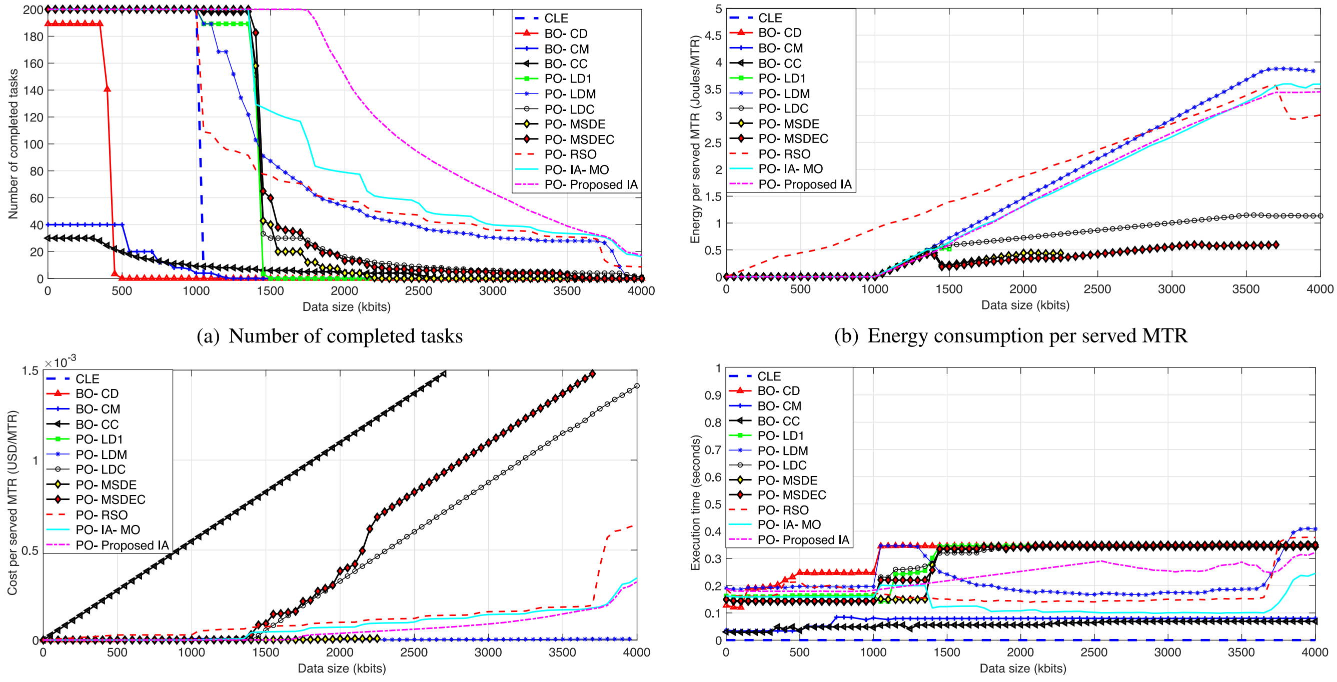
7.3.2. Hierarchical allocation performance evaluation

To compare the performance of the proposed hierarchical allocation with the multi-objective optimal allocation, we generated sub-optimal solutions using HA for the network scenarios considered in Fig. 4. As presented in Table 2, the HA and OA approaches were able to serve the same number of completed tasks in all the considered scenarios with sub-optimal performance in terms of monetary cost and energy consumption, with less time complexity. This is due to the fact that, the multi-objective OA aims at maximizing the number of completed tasks which is guaranteed in the first optimal sub-problem of the proposed HA approach. For instance, all the MTRs were served in NS2 using HA with 4% less monetary cost and 14% more energy consumption per served MTR. However, the execution time was reduced from 30 min to 12.45 s, compared to the OA approach.

The number of served MTRs determined by the solution of the first sub-problem in the HA approach plays also a major role in reducing its time complexity. For instance, the number of MTRs served in NS5 was reduced to only two out of 10 MTRs, which has notable impact on the complexity of solving the second HA sub-problem. For this reason, the execution time was reduced from 76 min to 50 s when the OA and HA approaches were simulated, respectively. Furthermore, the selection of the served MTRs by the first sub-problem may lead to sub-optimal solutions. For instance, the number of cloud connections were reduced from 30 in NS5 (Fig. 4(g)) to one MTR in NS6 (Fig. 4(h)); i.e.: either MTR 3 or MTR 7 can be served. The multi-objective OA approach was able to select the MTR providing simultaneously minimum energy and monetary cost which is MTR 3 in this case. However, the sub-optimal HA approach selected MTR 7 instead of MTR 3 which lead to 1.36% and 2% more energy and monetary cost compared to the multi-objective OA approach.

7.3.3. Iterative allocation performance evaluation

As presented in Table 2, the proposed IA approach was able to provide sub-optimal performance in terms of number of completed tasks, energy consumption and monetary cost with very low time complexity. For instance, considering NS4, the execution time was reduced from 106 and 81.4 min to 0.0465 s, compared to the OA and HA approaches, respectively. However, the IA approach was able to serve 3 MTRs out of 4. This is due to the fact that IA is affected by the order for serving the MTRs. As presented in Fig. 3 and detailed in Algorithm 1, we give priority to serve first MTRs with lower deadline and number



(a) Number of completed tasks

(b) Energy consumption per served MTR

(c) Monetary cost per served MTR

(d) Execution time

Fig. 5. Performance evaluation using different system models and offloading approaches while varying the computation data size.

of connections. Moreover, we aimed at reserving the cloud resources for serving more MTRs by favoring completion of tasks without MCC offloading. That is, if the task cannot be performed without the use of the cloud, we allocate the maximum allowed data to be executed locally, or offloaded to peer MTs, and MECs while fully utilizing their computation resources. Hence, the cooperating node will not be able to share their resources with more than one MTR. MTR 3 was then left without sufficient computation resources to complete its task in NS4. Similarly, in NS6, MTR 7 was served first which lead to sub-optimal performance in terms of energy consumption and monetary cost.

Moreover, we assess the performance of the proposed IA approach using different strategies and system models adopted in the open literature as presented in Section 7.1. We assume our network is composed of A= 200 MTRs and J= 200 MTs, 4 MECs and MCC. We assume ζ^c= 1 and T^{max}= 1 s. We generated results for 100 runs with different MTRs distributions, and evaluated the average of the following performance metrics: number of completed tasks in Fig. 5(a), energy consumption by served MTR in Fig. 5(b), monetary cost per served MTR in Fig. 5(c), and execution time in Fig. 5(d), while varying the computation task data size.

As presented in Fig. 5, complete local execution without using task offloading is energy and cost efficient. However, the computation capacity of MTRs is limited which restricts the MTRs to complete the tasks with large data sizes locally. Using fully their local computation capacity, the MTRs can process up to 1 Mbit within T^{max} = 1 s. This drops the number of completed tasks from 200 to 0 when the data size exceeds 1 Mbit. Binary offloading approaches without considering local execution provided limited number of completed tasks. This is due to the fact that the radio resources of the MEC and MCC are limited to 15 and 30 in our considered scenario. The BO-complete D2D was able to provide better performance since the number of cooperating devices for D2D offloading is 200 MTs, which allows more MTRs to communicate through Bluetooth short range connectivity and complete their tasks, compared to BO-CM and BO-CC.

Using local execution with computation data offloading including D2D, MEC and MCC offloading provided better performance in terms of number of completed tasks compared to CLE and BO models with a tradeoff cost in terms of energy consumption and monetary cost. The models PO-LD1, PO-MSDE and PO-MSDEC allowed the MTR to divide its task into two parts, one executed locally and the other is offloaded to be executed remotely. The performance of these approaches degraded with the increase of the data size. For instance, when the data size reached 1.5 Mbits, PO-LD1 showed zero completed tasks since the computation demands exceed the computation capability of the peer MTs. PO-MSDE and PO-MSDEC outperform PO-LD1 due to the fact that they allow the MTRs to select the best mode for computation data offloading including D2D offloading to one peer MT, and edge cloud offloading. The number of completed tasks using PO-MSDEC was higher than PO-MSDE due to the fact that PO-MSDEC allowed 7 selection modes including D2D, MEC and MCC offloading. PO-MSDEC was able to complete tasks up to 3.75 Mbits compared to 2.3 Mbits using PO-MSDE. PO-MSDEC consumed slightly less energy consumption per served MTR since the MCC provide high transmission rates, which reduces the time of transmission. The cost charged by MCC is higher compared to D2D and MEC offloading used in PO-MSDE which shows higher monetary cost. PO-LDC allowed the MTR to divide its task into three parts to be locally executed, offloaded to one peer MT and MCC, which allowed more tasks to be completed when the data size is large, namely up to 3.9 Mbits. Dividing the task into up to 8 parts to be locally executed or offloaded to 7 different peer MTs, simultaneously, allowed PO-LDM without MEC and MCC offloading to complete a larger number of computation tasks even with high data size of up to 3.9 Mbits where the computation capacity of the cooperating MTs was not able to serve any MTRs within the task deadline. PO-LDM cost is very low in terms of monetary cost, however, it is the highest in terms of energy consumption since we consider in our model the transmission, reception and computation energy consumption at the peer MTs and MTRs. Using PO-RSO provided higher number of served MTRs compared to conventional approaches since it allows the use all the possible connections to complete a task. However, the cooperating node is randomly selected which does not provide any guarantee on reducing the energy consumption and monetary cost. PO-RSO fully reserves all the available computation resources of the selected node, which prevents it to serve other MTRs and does not give any priority

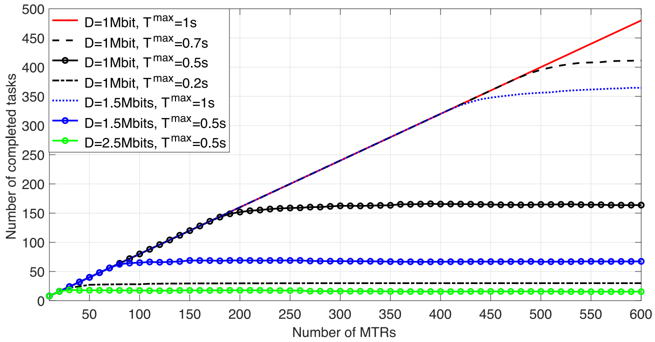


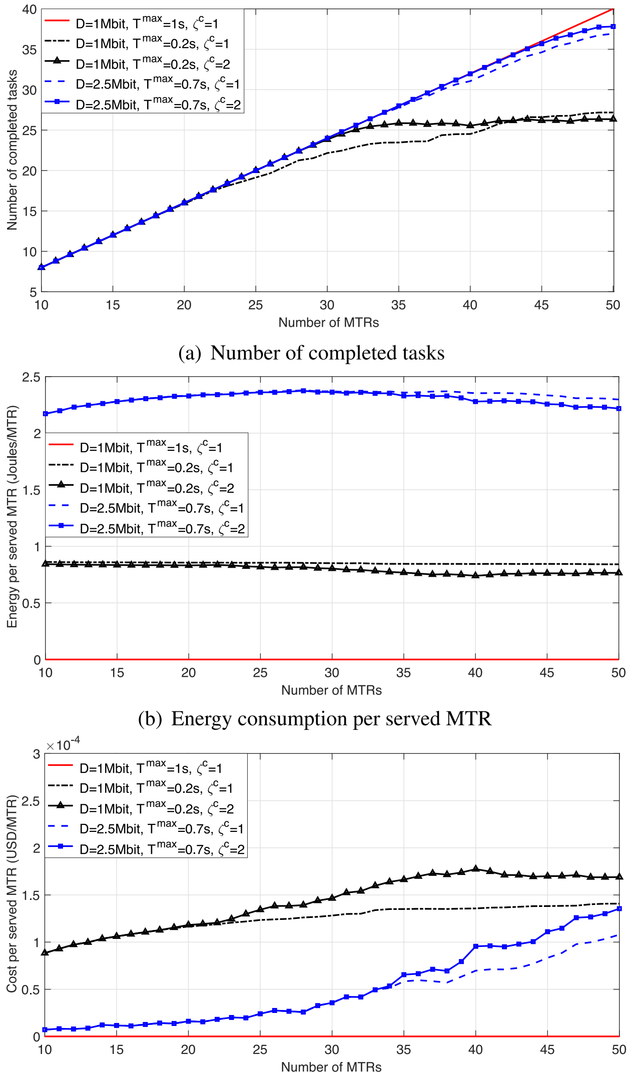
Fig. 6. Proposed IA approach performance evaluation while varying the number of MTRs A, data size and T^{max}.

to MTRs, nor saves the cloud resources by favoring the use of short range connectivity. Similarly, considering IA-MO approach, the MTR offloads its data to the node providing the maximum data allowed to be executed while fully utilizing its available computation resources which leads to lower performance compared to the proposed IA approach.

Our proposed IA approach outperforms all other approaches in terms of completed tasks with a tradeoff cost in terms of energy consumption and monetary cost. Despite their simplicity and very low average execution time which was 0.0002, 0.0721 and 0.0573 s for CLE, BO-CM and BO-CC, respectively, the baseline and binary offloading approaches performance was very limited. Compared to PO-LDM, the proposed approach provided on average 37% more completed tasks while consuming 8.24% less energy, and 60.81% more monetary cost. The average execution time of the proposed IA approach was 0.2354 s (on a 3.4 GHz Core i7) which was 8.4% more than the execution time of PO-LDM. Compared to PO-LDC and PO-MSDEC, IA provided on average 57% and 59% more completed tasks while consuming 32.35% and 47% more energy consumption and 65.29% and 64.4% less monetary cost, respectively. Moreover, the execution time was reduced by more than 16.5% and 17.2%, respectively. This is due to the fact that IA allows data offload to a larger number of cooperating nodes, including peer MTs, MECs, and cloud. It also allows more flexible data assignment where the MTRs select the most efficient solution providing the best tradeoff between energy consumption and monetary cost. The number of completed task is large even with high computation demands where the MTRs use more wireless interfaces and energy to complete their tasks within the deadline requirement, while minimizing monetary cost.

7.3.4. Study on the parameters: A, D, T^{max}, ζ^c and β

In this section, we first evaluate the dynamic performance of the proposed IA approach by generating average results over 200 time slots, while varying system parameters such as the number of MTRs A, data size D, delay tolerance T^{max} and maximum number of cellular subchannels ζ^c assigned per MTR. Our network is composed of 200 peer MTs, 4 MECs and MCC, while considering up to 600 MTRs. We assume a requester generates multiple tasks with a level of activity λ_A=0.8, i.e.: an MTR is active 80% of the 200 time slots considered. We also assume the time slot duration T_s is equal to the maximum delay tolerance of the tasks, and the assigned computation resources are reserved and then released on a time slot basis. As presented in Fig. 6, the number of completed tasks decreases while increasing the computation data size or decreasing the delay tolerance. When D= 1 Mbit and T^{max}= 1 s, all the tasks were completed using local execution. Decreasing T^{max} from 1 to 0.7 and 0.2 s, reduced the number of completed tasks from 488 to 412 and 30, respectively. For the same deadline T^{max}= 1 s, increasing the computation data size from 1 Mbit to 1.5 Mbits and 2.5 Mbits decreased the number of completed tasks from 488 to 67 and 15, respectively. Strict deadlines and large data sizes restrict the number of completed tasks due to the limitation of radio and computation resources. MTRs need to use more wireless interfaces to meet the task deadline which becomes more and more scarce when the number of MTRs increases. For this reason, the number of completed tasks becomes uniform after reaching system maximum radio and computation capacity.



(a) Number of completed tasks

(b) Energy consumption per served MTR

(c) Monetary cost per served MTR

Fig. 7. Proposed IA approach performance evaluation while varying the number of MTRs A, data size D, delay tolerance T^{max} and number of subchannels ζ^c assigned per MTR.

In Fig. 7, we focus our performance analysis on a smaller number of MTRS (up to 50), where resources are not scarce and MTRs can select more efficient solutions. Increasing ζ^c from 1 to 2, was able to achieve higher number of completed tasks when D= 2.5 Mbits and T^{max}= 0.7 s. More MTRs can then use two cellular subchannels to complete their tasks. However, this increase is limited by the available number of cellular subchannels (Ω^c = 30), which affects the number of completed tasks when larger number of MTRs are considered. In the case where the MTRs are requesting computation demands of D= 1 Mbits with a strict deadline of T^{max}= 0.2 s, allocating two subchannels to MTRs with bad channel conditions restricted other MTRs with better conditions to be served. Therefore, due to the limited number of cellular subchannels, the performance of the IA approach degraded in terms of number of completed tasks. Moreover, allocating multiple subchannels per MTR

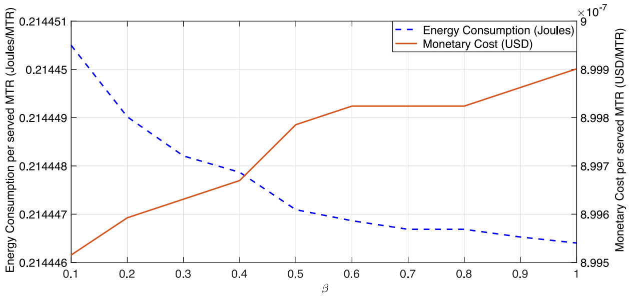


Fig. 8. Proposed IA approach performance evaluation while varying the weight of the energy consumption β.

increased the transmission rate, which decreased the energy consumption, and allowed more data to be offloaded to MCC with expensive charges, which increases the monetary cost as presented in Fig. 7(c).

To evaluate the tradeoff between energy consumption and monetary cost, we generated results for 100 runs with different MTRs distribution in Fig. 8 for a network composed of A= 20 MTRs, J= 200 MTs, 4 MECs and MCC, with D_i= 1.5 Mbits, and T^{max}= 1 s, while varying β the weight coefficient indicating the impact of minimizing the energy consumption in the utility function adopted in Algorithm 1. As presented in Fig. 8, increasing β reduced the energy consumption and increased the monetary cost.

7.4. Simulations results outcome

The proposed IA approach provides real-time task offloading decisions including computation and radio resource allocation aiming at minimizing the energy consumption and monetary cost. The latter is achieved by simultaneously utilizing the multiple wireless interfaces available at the MTRs and different offloading models including local execution, D2D, MEC and MCC offloading. Using partial data task offloading to multiple cooperating nodes allowed the proposed approach to outperform other alternative approaches in terms of maximizing the total number of completed tasks within the set deadline. Moreover, the results show that IA scales efficiently well for large real-time networks with hundreds of MTRs and high computation demands.

8. Conclusions

This paper addressed joint computing, communication and cost-aware task offloading aiming at maximizing the number of completed tasks while minimizing energy consumption and monetary cost in D2D-enabled heterogeneous MEC networks. The proposed solution adopts partial offloading where a requester offloads different parts of its computation data task simultaneously to multiple MTs, edge servers and cloud. We formulated the problem as a multi-objective optimization problem which is shown to be NP-hard. To reduce the time complexity, we propose hierarchical and iterative allocation approaches to provide fast sub-optimal solutions for real-time applications. We evaluated the proposed IA approach under different system parameters. The results show that the proposed approach outperforms existing approaches in providing a significant tradeoff between number of completed tasks, energy consumption, monetary cost and execution time while providing fast and efficient solutions for large real-time networks with up to 600 MTRs and high computation demands. As future work, the proposed approaches can be extended to consider subtask dependencies where the result of one subtask is input to a set of other subtasks.

CRediT authorship contribution statement

Nadine Abbas: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data curation, Writing – review & editing, Visualization. **Sanaa Sharafeddine:** Conceptualization, Methodology, Validation, Formal analysis, Writing – review & editing. **Azzam Mourad:** Conceptualization, Methodology, Formal analysis, Resources, Writing – review & editing. **Chadi Abou-Rjeily:** Conceptualization, Methodology, Formal analysis, Writing – review & editing. **Wissam Fawaz:** Conceptualization, Methodology, Formal analysis, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Cisco, Cisco visual networking index: Forecast and trends, 2017–2022, in: White Paper, Cisco, 2019.

[2] L. Bariah, et al., A prospective look: Key enabling technologies, applications and open research topics in 6G networks, IEEE Access 8 (2020) 174792–174820.

[3] L. Loven, T. Leppanen, E. Peltonen, J. Partala, E. Harjula, P. Porambage, M. Ylianttila, J. Riekki, EdgeAI: A vision for distributed, edge-native artificial intelligence in future 6G networks, 6G Wirel. Summit (2019).

[4] European Telecommunications Standards Institute, ETSI, Mobile edge computing: A key technology towards 5G, in: ETSI White Paper No.11, 2015.

[5] C. Jiang, X. Cheng, H. Gao, X. Zhou, J. Wan, Toward computation offloading in edge computing: A survey, IEEE Access 7 (2019) 131543–131558.

[6] H. Wu, Multi-objective decision-making for mobile cloud offloading: A survey, IEEE Access 6 (2018) 3962–3976.

[7] H. Xing, L. Liu, J. Xu, A. Nallanathan, Joint task assignment and resource allocation for D2D-enabled mobile-edge computing, IEEE Trans. Commun. 67 (6) (2019) 1750–1763.

[8] T. Dbouk, A. Mourad, H. Otrok, H. Tout, C. Talhi, A novel ad-hoc mobile edge cloud offering security services through intelligent resource-aware offloading, IEEE Trans. Netw. Serv. Manag. 16 (4) (2019) 1665–1680.

[9] D. Wang, H. Qin, B. Song, X. Du, M. Guizani, Resource allocation in information-centric wireless networking with D2D-enabled MEC: A deep reinforcement learning approach, IEEE Access 7 (2019) 114935–114944.

[10] G. Li, J. Cai, An online incentive mechanism for collaborative task offloading in mobile edge computing, IEEE Trans. Wirel. Commun. 19 (1) (2020) 624–636.

[11] U. Saleem, Y. Liu, S. Jangsher, X. Tao, Y. Li, Latency minimization for D2D-enabled partial computation offloading in mobile edge computing, IEEE Trans. Veh. Technol. 69 (4) (2020) 4472–4486.

[12] Y. He, J. Ren, G. Yu, Y. Cai, Joint computation offloading and resource allocation in D2D enabled MEC networks, in: Proceedings of the IEEE International Conference on Communications, 2019.

[13] Y. He, J. Ren, G. Yu, Y. Cai, D2D communications meet mobile edge computing for enhanced computation capacity in cellular networks, IEEE Trans. Wirel. Commun. 18 (3) (2019) 1750–1763.

[14] R. Chai, J. Lin, M. Chen, Q. Chen, Task execution cost minimization-based joint computation offloading and resource allocation for cellular D2D MEC systems, IEEE Syst. J. 13 (4) (2019) 4110–4121.

[15] Z. Wan, D. Xu, D. Xu, I. Ahmad, Joint computation offloading and resource allocation for NOMA-based multi-access mobile edge computing systems, Comput. Netw. 196 (2021).

[16] L. Qian, Y. Wu, J. Ouyang, Z. Shi, B. Lin, W. Jia, Latency optimization for cellular assisted mobile edge computing via non-orthogonal multiple access, IEEE Trans. Veh. Technol. 69 (5) (2020) 5494–5507.

[17] D. Xu, Q. Li, H. Zhu, Energy-saving computation offloading by joint data compression and resource allocation for mobile-edge computing, IEEE Commun. Lett. 23 (4) (2019) 704–707.

[18] Y. Lan, X. Wang, D. Wang, Y. Zhang, W. Wang, Mobile-edge computation offloading and resource allocation in heterogeneous wireless networks, in: Proceedings of the IEEE Wireless Communications and Networking Conference, 2019.

[19] K. Wang, et al., Joint offloading and charge cost minimization in mobile edge computing, IEEE Open J. Commun. Soc. 1 (2020) 205–216.

[20] Z. Zhao, W. Zhou, D. Deng, J. Xia, L. Fan, Intelligent mobile edge computing with pricing in internet of things, IEEE Access 8 (2020) 37727–37735.

[21] Z. Zhao, et al., On the design of computation offloading in fog radio access networks, IEEE Trans. Veh. Technol. 68 (7) (2019) 7136–7149.

[22] H.A. Alameddine, S. Sharafeddine, S. Sebbah, S. Ayoubi, C. Assi, Dynamic task offloading and scheduling for low-latency IoT services in multi-access edge computing, IEEE J. Sel. Areas Commun. 37 (3) (2019) 668–682.
[23] N. Kherraf, S. Sharafeddine, C.M. Assi, A. Ghrayeb, Latency and reliability-aware workload assignment in IoT networks with mobile edge clouds, IEEE Trans. Netw. Serv. Manag. 16 (4) (2019) 1435–1449.
[24] S. Arisdakessian, O.A. Wahab, A. Mourad, H. Otrok, N. Kara, FoGMatch: An intelligent multi-criteria IoT-fog scheduling approach using game theory, IEEE/ACM Trans. Netw. 28 (4) (2020) 1779–1789.
[25] P. Wang, Z. Zheng, B. Di, L. Song, HetMEC: Latency-optimal task assignment and resource allocation for heterogeneous multi-layer mobile edge computing, IEEE Trans. Wirel. Commun. 18 (10) (2019) 4942–4956.
[26] H. Wang, Z. Peng, Y. Pei, Offloading schemes in mobile edge computing with an assisted mechanism, IEEE Access 8 (2020) 50721–50732.
[27] A. Goldsmith, Wireless Communications, Cambridge University Press, 2005.
[28] C. Gomez, J. Oller, J. Paradell, Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology, Sensors 12 (9) (2012) 11734–11753.
[29] Bluetooth Specifications, Bluetooth core specification v 5.0, in: Bluetooth SIG Proprietary, 2016.
[30] IEEE Std 802.11-2007 (Revision of 802.11-1999), IEEE Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, 2007.
[31] Cisco, Channel Planning Best Practices, White Paper, 2016.
[32] J. Bisschop, AIMMS Optimization Modeling, AIMMS, 2020.
[33] O.E. Kundakcioglu, S. Alizamir, Generalized assignment problem, in: Encyclopedia of Optimization, Springer US, Boston, MA, 2008.
[34] M. Assi, R.A. Haraty, A survey of the knapsack problem, in: Proceedings of the International Arab Conference on Information Technology, 2018.
[35] S.O. Krumke, C. Thielen, The generalized assignment problem with minimum quantities, Eur. J. Oper. Res. 228 (1) (2013) 46–55.
[36] C. D'Ambrosio, S. Martello, M. Monaci, Lower and upper bounds for the non-linear generalized assignment problem, Comput. Oper. Res. 120 (2020) 104933.
[37] C. You, K. Huang, Exploiting non-causal CPU-state information for energy-efficient mobile cooperative computing, IEEE Trans. Wirel. Commun. 17 (6) (2018) 4104–4117.
[38] L. Huang, X. Feng, L. Zhang, L. Qian, Y. Wu, Multi-server multi-user multi-task computation offloading for mobile edge computing networks, Sensors 19 (6) (2019) 1446.
[39] Cloudorado - Cloud Computing Comparison Engine, Cloud server comparison - price & features, 2020, [online] Available at: https://www.cloudorado.com/cloud_server_comparison.jsp.
[40] S. Sharafeddine, K. Jahed, N. Abbas, E. Yaacoub, Z. Dawy, Exploiting multiple wireless interfaces in smartphones for traffic offloading, in: Proceedings of the First International Black Sea Conference on Communications and Networking, July 2013.
[41] K. Jahed, M. Younes, S. Sharafeddine, Energy measurements for mobile cooperative video streaming, in: Proceedings of the IFIP Wireless Days, 2012.
[42] S. Sen, C. Joe-Wong, S. Ha, M. Chiang, Smart data pricing (SDP): Economic solutions to network congestion, in: H. Haddadia, O. Bonaventure (Eds.), Recent Advances in Networking, 2013, pp. 221–274.
[43] N. Abbas, H.H. S. Sharafeddine, Z. Dawy, Price-aware traffic splitting in D2D HetNets with cost-energy-QoE tradeoffs, Comput. Netw. 172 (2020) 107169.

Nadine Abbas is a Visiting Assistant Professor at the Computer Science department at the Lebanese American University (LAU). Nadine earned a Bachelor of Engineering degree in Computer and Communication Engineering from Notre Dame University in 2008. She received her M.E. and Ph.D. degrees in Electrical and Computer Engineering from the American University of Beirut (AUB) in 2011 and 2017, respectively. Her research interests include mobile edge computing and wireless communications, mainly heterogeneous networks that support multi-radio networking.

Sanaa Sharafeddine is a Professor of Computer Science at the Lebanese American University. She received the B.E. and the M.E. degrees in Computer and Communications Engineering from the American University of Beirut in 1999 and 2001, respectively. She received her doctoral degree in Communications Engineering from Munich University of Technology (TUM) in 2005 in collaboration with Siemens AG research labs in Munich. Her research interests are in the general area of wireless networks with focus on UAV-aided communications, edge computing, D2D cooperation, and multimedia services. She joined the editorial boards of IEEE Networking Letters, Elsevier Ad Hoc Networks and IEEE Access. She received the International Rising Talent Award in 2015, L'Oreal-UNESCO Pan-Arab Regional Fellowship Award in 2013, and elevated to a senior member of the IEEE in 2010.

Azzam Mourad received his M.Sc. in CS from Laval University, Canada (2003) and Ph.D. in ECE from Concordia University, Canada (2008). He is currently a Professor of Computer Science with the Lebanese American University, a Visiting Professor of Computer Science with New York University Abu Dhabi and an Affiliate Professor with the Software Engineering and IT Department, Ecole de Technologie Superieure (ETS), Montreal, Canada. He has served/serves as an associate editor for IEEE TNSM, IEEE Network, IEEE OJCS, IET Quantum Com., and IEEE Com. Letters, the General Chair of IWCMC2020, the General Co-Chair of WiMob2016, and the Track Chair, a TPC member, and a reviewer for several prestigious journals and conferences. He is an IEEE senior member.

Chadi Abou-Rjeily (M'07, SM'13) is a Professor at the department of Electrical and Computer Engineering of the Lebanese American University. He received his BE degree in electrical engineering in 2002 from the Lebanese University. He received his MS and Ph.D. degrees in electrical engineering in 2003 and 2006, respectively, from the École Nationale Supérieure des Télécommunications (ENST), Paris, France. From September 2003 to February 2007, he was also a research fellow at the Laboratory of Electronics and Information Technology of the French Atomic Energy Commission (CEA-LETI). His research interests are in the code construction and transceiver design for wireless communication systems.

Wissam Fawaz is a Professor at the department of Electrical and Computer Engineering of the Lebanese American University (LAU), Byblos, Lebanon. He received his Ph.D. in Network and Information Technology from the University of Paris XIII in 2005. He is a senior member of the IEEE and currently serves as associate editor for the Springer Journal of Annals of Telecommunications. He served as associate editor for the IEEE Communications Letters from 2013 until 2017. His research interests are in the areas of delay tolerant as well as quality of service enabled Vehicular Ad-Hoc Networks and Buffer-aided Free Space Optical Communication Systems. He received a Fulbright research award in 2008.