Wireless and Service Allocation for Mobile Computation Offloading with Task Deadlines

Hong Chen*, Terence D. Todd*, Dongmei Zhao* and George Karakostas†
*Department of Electrical and Computer Engineering
†Department of Computing and Software
McMaster University
Hamilton, Ontario, CANADA
Email: {chenh151,todd,dzhao,karakos}@mcmaster.ca

***Abstract*—In mobile computation offloading (MCO), mobile devices (MDs) can choose to either execute tasks locally or have them executed on a remote edge server (ES). This paper addresses the problem of assigning the wireless communication bandwidth and the ES capacity used for the task execution, so that task completion time constraints are satisfied. The objective is to minimize the average power consumption of the mobile devices is minimized, subject to a cost budget constraint for obtaining the communication and computation resources. The paper includes contributions for both soft and hard task completion deadline constraints. The problems are first formulated as mixed integer nonlinear programs (MINLPs). Approximate solutions are then obtained by decomposing the problems into a collection of convex subproblems that can be efficiently solved. Results are presented that demonstrate the quality of the proposed solutions, which can achieve near optimum performance over a wide range of system parameters.**

***Index Terms*—Edge computing, mobile computation offloading, soft and hard task completion deadlines, cost budget constraints, power efficiency.**

I. Introduction

Mobile computation offloading (MCO) can be used to improve mobile device (MD) performance by running computational tasks on a remote cloud server rather than executing them locally [1]–[3]. Since the energy needed for task execution is incurred by the cloud server, a reduction in mobile device energy consumption can often be obtained [4]–[10]. During MCO, wireless communications is used by the MD to communicate with the cloud server. This interaction incurs MD energy use that would not otherwise exist if the task were executed at the MD. MCO also incurs added latency due to the time needed for the MD to interact with the cloud server [11], [12]. An *edge server* (ES) located close to the network base stations (BSs) is typically used to reduce this delay by providing high interconnection bandwidth between the BS and the ES [13].

The question of whether a given task should be offloaded has been studied extensively [14]–[23]. It is clear from this work that in order to obtain good performance, the offloading decisions should incorporate both the limited edge server computational capacity [21]–[23] and the temporal evolution of the system during the computation offload. This includes the queueing behaviour experienced by offloaded tasks awaiting execution at the ES [18]–[20]. Prior work has also considered the question of how to configure system resources so that MCO is best accommodated [14], [18]–[20], [24], [25]. These are the issues that are considered in our paper and involve the tradeoffs between wireless communication and edge server capacity assignment and how these affect the delay performance experienced by the MDs.

The wireless and execution capacity assignment problem in MCO can be informally stated as follows. A network leaseholder (NL) purchases both wireless channel capacity and edge server execution services, subject to a cost budget constraint. The leased resources are then used to provide MCO to a large set of mobile devices [26]. When an MD generates a task for execution, there is an associated deadline, which gives the time by which task execution should be completed with a high degree of certainty [27]. The objective is to find a joint wireless and ES resource assignment that minimizes the mean MD power consumption subject to the budget constraint and constraints on the task completion times. Note that this problem is different than that of network slice creation [28]. In this case, the NL simply purchases services from the network owner (NO), who prices the cost of unit wireless channel and computational resources. Due to the edge server placement, we consider the case where the dominant latencies are that of wireless access and edge server execution [13].

The paper is novel in that it includes formulations for both *soft and hard task completion deadlines*. In the *hard deadline* case, the completion time deadline given for every task must *always* be satisfied, i.e., each task must be uploaded and executed by the time that its associated deadline expires. This is a very stringent requirement, which is accomplished by including concurrent local execution (CLE) [29] into the problem formulation. In CLE, local execution of the task may be initiated while offloading is ongoing, so that the task completion deadline is always met. In the *soft deadline* case, task completion times are permitted to violate their given deadlines, but the probability that this happens must be below a given violation threshold [27], [30]. Soft deadlines use a statistical deadline constraint and depending on the chosen threshold, the deadline violation probability can be set as permissively as desired. In the remainder of the paper, we define multiple task classes, with class-specific deadlines and constraint violation probabilities. Note that one can view the hard deadline case as a special case of soft deadlines, where

the deadline violation factor is set to zero, meaning that no task deadlines can be violated. However, the hard deadline formulation requires CLE, which is not needed in the soft deadline case. As a result, the solution methods in the soft and hard deadline cases are different.

The inclusion of task deadline constraints significantly increases the difficulty of the problem compared to that of prior work with no completion time requirements or that uses a mean delay criterion [31], [32]. In order to obtain solutions to the problem, a queuing model is used to obtain the delay distribution experienced by tasks that are offloaded to the ES [32], [33]. This model is incorporated into the resulting optimization problems, which are formulated as mixed integer nonlinear programming problems (MINLPs) that are computationally hard to solve exactly. Approximate solutions are obtained by decomposing the non-convex non-linear formulation into a collection of convex subproblems that can be solved efficiently, and then picking the best of these solutions.

A variety of results are presented that characterize the tradeoffs between task deadline violation, average MD power consumption and the cost budget. Our results show the quality of the proposed solutions, which can achieve close-to-optimum performance for a wide range of system parameters. The results also show that with CLE, the proposed solution not only guarantees to respect *all* hard task completion deadlines, but does so with only slightly higher MD power consumption when compared to the soft task completion deadlines solution with a small deadline violation probability. On the other hand, we show that there is an apparent trade-off in the case of soft task completion deadlines between the average power consumption and the deadline violation probability. Namely, the average MD power consumption of our solution is significantly reduced when a higher deadline violation probability is tolerable.

The main contributions of the paper are summarized below.

- This paper addresses the problem of assigning computational and wireless channel resources for MCO, subject to task execution completion time deadlines. The work is the first that generates joint resource assignments for both soft and hard task deadlines using very general system modelling assumptions compared to prior work. The soft deadline case aims to create assignments so that the probability of task completion time deadline violation does not exceed a given violation threshold. In the hard deadline case, the paper is also unique in that it creates resource assignments where task completion time deadlines are always satisfied. This is done by incorporating CLE into the problem formulations. For this reason, this is the first paper that obtains system resource assignments for MCO that ensure that task completion time deadlines are always satisfied.
- Modeling both soft and hard job completion time targets significantly increases the difficulty of the problem compared to prior work with no completion time requirements or that uses a mean delay criterion [30] [31]. In both deadline cases, the paper addresses this by incorporating an ES queueing system into the problem formulation that models the delay distribution experienced by arriving tasks. The assignment problem is addressed by numerically inverting the estimated probability generating function of task completion time and incorporating the resulting probability density function (PDF) into the optimizations. These resource assignments are obtained under very general modeling assumptions, where the wireless channels are modeled as arbitrary BS specific sets of Markov processes and task execution times have a general probability distribution.
- The problems are first formulated as MINLPs, with *integral* decision variables for the number of wireless channels reserved, and a *continuous* decision variable for the portion of ES reserved. Even the relaxations of these MINLPs are difficult to solve, since they are non-convex. Hence, instead of following the common practice of solving the relaxation and rounding the fractional solution, we break the original non-convex MINLPs into collections of *convex* subproblems, that can be solved efficiently. This is achieved by the discretization of the *continuous* variable and the replacement of the *discrete* channel variables by approximate functions of the *continuous* blocking probabilities. Our solutions are approximate, and their accuracy depends on both the discretization granularity and the approximation functions used for blocking probabilities. On the other hand, they are based on very general assumptions, i.e., the existence of convex upper bound approximations of the inversion of blocking probabilities. The more restricted the system model is, the better these approximations are.

The remainder of the paper is organized as follows. In Section II the prior work most related to our paper is reviewed. The system model and problem formulation is then described in Section III. In Section III-A, the general design problem is first considered assuming soft task completion time deadlines, where the probability of deadline violation is bounded. Following this, in Section III-B a formulation is described when task completion times are subject to hard deadlines. The problem formulations in both cases are non-convex and difficult to deal with directly using conventional optimization approaches. In Section IV, approximation solutions are proposed where the original problems are decomposed into convex subproblems that can be efficiently solved. Both the soft and hard deadline cases are considered in Sections IV-A and IV-B. Section V then introduces some common system assumptions used in the remainder of the paper when solving the optimizations. Both the soft and hard deadline cases are then treated in detail in Sections V-A and V-B. In Section VI simulation results that demonstrate the proposed designs are given. Both the single class and multiple classes of tasks cases are considered in Sections VI-A and VI-B. Finally, we present our conclusions of the work in Section VII.

II. Related Work

A large amount of prior MCO work considers the problem based on system state inputs sampled at task generation times, i.e., the models assume that the system is static throughout the offload period [14], [15], [17]–[25], [35]–[37]. Instead, [30]–

TABLE I: Related Work Summary

References	Joint channel and computation resource assignment	Soft task deadlines	Hard task deadlines	Resource expense	Temporal evolution
[17] [21] [22] [23]			✓		
[24] [34]	✓				
[26]			✓	✓	
[31] [32]	✓				✓
[33]			✓		✓
[30]	✓	✓			✓
Our paper	✓	✓	✓	✓	✓

[33] considers that the wireless channels may change randomly during the offload.

When considering task offloading completion time, a latency minimization problem is investigated in [34], average delay of task completion is considered in [31], [32], a user satisfaction utility function is optimized in [37] based on the desired and actual task completion time and energy consumption. When tasks have hard delay constraints, the system may become infeasible [17], [21]–[23], [33], [35] and tasks can be dropped [33] when the required hard task completion time cannot be satisfied.

Instead of considering a flow of tasks with random arrival times, as in our paper, [30] makes offloading decisions for tasks in the current time slot, where task offloading with statistical quality of service (QoS) guarantees (i.e., tasks are allowed to complete before a given deadline with a probability above a given threshold) is considered.

Besides task completion time, reducing energy consumption is another common objective in mobile computation offloading. Examples of this include minimizing the total energy consumption of all MDs [22], [35], minimizing the energy consumption of the entire MCO system [21], or optimizing a utility function that is a weighted sum of task completion time and energy consumption [24], [25], [36].

Prior work has considered the optimization of communication and computation resources to improve MCO performance [14], [18]–[20], [24], [25]. The work in [17], [21]–[23], [26] focuses on the effect of radio resource allocations on offloading decisions. More specifically, task uploading decisions are jointly optimized with wireless channel assignments [17], [21], [23], [26], channel transmission time scheduling [22], [26], and MD transmission power allocations [17], [23], [26] for the task uploading. Comparing to binary offloading decisions, where an MD either offloads the entire task to the edge server or executes the task locally, partial offloading provides more flexibility in MCO [24], [37]–[41].

Table I summarizes the work described above that is most related to our paper, and compares it to this paper on five key properties:

Joint channel and computation resource assignment: The column denotes work where channel and computation resource assignments are jointly generated. Our work differs from the rest in that we assign aggregate channel resources from the network operator to each BS so that it can support its associated mobile device population, i.e., we do not allocate channel and computation resources of each BS and ES to individual MDs.

Soft task deadlines: The work selected in this column considers some form of soft (i.e., statistical) task deadlines. However, the models we use in this paper are quite different with more general underlying assumptions. Since our soft deadline model aims to set bounds on the probability of task deadline violation, we model the complete delay distribution experienced by executed tasks. This includes the BS channel delay (which is modeled by BS specific Markov processes) and the queueing delay experienced at the ES, where execution times can have a general distribution.

Hard task deadlines: Although there is other work selected in this column, a significant difference exists compared with our paper. Our work *always* satisfies all hard task deadlines by incorporating the CLE mechanism into the modeled system. The related work, instead, considers the existence of hard deadlines as a problem constraint that may result in problem infeasibility, which can never happen in our case.

Resource expense: This column denotes work where the resources provided to the MDs are charged by a third-party (e.g., network operator). The work selected considers computational resource expense but not on the wireless BS side. A network profit maximization problem is studied where an expense budget is not considered, unlike the case in our work.

Temporal evolution: Temporal evolution means that the offload periods may include stochastic changes to the wireless channels and the ES, so that this information must be modeled in the problem formulation, as in our paper. The randomness modeled in the selected work has different underlying assumptions compared to our paper.

III. System Model and Problem Formulation

As shown in Fig. 1, we consider a network that consists of N BSs that are owned and operated by a NO. The set of BSs is denoted by $\mathcal{N} = \{1, 2, \ldots, N\}$ and indexed by $n \in \mathcal{N}$. The network also contains an ES. Tasks generated by an MD can be offloaded through the wireless network and executed on the ES.

The NO permits a NL to rent wireless communication and ES computational capacity that the NL can use for mobile computation offloading for its MDs. When this is done, for each BS n, there are up to K_n available channels that can be selected by the NL. The cost of renting a channel from

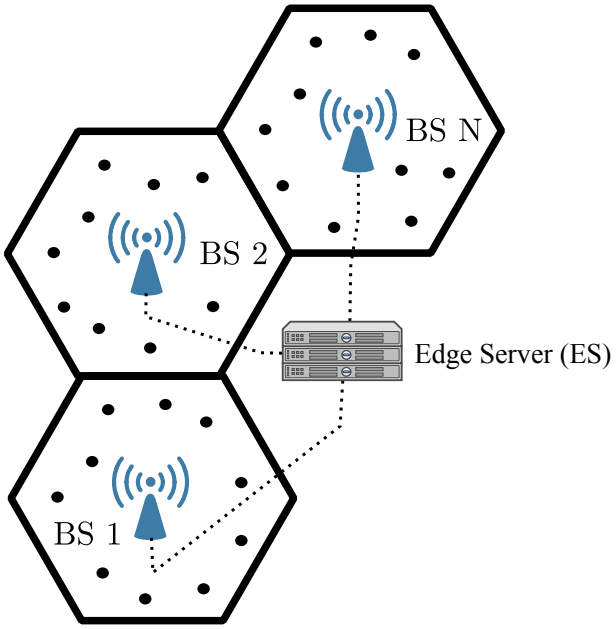


Fig. 1: System Model

BS n is set by the NO to α_n. When a channel is included in the agreement, the NO agrees to provision its network so that sufficient resources are available to allow the traffic generated on the channel to be carried to the ES with an acceptable delay with a high degree of certainty. Since the ES is located at the edge of the network, we focus on the dominant sources of delay, i.e., wireless access at the BSs and task execution at the ES [13].

In order to use the computing resources at the ES, the NL must also lease CPU resources at the ES. The cost (based on the number of CPU cycles per second) for leasing on the CPU resource is denoted by β. The maximum available CPU speed for rental is f^{C} CPU cycles per second.

When an agreement is made between the NO and NL, x_n is defined as the number of channels from BS n that are included, and $y \in [0,1]$ is defined as the fraction of maximum CPU speed at the ES that is included, i.e., the CPU speed available for the NL will be yf^{C}. It is assumed that the NL has a cost budget, denoted by $B^{\max}$. Accordingly, the total rent must satisfy the following constraint:

$$\sum_{n=1}^{N} \alpha_n x_n + \beta y f^{\mathrm{C}} \leq B^{\max}. \tag{1}$$

There are J classes of tasks generated by the MDs, which may need to be offloaded to the ES. Let $\mathcal{J} = \{1, 2, \ldots, J\}$ be the set of task classes. The class j of a task is defined by parameters s_j, q_j, and d_j, where s_j is the input data size in bits, q_j is the computation load in number of CPU cycles, and d_j is the deadline of the task in seconds. In what follows, $\tilde{d}_j = \lfloor d_j/\tau \rfloor$ is the task deadline rounded down to time slots of the same duration τ as the wireless transmission time slots (see below). The probability of a task generated by an MD belonging to class j is denoted by P_j^{C}; we assume that this probability is known, e.g., by observing the past history of offloading requests.

Our objective is to create a NO/NL contract for MCO. In MCO, tasks generated by an MD can be executed either locally (at the MD itself) or offloaded through the network and executed on the ES. We focus on two goals, each depending on how *hard* the task deadline constraint is. Our first goal is to accomplish this so that the mean mobile power consumption is minimized subject to the cost budget constraint and such that the probability that task execution deadline violation is bounded, i.e., the deadline constraints can be violated, i.e., *deadline constraints are soft*. Our second goal is to create a power-efficient, budget-respecting assignment that respects *all* task deadlines, i.e., *deadline constraints are hard*; for that purpose we will employ CLE [29].

We model the wireless channels between the MDs and the BSs as discrete-time Markov processes. It is assumed that there are I_n channel models for BS n, which are a function of the radio propagation environment that the MDs experience at that BS. $\mathcal{I}_n = \{1, 2, \ldots, I_n\}$ is the set of all wireless channel models in BS n. For each of the channel models, the Markovian transition probabilities are defined in the usual way, i.e., given the channel state in the current time slot, there is a probability associated to its transition to another state in the next time slot. The time slot duration is defined to be τ seconds. A class j task, offloaded to BS n by the MD, encounters channel model k with probability $P_{n,j,k}^{\mathrm{G}}$; as with task generation probabilities P_j^{C} above, we assume that this probability is also known, e.g., by observing the past history of offloading requests.

To obtain the design, the decision to offload the execution of a task is made using a *local execute on blocking* (LEB) mechanism as follows. When an MD in BS n generates a class j task, the MD offloads the task if at least one of the x_n channels is available for immediate use. Otherwise, the MD executes the task locally. When a channel is available, the MD begins the offload by uploading the s_j task bits needed for execution on the ES. The LEB mechanism is useful in that either local execution or remote offloading is initiated immediately at task release time, which may be advantageous when task deadlines are tight. It also provides a simple mechanism for assessing when the current level of local congestion is high, which would suggest that local execution is beneficial.

Tasks arrive at BS n according to a stationary process with average arrival rate λ_n tasks per second. According to the LEB mechanism, a new task is blocked from BS channel access if all the x_n channels are busy with uploading other tasks. We denote the task blocking probability at BS n by $P_{\mathrm{B}n}(x_n)$, which is a function of x_n. For the sake of notation simplicity, we use $P_{\mathrm{B}n}$ in the rest of the paper. Let p^{L} be the power needed in the MD to process tasks. When a class j task is blocked from offloading and executed locally, the local execution time is given as $L_j = q_j/f$, where f is the MD's execution speed in number of CPU cycles per time slot[1]. Define $\bar{L}$ as the average local execution time of tasks. Since the task blocking is caused by channel access, which is the same for all task classes, we

[1] L_j is normally measured in CPU cycles, but in order to apply CLE and to simplify the system, we round it up to a multiple of τ.

have $\bar{L} = \sum_{j=1}^{J} P_j^{\mathrm{C}} L_j$. The average energy consumption for executing a task locally is given by $p^{\mathrm{L}}\bar{L}$. Consider all the tasks that are generated in BS n and blocked from offloading in one second, then the mean energy for executing these tasks locally is

$$E_n^{\mathrm{L}}(x_n) = P_{\mathrm{B}n}\lambda_n p^{\mathrm{L}}\bar{L}, \tag{2}$$

which is the average power consumption of the MDs.

The wireless upload transmission time $t^{\mathrm{W}}_{n,j,k}$ of a jth class task in BS n when the wireless channel model is k, is measured in time slots. The mean wireless upload transmission time $\bar{t}^{\mathrm{W}}_{n,j,k}$ for jth class tasks in BS n according to channel model k can be calculated, since $\Pr[t^{\mathrm{W}}_{n,j,k} = l]$ can be computed for all l from channel model k. Moreover, the mean wireless transmission time $\bar{t}^{\mathrm{W}}_n$ for BS n is

$$\bar{t}^{\mathrm{W}}_n = \sum_{j=1}^{J}\sum_{k=1}^{I_n} P_j^{\mathrm{C}} P^{\mathrm{G}}_{n,j,k}\bar{t}^{\mathrm{W}}_{n,j,k}. \tag{3}$$

Under the stated assumptions, the aggregate mean task arrival rate λ at the ES is given by

$$\lambda = \textstyle\sum_{n=1}^{N}(1 - P_{\mathrm{B}n})\lambda_n. \tag{4}$$

As is normally the case for stability in a single server queueing system, the following constraint must always be satisfied,

$$\lambda < \mu^{\mathrm{C}}, \tag{5}$$

where μ^{C} denotes the mean service rate at the ES, i.e, $\mu^{\mathrm{C}} = yf^{\mathrm{C}}/\sum_{j=1}^{J} P_j^{\mathrm{C}} q_j$. As will become clear later, we can relax this constraint to $\lambda \le yf^{\mathrm{C}}/\sum_{j=1}^{J} P_j^{\mathrm{C}} q_j$ without affecting our proposed solutions.

Let $t^{\mathrm{C}}_{n,j,k}$ be the delay (including both queueing and execution time) experienced by a jth class task from BS n at the ES, under wireless channel model k. It takes continuous values, and $\Pr[t^{\mathrm{C}}_{n,j,k} \le t]$, for any $t \ge 0$, is a function of λ and μ^{C}. In what follows, $\tilde{t}^{\mathrm{C}}_{n,j,k}$ is the discretization of $t^{\mathrm{C}}_{n,j,k}$, measured in time slots; its distribution is calculated by

$$\Pr[\tilde{t}^{\mathrm{C}}_{n,j,k} = b] = \Pr[t^{\mathrm{C}}_{n,j,k} \le b\tau] - \Pr[t^{\mathrm{C}}_{n,j,k} \le (b-1)\tau] \tag{6}$$

for any number of time slots $b \ge 0$. Table II lists the related notation and their associated meanings.

TABLE II: Summary of Notation

Notation	Definition	Units
$\mathcal{N}$	Set of BSs, $\lvert\mathcal{N}\rvert = N$	
$\mathcal{J}$	Set of task classes, $\lvert\mathcal{J}\rvert = J$	
$\mathcal{I}_n$	Set of channel models of BS n, $\lvert\mathcal{I}_n\rvert = I_n$	
K_n	Number of available channels in BS n	
f^{C}	Maximum available ES capacity	CPU cycles/sec
α_n	Unit price of wireless channels from BS n	\$ per channel
β	Unit price of ES capacity	\$ per bps
x_n	Number of channels from BS n	
y	Fraction of maximum ES capacity	
$B^{\max}$	Cost budget	\$
s_j	Data size of a task in class j	bits
q_j	Computation load of a task in class j	CPU cycles
d_j	Deadline of a task in class j	sec
$\tilde{d}_j$	Discretized deadline of a task in class j	Time slots
P_j^{C}	Probability of a task belonging to class j	
$P^{\mathrm{G}}_{n,j,k}$	Probability of a class j task in BS n with channel model k	
$P_{\mathrm{B}n}$	Blocking probability in BS n	
μ^{C}	Mean service rate at the ES	Tasks/sec
λ_n	Average task arrival rate in BS n	Tasks/sec
λ	Aggregate average task arrival rate at ES	Tasks/sec
τ	Time slot	sec
$t^{\mathrm{W}}_{n,j,k}$	Wireless transmission time of a jth class task in BS n with channel model k	Time slots
$\bar{t}^{\mathrm{W}}_{n,j,k}$	Mean wireless transmission time of a jth class task in BS n with channel model k	Time slots
$\bar{t}^{\mathrm{W}}_n$	Mean task uploading transmission time in BS n	Time slots
$t^{\mathrm{C}}_{n,j,k}$	Execution time at ES for class j tasks from BS n with channel model k	sec
$\tilde{t}^{\mathrm{C}}_{n,j,k}$	Discretized value of $t^{\mathrm{C}}_{n,j,k}$	Time slots
t_j^{L}	Latest feasible starting time for local execution	Time slots
ε_j	Tolerable probability a class j task exceeds deadline	
p^{L}	Local energy consumption per time slot	Joules
p^{T}	Wireless transmission energy per time slot	Joules
E_n^{T}	Average MD power consumption for uploading tasks in BS n	Watts
E_n^{C}	Average MD power consumption for uploading and executing tasks in BS n	Watts

A. Problem Formulation with Soft Deadlines

We consider the distribution of total delay for an offloaded task, which is the sum of the *data upload* delay $t^{\mathrm{W}}_{n,j,k}$ and the *task execution at ES* delay $t^{\mathrm{C}}_{n,j,k}$, for BS n, task class j, and channel model k. Note that both delays are random variables. As mentioned earlier, the data transmission delay from the BS to the ES is negligible. In addition, in this paper we consider the case of a very small amount of data returned once the execution is completed, and, therefore, we consider only uploading delays between MD and BS.

We now give a formal definition of soft task deadlines. Following common practice (e.g., [27]) in modelling soft deadlines along the lines of QoS requirements, a jth class task in BS n under wireless channel model k, must have a total delay satisfying

$$\Pr[t^{\mathrm{W}}_{n,j,k} + t^{\mathrm{C}}_{n,j,k} \le d_j] \ge 1 - \varepsilon_j, \tag{7}$$

where $0 < \varepsilon_j \le 1$ is the (given) tolerated probability that the completion time of a class j task exceeds its deadline.[2] Note that $t^{\mathrm{W}}_{n,j,k}$ takes discrete values (number of time slots), $t^{\mathrm{C}}_{n,j,k}$ takes discrete values (number of CPU cycle periods), while d_j is continuous (in seconds), so (7) assumes that all quantities are first converted to secs. Its LHS is a function of x_n, y.

The joint probability distribution of total delay is

[2]The case $\varepsilon_j = 0$ corresponds to the case of hard deadlines, and will be dealt with in the next section.

$$\Pr[t^{\mathrm{W}}_{n,j,k} + t^{\mathrm{C}}_{n,j,k} \le d_j] = \sum_{l=1}^{l_{\max}} \Pr[t^{\mathrm{W}}_{n,j,k} = l] \Pr[t^{\mathrm{C}}_{n,j,k} \le d_j - l\tau], \quad (8)$$

where $l_{\max} = \lfloor (d_j - q_j/yf^{\mathrm{C}})/\tau \rfloor$ is the maximum value that l can take, since q_j/yf^{C} is the execution time at the ES without queueing.

The average power consumption of MDs in BS n to upload tasks that are granted channels for offloading is

$$E^{\mathrm{T}}_n(x_n) = (1 - P_{\mathrm{B}n})\lambda_n p^{\mathrm{T}} \bar{t}^{\mathrm{W}}_n, \quad (9)$$

where p^{T} is the transmission energy per time slot used by the MD for uploading the task bits. Therefore, the expected average power consumption of the MDs for uploading and executing tasks arriving at BS n is $E^{\mathrm{L}}_n(x_n) + E^{\mathrm{T}}_n(x_n)$.

Our objective is to create an allocation that minimizes $E^{\mathrm{L}}_n(x_n) + E^{\mathrm{T}}_n(x_n)$ under the cost budget and deadline constraints (1) and (7). The problem can be formulated as follows:

$$\min_{\mathbf{x},y} \sum_{n=1}^{N} [E^{\mathrm{L}}_n(x_n) + E^{\mathrm{T}}_n(x_n)] \text{ s.t.} \quad (10)$$

$$\sum_{n=1}^{N} \alpha_n x_n + \beta f^{\mathrm{C}} y \le B^{\max} \quad (11)$$

$$\Pr[t^{\mathrm{W}}_{n,j,k} + t^{\mathrm{C}}_{n,j,k} \le d_j] \ge 1 - \varepsilon_j, \quad \forall n, j, k \quad (12)$$

$$(f^{\mathrm{C}} / \sum_{j=1}^{J} P^{\mathrm{C}}_j q_j) y \ge \lambda \quad (13)$$

$$x_n \in \{0, 1, \ldots, K_n\}, \quad \forall n \in \mathcal{N} \quad (14)$$

$$0 \le y \le 1. \quad (15)$$

Constraints (11) and (12) are constraints (1) and (7). Constraint (13) is the (relaxed) queue stability requirement for ES; it is equivalent to (5), since equality leads to infinite mean queueing delay, which is never optimal. The optimization problem (10)-(15) is a MINLP problem. Constraint (14) ensures that the number of channels assigned does not exceed the maximum number available in each BS. Even the fractional relaxation of MINLP problem (10)-(15) is non-convex, due to its objective and constraints (12), and, as a result, it is computationally inefficient to solve it exactly. Hence we are going to propose approximate solutions for it.

B. Problem Formulation with Hard Deadlines

For the case of *hard* deadline constraints, i.e., when the task deadline *must* be respected, we employ CLE [29]. In CLE, local execution of the task may be initiated while offloading is ongoing, so that the task deadline is always met, even if offloading fails to finish in time due to the stochastic nature of the wireless channels. Guaranteeing task completion before its deadline may incur additional costs (due to potentially simultaneous local and remote execution of the same task).

When CLE is employed, and in order to ensure that the local execution of a task from class j finishes by its deadline, the latest feasible starting time for local execution is

$$t^{\mathrm{L}}_j = \tilde{d}_j - L_j + 1. \quad (16)$$

The expected wireless transmission power is still given by (9). However, due to the overlap of offloading and local execution because of CLE, there is an extra mean power consumption due to a (potential) overlap with local execution. This expected overlap power consumption is

$$E^{O}_{n,j,k}(x_n, y) = (1 - P_{\mathrm{B}n})\lambda_n \cdot \sum_{t=t^{\mathrm{L}}_j}^{\tilde{d}_j} \sum_{l=1}^{t - \lceil \frac{q_j}{yf^{\mathrm{C}}\tau} \rceil} \Pr[t^{\mathrm{W}}_{n,j,k} = l] \Pr[\tilde{t}^{\mathrm{C}}_{n,j,k} = t - l] \cdot p^{\mathrm{L}}(t - t^{\mathrm{L}}_j + 1), \quad (17)$$

where t is the number of time slots needed to complete the offloaded task, and $(t - t^{\mathrm{L}}_j + 1)$ is the offloading and local execution overlap. Note that $\Pr[\tilde{t}^{\mathrm{C}}_{n,j,k} = t - l]$ is a function of x_n and y.

In case the task offloading goes beyond the finish of the local execution of a task, there is an extra power consumption incurred, whose expected value is

$$E^{B}_{n,j,k}(x_n, y) = (1 - P_{\mathrm{B}n})\lambda_n \cdot \sum_{t=\tilde{d}_j+1}^{+\infty} \sum_{l=1}^{t - \lceil \frac{q_j}{yf^{\mathrm{C}}\tau} \rceil} \Pr[t^{\mathrm{W}}_{n,j,k} = l] \Pr[\tilde{t}^{\mathrm{C}}_{n,j,k} = t - l] p^{\mathrm{L}} L_j. \quad (18)$$

Hence, the expected power consumption of MDs for offloaded tasks in BS n in one second is

$$E^{\mathrm{C}}_n(x_n, y) = E^{T}_n(x_n) + \sum_{j=1}^{J} \sum_{k=1}^{I_n} P^{\mathrm{C}}_j P^{\mathrm{G}}_{n,j,k} [E^{O}_{n,j,k}(x_n, y) + E^{B}_{n,j,k}(x_n, y)], \quad (19)$$

and the expected power consumption of MDs for tasks arriving at BS n in one second is $E^{\mathrm{L}}_n(x_n) + E^{\mathrm{C}}_n(x_n, y)$.

As before, our objective is to minimize the total expected power consumption of the MDs for uploading and executing the tasks that are generated in one second, but now subject to hard deadline constraints. The problem is formulated as follows:

$$\min_{\mathbf{x},y} \sum_{n=1}^{N} [E^{\mathrm{L}}_n(x_n) + E^{\mathrm{C}}_n(x_n, y)] \text{ s.t.} \quad (20)$$

$$\sum_{n=1}^{N} \alpha_n x_n + \beta f^{\mathrm{C}} y \le B^{\max} \quad (21)$$

$$(f^{\mathrm{C}} / \sum_{j=1}^{J} P^{\mathrm{C}}_j q_j) y \ge \lambda \quad (22)$$

$$x_n \in \{0, 1, \ldots, K_n\}, \quad \forall n \in \mathcal{N} \quad (23)$$

$$0 \le y \le 1. \quad (24)$$

IV. General Approximate Allocation Solutions

In this section, we propose approximate solutions for optimization problems (10)-(15) and (20)-(24), by decomposing them into convex optimization subproblems which can be solved efficiently.

A. Approximate Solution for Soft Deadlines

In this subsection, we propose an approximate solution for the optimization problem (10)-(15) by decomposing it into several convex subproblems that can be solved efficiently, solve them, and then keep the best solution. More specifically, we discretize variable $y \in [0,1]$ by breaking $[0,1]$ into Y equal segments, so that y takes values $y_a = a/Y$, for $a = 0, 1, \ldots, Y$. With y fixed, we show that the relaxation of (10)-(15) can be approximated by a convex optimization problem, which can be solved in polynomial time. The resulting (fractional) x_n's are then rounded to integer values (and this is another source of suboptimality for our solution method). After solving the resulting $Y+1$ problems, we output the minimum solution x^*, y^*. Obviously, the quality of the approximation depends on the discretization parameter Y.

We consider the relaxed version of problem (10)-(15), i.e., constraint (14) has been replaced by $x_n \geq 0, \forall n$. With y fixed, we show that the non-convex problem (10)-(15) can be transformed into an equivalent convex optimization problem with the $P_{\mathrm{B}n}$'s as the decision variables.

Lemma 4.1: When y is fixed, constraints (12), (13) can be replaced by constraint

$$\sum_{n=1}^{N} (1 - P_{\mathrm{B}n})\lambda_n \leq \lambda^*. \tag{25}$$

Proof: Note that $\Pr[t^{\mathrm{W}}_{n,j,k} + t^{\mathrm{C}}_{n,j,k} \leq d_j]$ is a monotonically decreasing function of the aggregate mean task arrival rate λ. Hence, by binary search in the range $[0, yf^{\mathrm{C}}/\sum_{j=1}^{J} P^{\mathrm{C}}_j q_j]$, we can approximate within any desired accuracy the maximum possible value of λ that satisfies constraints (12) for all n, j, k. Let λ^* be this maximum value (note that $\lambda^* < \mu^{\mathrm{C}}$, so stability is ensured). Using (4), the lemma follows. □

Next, we note that the blocking probability $P_{\mathrm{B}n}$ is monotonically decreasing in x_n. Let $P^{\min}_{\mathrm{B}n}$ be the blocking probability when $x_n = K_n$. Then we have the following

Lemma 4.2: When y is fixed, constraints (14) can be replaced by the equivalent constraints

$$P^{\min}_{\mathrm{B}n} \leq P_{\mathrm{B}n} \leq 1, \ \forall n \in \mathcal{N}. \tag{26}$$

Constraint (11) is the only remaining constraint with an explicit dependence on the x_n's. Since $P_{\mathrm{B}n}$ is a function of x_n, one could potentially use its inverse to replace x_n with a function of $P_{\mathrm{B}n}$. However, such an inversion function may not exist explicitly (and even if it does, it may be non-convex). In its stead, we can use a convex upper bound approximation F of the inversion of blocking probability, so that

$$x_n \leq F(P_{\mathrm{B}n}), \ \forall n \in \mathcal{N}. \tag{27}$$

Hence, the new convex optimization problem that approximates the original one when y is fixed, is the following:

$$\min_{\mathbf{P}_{\mathrm{B}}} \ \sum_{n=1}^{N} [E^{\mathrm{L}}_n(P_{\mathrm{B}n}) + E^{\mathrm{T}}_n(P_{\mathrm{B}n})] \text{ s.t.} \tag{28}$$

$$\sum_{n=1}^{N} \alpha_n F(P_{\mathrm{B}n}) \leq B^{\max} - \beta f^{\mathrm{C}} y \tag{29}$$

$$\sum_{n=1}^{N} (1 - P_{\mathrm{B}n})\lambda_n \leq \lambda^* \tag{30}$$

$$P^{\min}_{\mathrm{B}n} \leq P_{\mathrm{B}n} \leq 1, \qquad \forall n \in \mathcal{N}. \tag{31}$$

After solving (28)-(31) and obtaining the $P_{\mathrm{B}n}$'s, we can compute the largest integral x^*_n which achieves a blocking probability equal to or bigger than $P_{\mathrm{B}n}$, for all $n \in \mathcal{N}$.

Algorithm 1 General Case Approximation for Soft Deadlines (GCASD)

Require: $\lambda_n, p^{\mathrm{T}}, p^{\mathrm{L}}, \alpha_n, K_n, \beta, f^{\mathrm{C}}, Y, s_j, d_j, q_j, P^{\mathrm{C}}_j, P^{\mathrm{G}}_{n,j,k}$, PDFs of $t^{\mathrm{W}}, t^{\mathrm{C}}$

1: $cost^* = \infty$
2: **for all** $a = 0, \ldots, Y$ **do**
3: $y = a/Y$
4: Obtain λ^*, the upper bound of λ, by binary search in $[0, \mu^{\mathrm{C}}]$
5: $[P_{\mathrm{B}}, cost] =$ [solution, objective] of (28)-(31)
6: $x_{int} =$ max integral x with blocking probabilities $\geq P_{\mathrm{B}}$
7: **if** $cost < cost^*$ **then**
8: $x^* = x_{int}; y^* = y; cost^* = cost$
9: **end if**
10: **end for**
11: **return** x^*, y^*

Algorithm GCASD (cf. Algorithm 1) codifies the solution method described above.

Theorem 4.1: Algorithm GCASD runs in $O(Y(\mathcal{L} + \log \frac{\mu^{\mathrm{C}}}{\epsilon} + N \log K_{\max}))$ time, where $O(\mathcal{L})$ is the running time for solving convex program (28)-(31).

Proof: Line 4 of the algorithm applies binary search in $[0, \mu^{\mathrm{C}}]$ in order to get a λ^* within ϵ of the optimal and takes time $O(\log \frac{\mu^{\mathrm{C}}}{\epsilon})$. Line 5 solves the convex problem (28)-(31) in time $O(\mathcal{L})$, e.g., by interior point methods (cf. Ch. 11 of [42]). Line 6 takes time $O(N \log K_{\max})$, by applying binary search in the range $[0, K_{\max}]$ for each $x_n, n = 1, 2, \ldots, N$ (recall that $K_{\max}$ is the largest K_n). Hence every iteration of the for-loop of lines 2-10 runs in time $O(\mathcal{L} + \log \frac{\mu^{\mathrm{C}}}{\epsilon} + N \log K_{\max})$, and there are $O(Y)$ iterations (recall that Y is the granularity of y). The theorem follows. □

B. Approximate Solution for Hard Deadlines

In this subsection, we use a similar approach in order to solve (20)-(24). Here we decompose the original problem into several subproblems by discretizing both variable y as before, *and* λ. Then, for every possible (fixed) pair (y, λ), the non-convex problem (20)-(24) can be transformed into a convex optimization problem with $P_{\mathrm{B}n}$ as its decision variables, which can be solved in polynomial time. By calculating the pair (y^*, λ^*) whose subproblem achieves minimum average power consumption, integer values x^*_n for the original optimization problem are obtained from $P^*_{\mathrm{B}n}$.

In more detail, we discretize $y \in [0,1]$ by breaking $[0,1]$ into Y equal segments, and then we discretize $\lambda \in [0, yf^{\mathrm{C}}/\sum_{j=1}^{J} P^{\mathrm{C}}_j q_j]$ by breaking interval

$[0, yf^{\mathrm{C}}/\sum_{j=1}^{J} P_j^{\mathrm{C}} q_j]$ into Λ equal segments. At iteration (m,i) of this discretization, $y = y^{(m)}$ and $\lambda = \lambda^{(i)}$ are fixed. Then $\Pr[\tilde{t}^{\mathrm{C}}_{n,j,k} = t - l]$ can be calculated directly for any t and l, and the original optimization problem (20)-(24) becomes

$$\min_{\mathbf{x}} \sum_{n=1}^{N} [E_n^{\mathrm{L}}(x_n) + E_n^{\mathrm{C}}(x_n)] \text{ s.t.} \tag{32}$$

$$\sum_{n=1}^{N} \alpha_n x_n \leq B^{\max} - \beta f^{C} y^{(m)} \tag{33}$$

$$\sum_{n=1}^{N} (1 - P_{\mathrm{B}n}) \lambda_n \leq \lambda^{(i)} \tag{34}$$

$$x_n \in \{0, 1, \ldots, K_n\}, \qquad \forall n \in \mathcal{N}. \tag{35}$$

This is still a non-convex non-linear integer program, which cannot be solved efficiently. As in Section III, and by using (26)-(27), it becomes

$$\min_{\mathbf{P}_{\mathrm{B}}} \sum_{n=1}^{N} [E_n^{\mathrm{L}}(P_{\mathrm{B}n}) + E_n^{\mathrm{C}}(P_{\mathrm{B}n})] \text{ s.t.} \tag{36}$$

$$\sum_{n=1}^{N} \alpha_n F(P_{\mathrm{B}n}) \leq B^{\max} - \beta f^{C} y^{(m)} \tag{37}$$

$$\sum_{n=1}^{N} (1 - P_{\mathrm{B}n}) \lambda_n \leq \lambda^{(i)} \tag{38}$$

$$P_{\mathrm{B}n}^{\min} \leq P_{\mathrm{B}n} \leq 1, \qquad \forall n \in \mathcal{N}. \tag{39}$$

Problem (36)-(39) is a convex program and can be solved efficiently. Hence, we can obtain the optimal blocking probabilities $P^*_{\mathrm{B}n}$, corresponding to a pair $(y^{(m)}, \lambda^{(i)})$. We can compute the largest integral x_n^* which achieves blocking probabilities no smaller than $P^*_{\mathrm{B}n}$, for all $n \in \mathcal{N}$, by using binary search based on the fact that the P_{Bn}'s are decreasing functions of the x_n's. After collecting the solutions for all iterations (m,i), we output the minimum cost one $\mathbf{x}^*, y^*$.

Algorithm 2 General Case Approximation for Hard Deadlines (GCAHD)

Require: $\lambda_n, p^{\mathrm{T}}, p^{\mathrm{L}}, \alpha_n, K_n, \beta, f^{\mathrm{C}}, Y, s_j, d_j, q_j, \Lambda, P_j^{\mathrm{C}}, P^{\mathrm{G}}_{n,j,k}$, PDFs of $t^{\mathrm{W}}, t^{\mathrm{C}}$

1: $cost^* = \infty, y = 0, \lambda = 0$
2: **while** $y \leq 1$ **do**
3: **while** $\lambda \leq yf^{\mathrm{C}}/\sum_{j=1}^{J} P_j^{\mathrm{C}} q_j$ **do**
4: $[P_{\mathrm{B}}, cost] =$ [solution, objective] of (36)-(39)
5: $x_{int} =$ max integral x with blocking probabilities $\geq P_{\mathrm{B}}$
6: **if** $cost < cost^*$ **then**
7: $x^* = x_{int}; y^* = y; cost^* = cost$
8: **end if**
9: $\lambda = \lambda + \frac{yf^{\mathrm{C}}/\sum_{j=1}^{J} P_j^{\mathrm{C}} q_j}{\Lambda}$
10: **end while**
11: $y = y + \frac{1}{Y}$
12: **end while**
13: **return** x^*, y^*

Algorithm GCAHD (cf. Algorithm 2) codifies the solution method described above.

Theorem 4.2: Algorithm GCAHD runs in $O(Y\Lambda(\mathcal{L} + N \log K_{\max}))$ time, where $O(\mathcal{L})$ is the running time for solving convex program (36)-(39).

Proof: Line 4 solves convex problem (36)-(39) in time $O(\mathcal{L})$, e.g., by interior point methods (cf. Ch. 11 of [42]). Line 5 takes time $O(N \log K_{\max})$, by applying binary search in the range $[0, K_{\max}]$ for each $x_n, n = 1, 2, \ldots, N$ (recall that $K_{\max}$ is the largest K_n). Therefore, an iteration of the inner while-loop (lines 3-10) takes time $O(\mathcal{L} + N \log K_{\max})$, for a total of Λ iterations, while the outer while-loop (lines 2-12) runs for a total of Y iterations (recall that Y and Λ are the granularity of y and λ respectively). The theorem follows. □

V. Task Arrival and Offloading Assumptions

In the remainder of this paper, we assume that tasks arrive from the MDs at BS n according to a Poisson process with mean arrival rate λ_n. The Poisson process assumption is commonly made in this type of situation, since the number of mobile devices in a given coverage area is typically quite large, each contributing to a small fraction of the total load [43]. In this case, we can invoke the *insensitivity property* of the Erlang B formula, to compute the probability of blocking at each BS [44]. Note that, typically, the Erlang B result is derived using the $M/M/N/N$ Markovian queue, which assumes exponentially distributed channel upload (i.e., service) times [45]. Due to insensitivity, the result holds for any service time distribution with the same mean. Therefore, the blocking probability for a task arriving at BS n is

$$P_{\mathrm{B}n} = \left(\frac{\lambda_n}{\mu_n^{\mathrm{W}}}\right)^{x_n} \frac{1}{x_n!} \left[\sum_{r=0}^{x_n} \left(\frac{\lambda_n}{\mu_n^{\mathrm{W}}}\right)^{r} \frac{1}{r!}\right]^{-1} \tag{40}$$

where μ_n^{W} denotes the mean service rate, which can be calculated by $\mu_n^{\mathrm{W}} = 1/\bar{t}_n^{\mathrm{W}}$. Function (40) is convex in x_n [46].

Note that due to the Poisson process task arrival assumption, the channel state sampled by arriving tasks is given by the steady-state equilibrium probability distribution of the Markovian channel at that MD. This follows from the PASTA rule [47].

We assume that the aggregate task arrival process at ES is Poisson [48], and, therefore, arriving tasks sample the asymptotic equilibrium state distribution of ES. This approximation is justified due to the mixing of arrivals at ES from BSs operating independently. In this case, ES can be modeled as an $M/G/1$ queue, whose waiting time is given by the random variable w^{C}. Given λ and knowledge of the data upload distribution, the distribution of w^{C} can be obtained by numerical inversion of the probability generating function of system waiting time for $M/G/1$ [43]. In this case, the execution time of a task at the ES depends only on which class it belongs to, i.e., $t^{\mathrm{C}}_{n,j,k} = t^{\mathrm{C}}_j$, for all n and k, and $t^{\mathrm{C}}_j = w^{\mathrm{C}} + q_j/yf^{\mathrm{C}}$. Thus, $\Pr[t^{\mathrm{W}}_{n,j,k} + t^{\mathrm{C}}_j \leq d_j]$ can be easily obtained.

When applying algorithms GCASD (Algorithm 1) and GCAHD (Algorithm 2) in this case, the upper bound F used in problem (28)-(31) and (36)-(39) becomes [49]:

$$x_n \leq \frac{\lambda_n}{\mu_n^{\mathrm{W}}}(1-P_{\mathrm{B}n}) + \frac{1}{P_{\mathrm{B}n}},\ \forall n. \tag{41}$$

A. Approximation with Soft Deadlines

In this case, problem (28)-(31) becomes:

$$\min_{\mathbf{P}_{\mathrm{B}}} \sum_{n=1}^{N}[E_n^{\mathrm{L}}(P_{\mathrm{B}n}) + E_n^{\mathrm{T}}(P_{\mathrm{B}n})] \text{ s.t.} \tag{42}$$

$$\sum_{n=1}^{N}\alpha_n\left(\frac{\lambda_n}{\mu_n^{\mathrm{W}}}(1-P_{\mathrm{B}n}) + \frac{1}{P_{\mathrm{B}n}}\right) \leq B^{\max} - \beta f^{\mathrm{C}} y \tag{43}$$

$$\sum_{n=1}^{N}(1-P_{\mathrm{B}n})\lambda_n \leq \lambda^* \tag{44}$$

$$P_{\mathrm{B}n}^{\min} \leq P_{\mathrm{B}n} \leq 1, \qquad \forall n \in \mathcal{N}. \tag{45}$$

Problem (42)-(45) is convex, and Algorithm 1 can be implemented efficiently according to Theorem 4.1.

B. Approximation with Hard Deadlines

In this case, problem (36)-(39) becomes

$$\min_{\mathbf{P}_{\mathrm{B}}} \sum_{n=1}^{N}[E_n^{\mathrm{L}}(P_{\mathrm{B}n}) + E_n^{\mathrm{C}}(P_{\mathrm{B}n})] \text{ s.t.} \tag{46}$$

$$\sum_{n=1}^{N}\alpha_n\left(\frac{\lambda_n}{\mu_n^{\mathrm{W}}}(1-P_{\mathrm{B}n}) + \frac{1}{P_{\mathrm{B}n}}\right) \leq B^{\max} - \beta f^{C} y^{(m)} \tag{47}$$

$$\sum_{n=1}^{N}(1-P_{\mathrm{B}n})\lambda_n \leq \lambda^{(i)} \tag{48}$$

$$P_{\mathrm{B}n}^{\min} \leq P_{\mathrm{B}n} \leq 1, \qquad \forall n \in \mathcal{N}. \tag{49}$$

Problem (46)-(49) is convex, and Algorithm 2 can be implemented efficiently according to Theorem 4.2.

VI. Simulation Results

In this section, we present simulation results to demonstrate the performance of our proposed algorithms GCASD (Algorithm 1) and GCAHD (Algorithm 2). We adopt the two-state Gilbert-Elliot channel model [50], i.e., the channel states change by following a Markov chain with two states, "Good" (G) and "Bad" (B). This model is commonly used to characterize the effects of burst noise in wireless channels, where the channel can abruptly transition between good and bad conditions [51]. The Gilbert-Elliot channel is a difficult one for computation offloading algorithms to deal with compared to those where there is much more correlation in the channel quality as the offloading progresses. Let B_g and B_b, respectively, be the data transmission rate when the channel is in the G and B states. We consider that all channels have the same B_g and B_b values but differ in their state transition probabilities that result in different propagation models. The transition probabilities for propagation model k in BS n are denoted as $P_{n,k}^{\mathrm{GG}}, P_{n,k}^{\mathrm{GB}}$, $P_{n,k}^{\mathrm{BG}}$, and $P_{n,k}^{\mathrm{BB}}$. In each time slot, the channel state Markov chain transitions in accordance with these probabilities. Denote $\pi_{n,k}^{\mathrm{G}}$ and $\pi_{n,k}^{\mathrm{B}}$, respectively, as the stationary probabilities of a channel in BS n for propagation model k being in the G and B states. Two sets of simulations are performed with set 1 for single class of tasks and set 2 for multiple classes of tasks. Default parameters used in the simulations are summarized in Table III. The parameter settings that we use were taken from the references [23], [26] and [33]. These references summarize parameter settings for various types of applications including those that are inherently delay sensitive, such as gaming, face recognition and healthcare use. We intentionally use a wide range of parameter values based on the referenced ranges so that we can make conclusions that apply in general settings.

TABLE III: Default Parameters

Parameter	Value in set 1	Value in set 2
τ	1 s	
p^{L}	250 mW	
p^{T}	2.5 mW	
λ_n	11, 13, 15 tasks/s	
K_n	15, 15, 20	
α_n	1, 1, 1 \$	
β	0.3×10^{-6} \$	0.25×10^{-6} \$
f^{C}	75M cycles/s	200M cycles/s
f	1M cycles/s	2M cycles/s
$B^{\max}$	140 \$	90 \$
Bg, B_b	2M, 0 bits per time slot	5M, 1M bits per time slot
s_j	2M bits	5M, 10M, 15M bits
d_j	4 s	6, 11, 16 s
q_j	3M CPU cycles	10M, 20M, 30M CPU cycles

A. Simulation set 1: single class of tasks

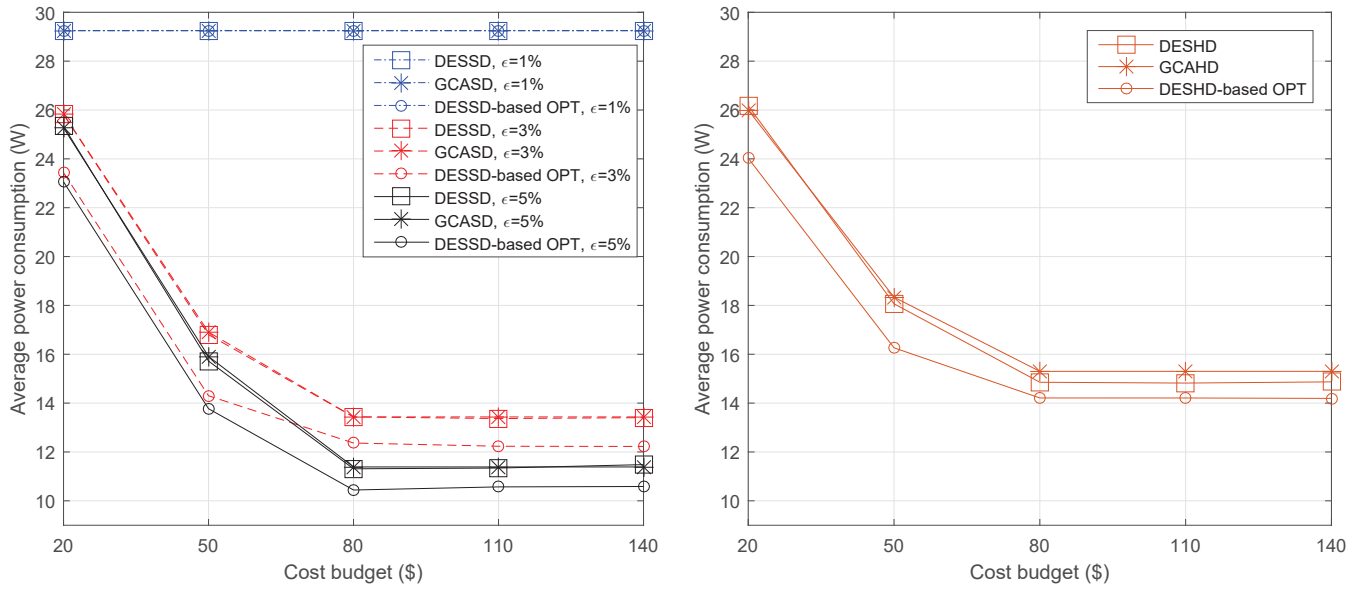
In this subsection, we will assume that all the tasks generated at the MDs have the same data size s and same computation load q, i.e., $s_j = s$ and $q_j = q$ for all j. When the channel is in the G state, the transmission rate of the wireless channel allows a task to be uploaded within one time slot; while when the channel is in the B state, the data transmission rate is zero. Since there is only one class of the tasks, subscript j can be dropped from the notation.

Let $t_{n,k}^{\mathrm{W}}$ be the time needed for uploading a task in BS n with channel model k. The probability that one task in BS n with channel model k can be uploaded in l time slots is given as follows

$$\Pr[t_{n,k}^{\mathrm{W}} = l] = \begin{cases} \pi_{n,k}^{\mathrm{G}}, & \text{when } l = 1 \\ \pi_{n,k}^{\mathrm{B}} {P_{n,k}^{\mathrm{BB}}}^{l-2} P_{n,k}^{\mathrm{BG}}, & \text{when } l \geq 2 \end{cases} \tag{50}$$

The mean wireless transmission time of a task in BS n uploaded through a channel with propagation model k can be calculated as follows

$$\bar{t}_{n,k}^{\mathrm{W}} = \sum_{l=1}^{\infty} l \Pr[t_{n,k}^{\mathrm{W}} = l] = 1 + \frac{P_{n,k}^{\mathrm{GB}}}{{P_{n,k}^{\mathrm{BG}}}^2 + P_{n,k}^{\mathrm{GB}} P_{n,k}^{\mathrm{BG}}}. \tag{51}$$



(a) Soft deadlines

(b) Hard deadlines

Fig. 2: Average power consumption versus cost budget (Single class of tasks)

Based on this, the mean wireless transmission time of the tasks in BS n is $\bar{t}_n^{\mathrm{W}} = \sum_{k=1}^{I_n} P_{n,k}^{\mathrm{G}} \bar{t}_{n,k}^{\mathrm{W}}$, where $P_{n,k}^{\mathrm{G}}$ is the probability that a task in BS n is uploaded through a channel with propagation model k.

With a single class of tasks, the ES server becomes an $M/D/1$ queueing system, $t_{n,j,k}^{\mathrm{C}} = t^{\mathrm{C}}$ for all n, j and k, and the distribution of delay is given by [52]

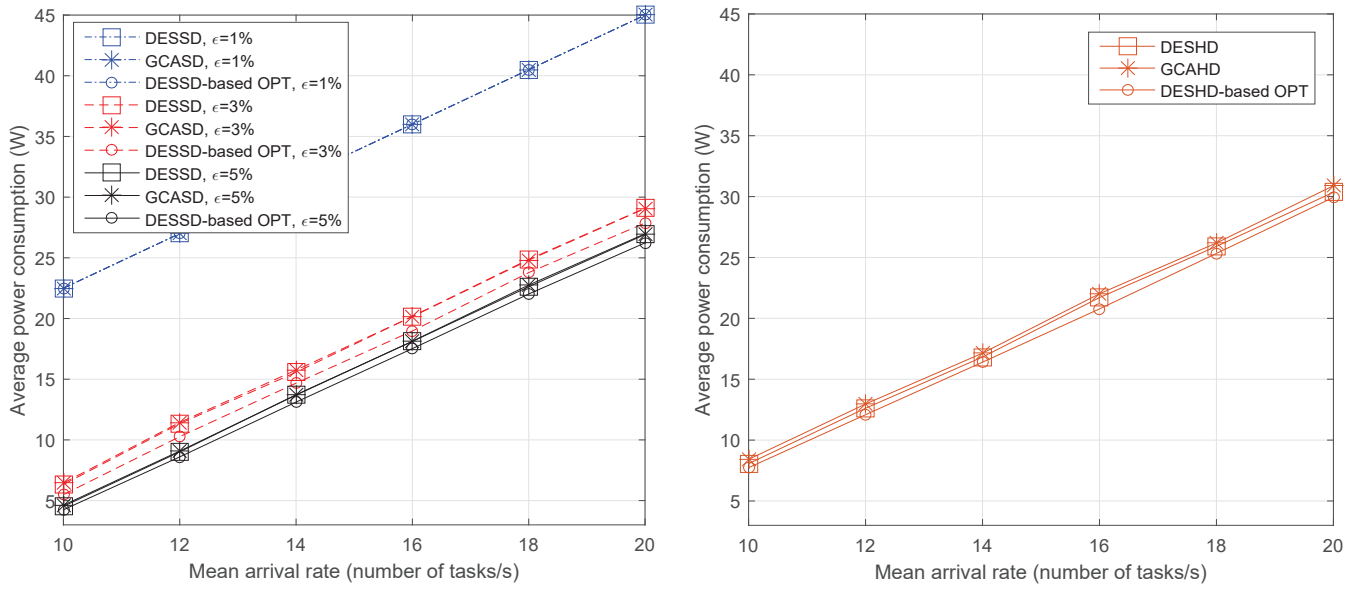
$$\Pr[t^{\mathrm{C}} \leq \hat{t}] = \left(1 - \frac{\lambda}{\mu^{\mathrm{C}}}\right) \sum_{z=0}^{\lfloor \hat{t}\mu^{\mathrm{C}} \rfloor} \frac{[\lambda(\frac{z}{\mu^{\mathrm{C}}} - \hat{t})]^z}{z!} e^{-\lambda(\frac{z}{\mu^{\mathrm{C}}} - \hat{t})} \quad (52)$$

where $\mu^{\mathrm{C}} = y f^{\mathrm{C}}/q$.

For comparison, we also run a discrete event simulation (DES) of the system using the x_n's and y solutions obtained from the proposed algorithms to validate our model assumptions, and these solutions are denoted as DESSD and DESHD, respectively, for the soft deadline (SD) and hard deadline (HD) cases. In addition, we simulate a DES-based OPT scheme for each proposed algorithm as follows. For GCASD, we first obtain all the possible combinations of x_n's under constraint (14); for a given combination of x_n's, we can obtain the solution of y based on (11) and (15), and then check if constraint (13) is satisfied based on the current set of x_n's and y. If not, we go to the next set of x_n's and repeat this procedure. If it is satisfied, we use this set of x_n's and y to run the DES for the system, and then check if (12) is satisfied. If not, we proceed to the next combination of x_n's and repeat the above procedure. If the constraints are satisfied, we save the obtained average power. After going through all the possible combinations of x_n's, we obtain the minimum average power and the corresponding x_n's and y. For GCAHD, we first obtain all the possible combinations of x_n's under constraint (23); for a given combination of x_n's, we can obtain the solution of y based on (21) and (24), and then check if constraint (22) is satisfied based on the current set of x_n's and y. If not, we go to the next set of x_n's and repeat this procedure. If it is satisfied, we use this set of x_n's and y to run the DES for the system. Then, we save the obtained mean power consumption. After going through all the possible combinations of x_n's, we obtain the minimum average power and the corresponding x_n's and y.

In the simulation, we consider a cellular network consisting of 3 BSs. There are two propagation models at each BS with transition probabilities $P_{n,1}^{\mathrm{GG}} = 0.9$, $P_{n,2}^{\mathrm{GG}} = 0.7$, $P_{n,1}^{\mathrm{BB}} = 0.1$, and $P_{n,2}^{\mathrm{BB}} = 0.3$ for $n = 1, 2, 3$. The probabilities of the different channel models in BS 1 are $P_{1,1}^{\mathrm{G}} = 0.8$ and $P_{1,2}^{\mathrm{G}} = 0.2$; and those in BSs 2 and 3 are $P_{2,1}^{\mathrm{G}} = 0.5$, $P_{2,2}^{\mathrm{G}} = 0.5$, $P_{3,1}^{\mathrm{G}} = 0.2$, and $P_{3,2}^{\mathrm{G}} = 0.8$.

Figs. 2(a) and 2(b) show the average power consumption of MDs versus $B^{\max}$ for the SD and HD cases, respectively. In Fig. 2(a), when the tolerable violation of latency ε is 1%, the average power consumption of MDs is a constant for all the solutions. This is because all the tasks are executed locally regardless of the cost budget, since the tight delay constraints cannot be satisfied if a task is offloaded. When ε is 3% or 5%, some tasks are allowed to be offloaded, and the average power consumption of the MDs decreases with $B^{\max}$ for all the solutions. This happens since, when the cost budget is small, the optimization is constrained by the cost budget, which limits the number of offloaded tasks; and with the increase of $B^{\max}$, more channel and ES resource is available, leading to more MDs offloading their tasks. When $B^{\max}$ is large, the budget constraint is loose, and the task offloading completion is mainly affected by the changing wireless transmission conditions. Fig. 2(a) also shows that the average MD power consumption decreases with ε for all the solutions, since larger ε makes it easier to meet the latency



(a) Soft deadlines

(b) Hard deadlines

Fig. 3: Average power consumption versus mean arrival rate (Single class of tasks)

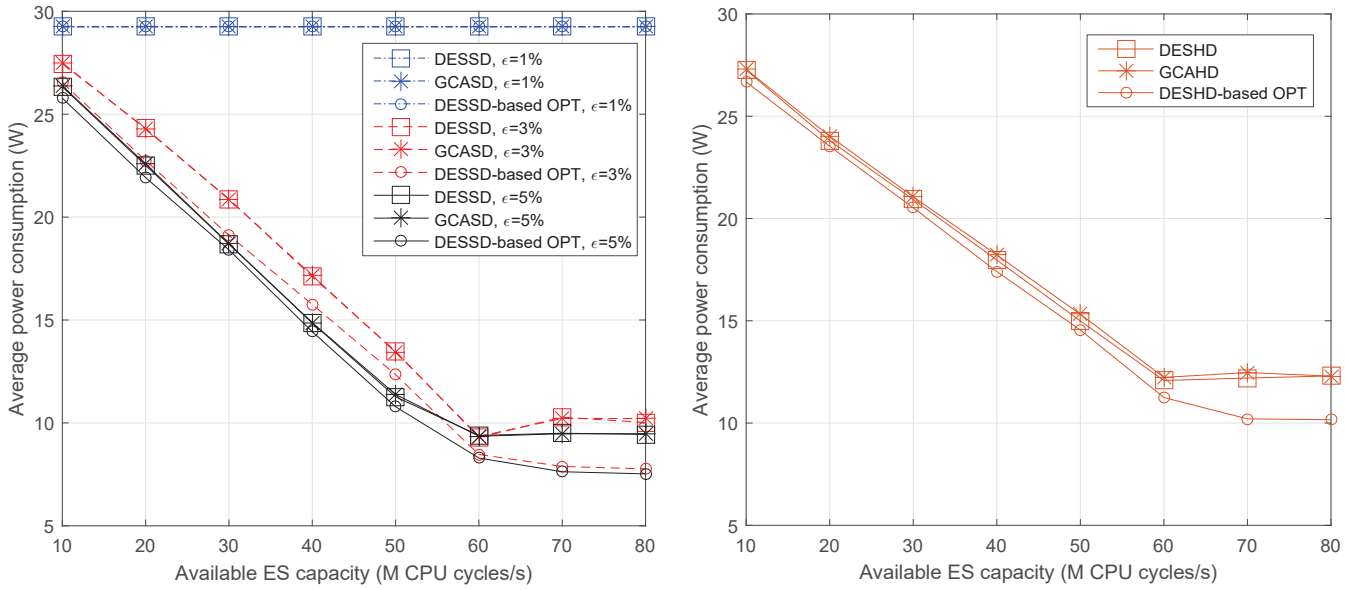
constraint through offloading, which results in more offloaded tasks and saves power in the MDs.

By comparing the average MD power consumption for $\varepsilon = 3\%$ and $\varepsilon = 5\%$ in Fig. 2(a), it is seen that the gap is small when the cost budget is small. The gap then increases as the cost budget increases, and finally becomes constant when the cost budget is sufficiently large. When the cost budget is low, the number of channels is small, which forces most tasks to be executed locally, regardless of the value of ε. As the cost budget increases, more channels are available, and the offloading decisions are determined by both ε and the available channel resources. When the cost budget is sufficiently high, the offloading decisions are mainly determined by the value of ε. The figure also shows that the average MD power consumption using GCASD is almost the same as using DESSD, which validates the model and approximations used in designing GCASD. The performance of GCASD is also close to DESSD-based OPT, which further shows good performance of the former.

By comparing Figs. 2(b) and 2(a), it can be seen that the average MD power consumption for the HD case is slightly higher than that for the SD case with $\varepsilon = 3\%$ and much lower than that for the SD case with $\varepsilon = 1\%$. For the SD case, when $\varepsilon = 1\%$, the tight (soft) delay constraint forces all the tasks to be executed locally, which results in the highest average power consumption of the MDs; and the power consumption decreases as ε increases and more tasks are allowed to be offloaded. Without having to use CLE, the SD solutions result in lower average MD power consumption than the corresponding HD solutions. However, this is at a price that up to ε of the tasks do not meet their completion deadlines. On the other hand, using CLE in the GCAHD only incur slightly higher power consumption of the MDs compared to GCASD when $\varepsilon = 3\%$ For the HD case, the total average power consumption of the MDs decreases with $B^{\max}$ when $B^{\max}$ is small and becomes a constant when $B^{\max}$ becomes larger for all schemes, which is the same as that of the SD case with $\varepsilon = 3\%$ and 5%.

Figs. 3(a) and 3(b) show the average power consumption versus λ_n (same for all BSs) for the SD and HD cases, respectively. The figures show that the power consumption increases linearly with λ_n for all schemes, since both the local execution power and the uploading transmission power are proportional to the mean task arrival rate. The average MD power consumption using GCAHD is close to that using GCASD with $\varepsilon = 3\%$ but much lower than that using GCASD with $\varepsilon = 1\%$. This demonstrates that the use of CLE in GCAHD is minimized, while always ensuring the HD of the tasks. Fig. 3(a) shows that the performance of GCASD is very close to DESSD and DESSD-based OPT; and Fig. 3(b) shows that the performance of GCAHD is very close to DESHD and DESHD-based OPT. These observations are consistent with the ones from Figs. 2(a) and 2(b). This further demonstrates the good performance of GCASD and GCAHD and validates the model and approximations used in designing the proposed algorithms.

Figs. 4(a) and 4(b) show the average power consumption of the MDs versus f^{C}, which is the ES capacity, for the SD and HD cases, respectively. For the SD case with $\varepsilon = 1\%$, all tasks are executed locally; and when $\varepsilon = 3\%$ and 5%, offloading is possible for some tasks, and the number of tasks that can be offloaded increases with the ES capacity, resulting in lower power consumption of the MDs. As the ES capacity is sufficiently high, the average power consumption of MDs becomes a constant, since the offloading decisions are determined by the cost budget which limits the number



(a) Soft deadlines

(b) Hard deadlines

Fig. 4: Average power consumption versus available ES capacity (Single class of tasks)

of wireless channels for uploading tasks. Note that the slight increase in average power consumption when f^{C} is between 60 and 80 is caused by the discretization errors of variable y in algorithms 1 and 2. Increasing the Y values in the algorithms helps reduce the discretization errors but significantly increase the amount of time for running the simulations. Comparing the average power consumption of the HD and the SD cases shown in Figs. 4(a) and 4(b), we have consistent observations as in previous figures.

B. Simulation set 2: multiple classes of tasks

In this subsection, tasks have multiple classes. The two-state Gilbert-Elliot channels are considered. Let B_g and B_b, respectively, be the data transmission rates when a channel is in the G and B states. Given the channel state transision probabilities, the distribution of wireless transmission time $t^{\mathrm{W}}_{n,j,k}$ for uploading a class j task in BS n through a channel with propagation model k can be calculated from [29].

At the ES, the system of serving the uploaded tasks becomes an $M/G/1$ queueing system. Let B be a random variable representing the execution time of the tasks. We have $\Pr[B = \frac{q_j}{yf^{\mathrm{C}}}] = P^{\mathrm{C}}_j$, then the probability density function of B can be written as

$$f_B(\tilde{b}) = \sum_{j=1}^{J} \Pr\left[B = \frac{q_j}{yf^{\mathrm{C}}}\right] \delta\left(\tilde{b} - \frac{q_j}{yf^{\mathrm{C}}}\right) = \sum_{j=1}^{J} P^{\mathrm{C}}_j \delta\left(\tilde{b} - \frac{q_j}{yf^{\mathrm{C}}}\right), \tag{53}$$

and the Laplace-Stieltjes transform of $f_B(\tilde{b})$ is given by

$$g(s) = \sum_{j=1}^{J} P^{\mathrm{C}}_j e^{-\frac{q_j}{yf^{\mathrm{C}}}s}. \tag{54}$$

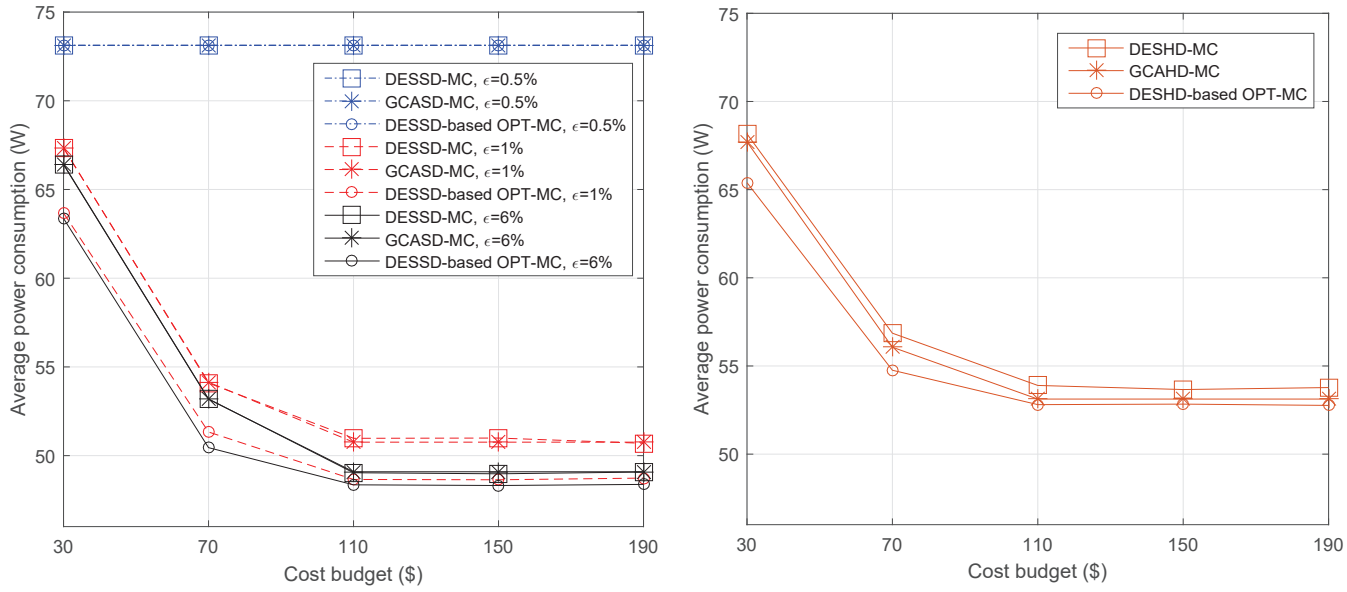
The Laplace-Stieltjes transform of the probability density function of queuing time w^{C} is given by the Pollaczek-Khinchine transform [43] as

$$W^*(s) = \frac{(1-\lambda\bar{b})s}{s - \lambda(1-g(s))}, \tag{55}$$

where $\bar{b}$ is the mean of B. The distribution of w^{C} can be obtained by numerical inversion of (55).

In the simulation, we consider a cellular network consisting of 3 BSs, 3 task classes, and 2 channel propagation models. The channel state transition probabilities are $P^{\mathrm{GG}}_{n,1} = 0.9$, $P^{\mathrm{BB}}_{n,1} = 0.1$, $P^{\mathrm{GG}}_{n,2} = 0.6$, and $P^{\mathrm{BB}}_{n,1} = 0.4$ for $n = 1, 2, 3$. The probabilities of accessing channels with different propagation models in BS 1 are $P^{\mathrm{G}}_{1,1} = 0.8$ and $P^{\mathrm{G}}_{1,2} = 0.2$; those in BSs 2 and 3 are $P^{\mathrm{G}}_{2,1} = 0.5$, $P^{\mathrm{G}}_{2,2} = 0.5$, $P^{\mathrm{G}}_{3,1} = 0.2$, and $P^{\mathrm{G}}_{3,2} = 0.8$. The probabilities of a task belonging to different classes are $P^{\mathrm{C}}_1 = 0.6$, $P^{\mathrm{C}}_2 = 0.3$, and $P^{\mathrm{C}}_3 = 0.1$.

Figs. 5(a) and 5(b) show the average power consumption of MDs versus $B^{\max}$ for the SD and HD cases, respectively. In Fig. 5(a), when ε is 0.5%, all the tasks are executed locally regardless of the cost budget, since offloading cannot satisfy the tight delay constraints. When ε is 1% or 6%, the average power consumption of MDs decreases with $B^{\max}$ and then becomes a constant. By comparing the power consumption of the MD in the SD and HD cases, we can see that the average power consumption of MDs for the HD case is slightly higher than that for the SD case with $\varepsilon = 1\%$ and much lower than that for the SD case with $\varepsilon = 0.5\%$. Figs. 6(a) and 6(b) show the total average power consumption of the MDs versus f^{C}. All the results show that our GCASD and GCAHD solutions achieve the average power consumption performance that is very close to DES-based OPT, and the observations in the multi-class simulations are consistent with the single-class simulations.



(a) Soft deadlines

(b) Hard deadlines

Fig. 5: Average power consumption versus cost budget (Multiple classes of tasks)

VII. Conclusions

This paper has studied joint wireless network and task service allocation for mobile computation offloading. The objective is to minimize the total average power consumption of MDs for completing the arriving tasks, while satisfying the delay constraints of tasks and the cost budget of the network customer. The formulations presented included both soft and hard task completion time deadlines. The designs were formulated as MINLPs and approximate solutions were obtained by decomposing the formulations into convex sub-problems. Simulation results were presented that characterize the performance of the system and show various tradeoffs between task deadline violation, average mobile device power consumption and the cost budget. Results were presented that demonstrate the quality of the proposed solutions, which can achieve close-to-optimum performance over a wide range of system parameters. The optimum allocation was obtained by doing exhaustive search-based discrete event simulations for assigning the wireless channels from each BSs and ES capacity.

References

[1] T. H. Noor, S. Zeadally, A. Alfazi, and Q. Z. Sheng, "Mobile cloud computing: Challenges and future research directions," *Journal of Network and Computer Applications*, vol. 115, pp. 70–85, 2018.

[2] Y. Kwon, H. Yi, D. Kwon, S. Yang, Y. Cho, and Y. Paek, "Precise execution offloading for applications with dynamic behavior in mobile cloud computing," *Pervasive and Mobile Computing*, vol. 27, pp. 58–74, 2016.

[3] H. Ba, W. Heinzelman, C.-A. Janssen, and J. Shi, "Mobile computing-a green computing resource," in *2013 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2013, pp. 4451–4456.

[4] Z. Gu, R. Takahashi, and Y. Fukazawa, "Real-time resources allocation framework for multi-task offloading in mobile cloud computing," in *2019 International Conference on Computer, Information and Telecommunication Systems (CITS)*. IEEE, 2019, pp. 1–5.

[5] J. Zhang, X. Hu, Z. Ning, E. C.-H. Ngai, L. Zhou, J. Wei, J. Cheng, and B. Hu, "Energy-latency tradeoff for energy-aware offloading in mobile edge computing networks," *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 2633–2645, 2017.

[6] G. Huerta-Canepa and D. Lee, "A virtual cloud computing provider for mobile devices," in *Proceedings of the 1st ACM Workshop on Mobile Cloud Computing Services: Social Networks and Beyond*, June 2010, p. 6.

[7] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "CloneCloud: Elastic execution between mobile device and cloud," in *Proceedings of the Sixth Conference on Computer Systems*, ser. EuroSys '11. New York, NY, USA: ACM, 2011, pp. 301–314. [Online]. Available: http://doi.acm.org/10.1145/1966445.1966473

[8] Y. Shi, S. Chen, and X. Xu, "MAGA: A mobility-aware computation offloading decision for distributed mobile cloud computing," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 164–174, Feb 2018.

[9] S. Zhou, Y. Sun, Z. Jiang, and Z. Niu, "Exploiting moving intelligence: Delay-optimized computation offloading in vehicular fog networks," *IEEE Communications Magazine*, vol. 57, no. 5, pp. 49–55, May 2019.

[10] D. Mazza, D. Tarchi, and G. E. Corazza, "A unified urban mobile cloud computing offloading mechanism for smart cities," *IEEE Communications Magazine*, vol. 55, no. 3, pp. 30–37, March 2017.

[11] H. A. Alameddine, S. Sharafeddine, S. Sebbah, S. Ayoubi, and C. Assi, "Dynamic task offloading and scheduling for low-latency IoT services in multi-access edge computing," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 3, pp. 668–682, 2019.

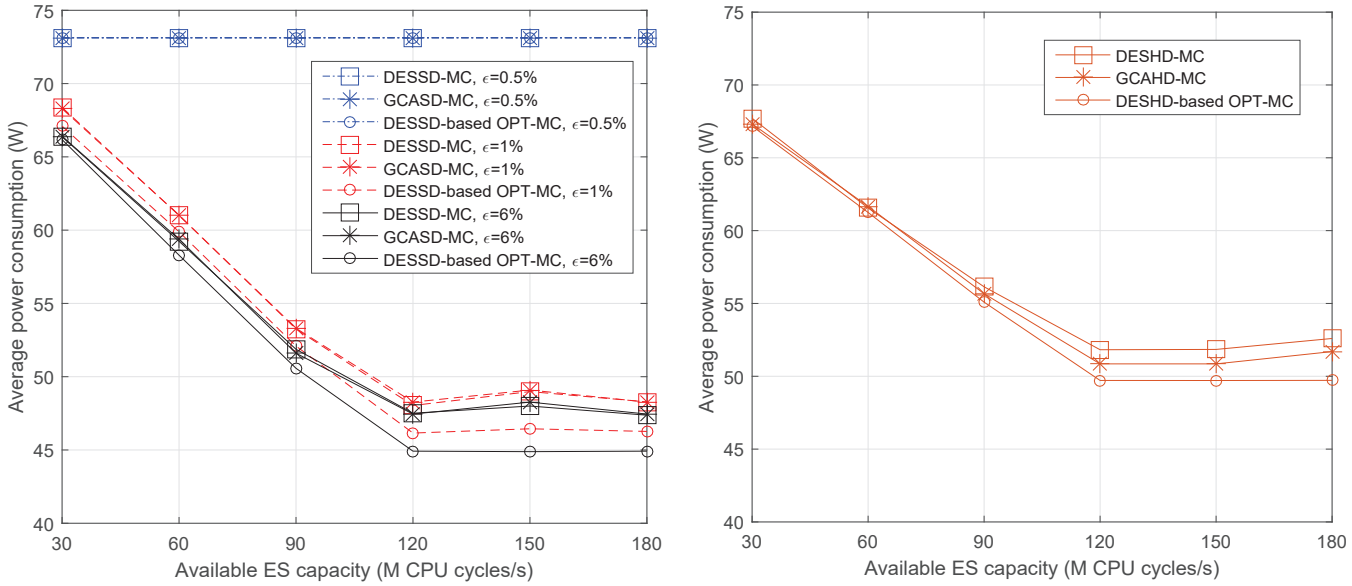
[12] J. Liu, J. Ren, Y. Zhang, X. Peng, Y. Zhang, and Y. Yang, "Efficient dependent task offloading for multiple applications in MEC-cloud system," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2021.

[13] Huawei Inc., "5G network architecture - a high-level perspective," https://www.huawei.com/en/technology-insights/industry-insights/outlook/mobile-broadband/insights-reports/5g-network-architecture, 2016.

[14] O. Muñoz, A. Pascual-Iserte, and J. Vidal, "Optimization of radio and computational resources for energy efficiency in latency-constrained application offloading," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 10, pp. 4738–4755, October 2015.

[15] B. Dab, N. Aitsaadi, and R. Langar, "Joint optimization of offloading and resource allocation scheme for mobile edge computing," in *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, 2019, pp. 1–7.

[16] M. Sheng, Y. Wang, X. Wang, and J. Li, "Energy-efficient multiuser partial computation offloading with collaboration of terminals, radio ac-



(a) Soft deadlines

(b) Hard deadlines

Fig. 6: Average power consumption versus available ES capacity (Multiple classes of tasks)

cess network, and edge server," *IEEE Transactions on Communications*, vol. 68, no. 3, pp. 1524–1537, 2020.

[17] H. Chen, D. Zhao, Q. Chen, and R. Chai, "Joint computation offloading and radio resource allocations in small-cell wireless cellular networks," *IEEE Transactions on Green Communications and Networking*, vol. 4, no. 3, pp. 745–758, 2020.

[18] J. Du, L. Zhao, J. Feng, and X. Chu, "Computation offloading and resource allocation in mixed fog/cloud computing systems with min-max fairness guarantee," *IEEE Transactions on Communications*, vol. 66, no. 4, pp. 1594–1608, April 2018.

[19] X. Yang, X. Yu, H. Huang, and H. Zhu, "Energy efficiency based joint computation offloading and resource allocation in multi-access MEC systems," *IEEE Access*, vol. 7, pp. 117 054–117 062, 2019.

[20] J. Zhang, W. Xia, F. Yan, and L. Shen, "Joint computation offloading and resource allocation optimization in heterogeneous networks with mobile edge computing," *IEEE Access*, vol. 6, pp. 19 324–19 337, 2018.

[21] X. Chen, Z. Liu, Y. Chen, and Z. Li, "Mobile edge computing based task offloading and resource allocation in 5G ultra-dense networks," *IEEE Access*, vol. 7, pp. 184 172–184 182, 2019.

[22] S. Mu, Z. Zhong, and D. Zhao, "Energy-efficient and delay-fair mobile computation offloading," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 15 746–15 759, 2020.

[23] M. Masoudi and C. Cavdar, "Device vs edge computing for mobile services: Delay-aware decision making to minimize power consumption," *IEEE Transactions on Mobile Computing*, vol. 20, no. 12, pp. 3324–3337, 2021.

[24] X. Chen, Y. Cai, Q. Shi, M. Zhao, B. Champagne, and L. Hanzo, "Efficient resource allocation for relay-assisted computation offloading in mobile-edge computing," *IEEE Internet of Things Journal*, vol. 7, no. 3, pp. 2452–2468, 2020.

[25] S. Nath, Y. Li, J. Wu, and P. Fan, "Multi-user multi-channel computation offloading and resource allocation for mobile edge computing," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1–6.

[26] C. Yi, S. Huang, and J. Cai, "Joint resource allocation for device-to-device communication assisted fog computing," *IEEE Transactions on Mobile Computing*, vol. 20, no. 3, pp. 1076–1091, 2021.

[27] H. Park, Y. Jin, J. Yoon, and Y. Yi, "On the economic effects of user-oriented delayed wi-fi offloading," *IEEE Transactions on Wireless Communications*, vol. 15, no. 4, p. 26842697, 2016.

[28] L. Cominardi, T. Deiss, M. Filippou, V. Sciancalepore, F. Giust, and D. Sabella, "MEC support for network slicing: Status and limitations from a standardization viewpoint," *IEEE Communications Standards Magazine*, vol. 4, no. 2, pp. 22–30, 2020.

[29] A. Hekmati, P. Teymoori, T. D. Todd, D. Zhao, and G. Karakostas, "Optimal mobile computation offloading with hard deadline constraints," *IEEE Transactions on Mobile Computing*, vol. 19, no. 9, pp. 2160–2173, 2020.

[30] Q. Li, S. Wang, A. Zhou, X. Ma, F. Yang, and A. X. Liu, "QoS driven task offloading with statistical guarantee in mobile edge computing," *IEEE Transactions on Mobile Computing*, vol. 21, no. 1, pp. 278–290, 2022.

[31] Y. Deng, Z. Chen, and X. Chen, "Resource allocation for multi-user mobile-edge computing systems with delay constraints," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 1–6.

[32] C. W. Zaw, N. H. Tran, Z. Han, and C. S. Hong, "Radio and computing resource allocation in co-located edge computing: A generalized nash equilibrium model," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2021.

[33] S. Yue, J. Ren, N. Qiao, Y. Zhang, H. Jiang, Y. Zhang, and Y. Yang, "TODG: Distributed task offloading with delay guarantees for edge computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 7, pp. 1650–1665, 2022.

[34] J. Ren, G. Yu, Y. Cai, and Y. He, "Latency optimization for resource allocation in mobile-edge computation offloading," *IEEE Transactions on Wireless Communications*, vol. 17, no. 8, pp. 5506–5519, 2018.

[35] Y. Geng, Y. Yang, and G. Cao, "Energy-efficient computation offloading for multicore-based mobile devices," in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, 2018, pp. 46–54.

[36] M.-H. Chen, B. Liang, and M. Dong, "Multi-user multi-task offloading and resource allocation in mobile cloud systems," *IEEE Transactions on Wireless Communications*, vol. 17, no. 10, pp. 6790–6805, 2018.

[37] P. A. Apostolopoulos, G. Fragkos, E. E. Tsiropoulou, and S. Papavassiliou, "Data offloading in UAV-assisted multi-access edge computing systems under resource uncertainty," *IEEE Transactions on Mobile Computing*, vol. 22, no. 1, pp. 175–190, 2023.

[38] J. Peng, H. Qiu, J. Cai, W. Xu, and J. Wang, "D2D-assisted multi-user cooperative partial offloading, transmission scheduling and computation allocating for MEC," *IEEE Transactions on Wireless Communications*, vol. 20, no. 8, pp. 4858–4873, 2021.

[39] W. Feng, J. Tang, N. Zhao, X. Zhang, X. Wang, K.-K. Wong, and J. A. Chambers, "Hybrid beamforming design and resource allocation for UAV-aided wireless-powered mobile edge computing networks with NOMA," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 11, pp. 3271–3286, 2021.

[40] X. Cao, F. Wang, J. Xu, R. Zhang, and S. Cui, "Joint computation and communication cooperation for energy-efficient mobile edge comput-

ing," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4188–4200, 2019.
[41] S. Mu, Z. Zhong, D. Zhao, and M. Ni, "Joint job partitioning and collaborative computation offloading for internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 1046–1059, 2019.
[42] S. P. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2014.
[43] A. Y. Khintchine, "Mathematical theory of a stationary queue," *Matematicheskii Sbornik*, vol. 39, no. 4, pp. 73–84, 1932.
[44] D. Y. Burman, "Insensitivity in queueing systems," *Advances in Applied Probability*, vol. 13, no. 4, pp. 846–859, 1981.
[45] D. J. Daley and L. D. Servi, "Idle and busy periods in stable M/M/k queues," *Journal of Applied Probability*, vol. 35, no. 4, pp. 950–962, 1998.
[46] E. J. Messerli, "B.S.T.J. brief: Proof of a convexity property of the erlang B formula," *The Bell System Technical Journal*, vol. 51, no. 4, pp. 951–953, 1972.
[47] R. W. Wolff, "Poisson arrivals see time averages," *Operations Research*, vol. 30, no. 2, pp. 223–414, 1982.
[48] D. N. Shanbhag and D. G. Tambouratzis, "Erlang's formula and some results on the departure process for a loss system," *Journal of Applied Probability*, vol. 10, no. 1, pp. 233–240, 1973.
[49] S. A. Berezner, A. E. Krzesinski, and P. G. Taylor, "On the inverse of erlang's function," *Journal of Applied Probability*, vol. 35, no. 1, pp. 246–252, 1998.
[50] E. N. Gilbert, "Capacity of a burst-noise channel," *The Bell System Technical Journal*, vol. 39, no. 5, pp. 1253–1265, 1960.
[51] T. Blazek and C. F. Mecklenbräuker, "Measurement-based burst-error performance modeling for cooperative intelligent transport systems," *IEEE Transactions on Intelligent Transportation Systems*, no. 99, pp. 1–10, 2018.
[52] G. J. Franx, "A simple solution for the M/D/1 waiting time distribution," *Operations Research Letters*, vol. 29, no. 5, pp. 221–229, 2001.