Analysis of the Energy-Response Time Tradeoff for Mobile Cloud Offloading Using Combined Metrics

Huaming Wu
Free University of Berlin, Germany
Email: huaming.wu@fu-berlin.de

William Knottenbelt
Imperial College London, UK
Email: wjk@doc.ic.ac.uk

Katinka Wolter
Free University of Berlin, Germany
Email: katinka.wolter@fu-berlin.de

***Abstract*—Mobile offloading migrates heavy computation from mobile devices to cloud servers using one or more communication network channels. Communication interfaces vary in speed, energy consumption and degree of availability. We assume two interfaces: WiFi, which is fast with low energy demand but not always present and cellular, which is slightly slower has higher energy consumption but is present at all times. We study two different communication strategies: one that selects the best available interface for each transmitted packet and the other multiplexes data across available communication channels. Since the latter may experience interrupts in the WiFi connection packets can be delayed. We call it interrupted strategy as opposed to the uninterrupted strategy that transmits packets only over currently available networks.**

Two key concerns of mobile offloading are the energy use of the mobile terminal and the response time experienced by the user of the mobile device. In this context, we investigate three different metrics that express the energy-performance tradeoff, the known Energy-Response time Weighted Sum (EWRS), the Energy-Response time Product (ERP) and the Energy-Response time Weighted Product (ERWP) metric.

We apply the metrics to the two different offloading strategies and find that the conclusions drawn from the analysis depend on the considered metric. In particular, while an additive metric is not normalised, which implies that the term using smaller scale is always favoured, the ERWP metric, which is new in this paper, allows to assign importance to both aspects without being misled by different scales. It combines the advantages of an additive metric and a product.

The interrupted strategy can save energy especially if the focus in the tradeoff metric lies on the energy aspect. In general one can say that the uninterrupted strategy is faster, while the interrupted strategy uses less energy. A fast connection improves the response time much more than the fast repair of a failed connection. In conclusion, a short down-time of the transmission channel can mostly be tolerated.

***Index Terms*—Energy-Performance Tradeoff, Queuing Model, Offloading, Heterogeneous Networks, Mobile Cloud Computing**

I. Introduction

Mobile cloud computing aims at combining the strength of cloud computing with the convenience of mobile terminals. However, limited radio resources or limitations of other communication channels as well as lack of sufficient battery power may significantly impede the improvement of service quality [1] anticipated by using cloud services. Nonetheless computation offloading, which migrates computation-intensive tasks from mobile devices to a remote cloud infrastructure via a network, is a popular approach to alleviate the shortcomings of resource-constrained mobile devices. Since offloading an application to the cloud is not always possible or effective, the decision as to whether to execute a program locally or to offload it requires careful consideration of the nature of the computation and the communication channels available.

Mobile devices are often equipped with multiple wireless interfaces (e.g. cellular and WiFi) for data transfer, with different availabilities, delays and energy costs. Response time and energy consumption are two primary concerns for mobile systems that must be considered when making offloading decisions. However, different applications usually give different relative importance to both factors. For delay-tolerant applications (e.g. iCloud, Dropbox, RSS feeds and participatory sensing), response time is less critical and optimising energy usage is more relevant. Some information is not time-critical and its submission to the server may be delayed until the device enters an energy-efficient network. For delay-sensitive applications (e.g. speed chess game, face recognition, video conferencing and vehicular communications), fast response time is of primary concern while energy consumption is less important. The offloading scheme in which cloud services are accessible with short network latency (e.g. WiFi network) can serve in a better way. Therefore, there exists a fundamental tradeoff between energy consumption and response time in the expected usability of applications [2].

Recently, several researchers have worked on optimising the tradeoff between energy consumption and response time. In [3] and [4], the energy-delay tradeoff has been studied by deciding whether or not and by means of which communication interface to offload a whole application. Instead, an application can consist of several components, or jobs, that are treated separately, and thus offloading decisions should be made for every component. Accordingly, a queueing model has been proposed in [5] to capture the tradeoff between the energy consumption and the response time for mobile cloud offloading based on an additive energy-performance metric, where static and dynamic offloading polices were analysed.

Seamless offloading operation by switching between several transmission technologies has been proposed in [6]. In addition, this work examined the tradeoff between energy consumption for WiFi search and transmission rate when the WiFi network was intermittently available. A stochastic model for that scenario has been developed in [7] using various performance metrics and also intermittently available access links. Energy-efficient delayed network selection has been suggested in [2] and [8] to optimise the tradeoff between energy usage and delay in data transmission by intentionally deferring data transmission until the device meets an energy-efficient network.

The main contributions of this paper are as follows:

- We propose two strategies for mobile cloud offloading and analyse them by means of analytic queueing models: the uninterrupted offloading strategy and the interrupted offloading strategy. The uninterrupted strategy uses WiFi

978-1-4673-8422-3/15 $31.00 © 2015 IEEE
DOI 10.1109/ITC.2015.23

(which we assume to have the best transmission characteristics) whenever possible, but switches to a cellular interface if no WiFi connection exists [9]. The interrupted strategy assigns jobs upon arrival to one of two parallel interfaces with different characteristics which are modelled as two parallel queues[1] (e.g. cellular or WiFi transmission). Data transmission of the WiFi queue can be interrupted for short periods when the connection is lost.

- The models are compared using several metrics. We apply the previously studied Energy-Response time Weighted Sum (ERWS) and compare it with the Energy-Response time Product (ERP). After discussing advantages and disadvantages of both metrics we introduce the Energy-Response time Weighted Product (ERWP), which combines the advantages of both previously studied metrics.

The remainder of this paper is organised as follows. In Section II, we introduce the offloading system and the queueing model for offloading as well as the three considered metrics. In Section III, we analyse the uninterrupted offloading strategy based on the ERWP metric. The interrupted offloading strategy is proposed and analysed in Section IV. Section V evaluates metrics and models using numerical examples. Section VI concludes.

II. System Overview

In this section, we first introduce the overall offloading system, and then focus on its submodules: local execution and remote execution, respectively. Finally, we combine them by using metrics to capture the tradeoff between the mean energy consumption and mean response time.

A. The Offloading System

We model mobile offloading as a queueing system as depicted in Fig. 1. The mobile device, the cloud and the wireless networks are represented as queueing nodes to capture the resource contention and delay on the system. The mobile device executes an application with different types of jobs that can be classified into the following two classes. Each time a job is executed, a decision must be taken into which class it belongs:

- **Unoffloadable**: some jobs should be unconditionally processed locally on the mobile device, either because transferring the relevant information would take more time and energy or because these tasks must access local devices (camera, sensors, user interface, etc.) [10]. Local processing consumes battery power of the mobile device but there are no communication costs or delays.
- **Offloadable**: some jobs are flexible tasks that can be processed either locally on the mobile device, or remotely in a cloud infrastructure through computation offloading. Many tasks fall into this category, and the offloading decision depends on whether the communication costs outweigh the difference between local and remote costs [11].

We do not need to take offloading decisions for unoffloadable jobs. However, as for offloadable ones, the mobile device should judiciously make decisions that optimise the response time energy tradeoff expressed in one of the metrics defined at the end of this section.

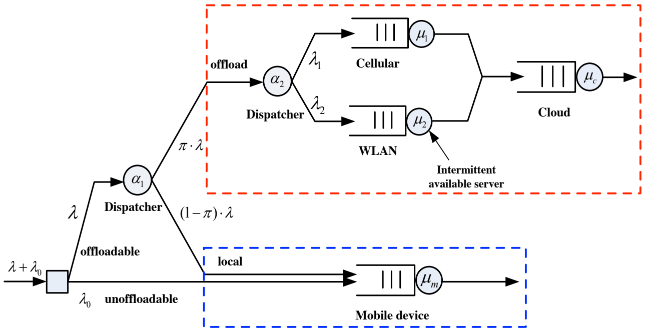


Figure 1. A queueing system for mobile cloud offloading

As indicated in Fig. 1, job arrivals at the mobile device are assumed to follow a Poisson process with an average arrival rate of $\lambda + \lambda_0$, where λ and λ_0 are the rates of offloadable and unoffloadable jobs, respectively. The arrival rate is based on the behavior of the application. The unoffloadable jobs with an arrival rate λ_0 are unconditionally executed locally. As for the offloadable ones, these arrive at rate λ, and the mobile device chooses to offload each job with a probability $0 \leq \pi \leq 1$. As in [12], jobs are offloaded to the cloud following a Poisson process with an average rate of $\lambda_c = \pi \cdot \lambda$, the offloading rate. Similarly, jobs that are proceed locally instead of being offloaded follow a Poisson process with rate $\lambda_m = (1 - \pi) \cdot \lambda$.

There are several ways to offload computation to the cloud, e.g. via a cellular connection (e.g. 2G, 3G and 4G), which is assumed to be the costly connection, or via intermittently available WiFi. We assume that the cellular interface can provide ubiquitous coverage for mobile devices in a wide area, but has lower data transmission rate and consumes more transmission energy than the WiFi interface. In many cases these assumptions are realistic. The mobile device, the cellular and WiFi connections are modelled as $M/M/1$-FCFS queues. The remote cloud is modelled as an $M/M/\infty$ queue, i.e. as a delay center [13]. We denote $1/\mu_m$ and $1/\mu_c$ the expected execution time of jobs on the mobile device and the cloud, respectively. The expected rates to transfer data to the cloud over the cellular network and WiFi are μ_1 and μ_2, respectively.

Two dispatchers are needed: α_1 is used to allocate the offloadable jobs either to the cloud or to the mobile device, while α_2 is used to offload the jobs either via a cellular connection or a WiFi network to the cloud. It should be noted that when $\pi = 0$, all offloadable jobs are processed locally, when $\pi = 1$, they are all offloaded to the cloud. The total cost, in terms of energy or response time for processing all offloadable jobs, is composed of the remote cost (sending some jobs to the cloud and waiting for the cloud to complete them), and the local cost (processing the rest jobs locally on the mobile device).

Our objective is to minimise the mean energy consumption and the mean response time. Since only one option exists for the jobs that cannot be offloaded (as they will always be processed locally) our attention is focused on on the offloadable jobs, where the right decision must be taken whether or not to offload and which interface to use for offloading. The key elements for the considered offloading system are as shown inside the blue block (local execution) and the red block (remote execution), which are analysed separately in the following subsections.

[1] For simplicity we will call the two considered networks WiFi and cellular, but this could be any other technology with equivalent characteristics.

B. Local Execution

As shown inside the blue dotted block in Fig. 1, there are two kinds of jobs (offloadable and unoffloadable) arriving to the processor of the mobile device. We adopt the preemptive scheduling policy here. That is, from the perspective of an offloadable job, the unoffloadable jobs do not exist, since service to the unoffloadable jobs is immediately interrupted upon the arrival of an offloadable job. To the offloadable jobs, the system behaves like an $M/M/1$ queue with arrival rate λ_m and service rate μ_m.

The workload, or utilisation, i.e. the fraction of time when the server is busy, is denoted as: $\rho_m = \lambda_m/\mu_m$. The mean number of jobs in an $M/M/1$ queue is given by:

$$\mathbb{E}[N_m] = \frac{\rho_m}{1-\rho_m}. \tag{1}$$

By Little's Law we obtain the mean response time as:

$$\mathbb{E}[T_m] = \frac{1}{\lambda_m}\mathbb{E}[N_m]. \tag{2}$$

We assume the mobile device consumes energy only when there are jobs in the system and that the mobile device operates at a constant power p_m whenever it is busy [14]. Since $P_m = \lambda_m \mathcal{E}_m$ is the consumed power, the mean energy consumption $\mathbb{E}[\mathcal{E}_m]$ can be more conveniently expressed as:

$$\begin{aligned}\mathbb{E}[\mathcal{E}_m] &= \frac{1}{\lambda_m}\cdot\mathbb{E}[P_m] = \frac{1}{\lambda_m}\cdot p_m \Pr\{N_m > 0\} \\ &= \frac{1}{\lambda_m}\cdot p_m \rho_m, \end{aligned} \tag{3}$$

where $\Pr$ denotes the probability operation.

C. Remote Execution

As shown inside the red dotted block in Fig. 1, the remote execution includes the transmission model and the cloud model. To facilitate the analysis of the mobile cloud offloading system, we assume that a cellular network is available to mobile users all the time while the availability of a WiFi network depends on the location. That is, mobile users move in and out of a WiFi coverage area. We model this time variation of the WiFi connection state by the ON-OFF alternating renewal process $\left(T_{\text{ON}}^{(j)}, T_{\text{OFF}}^{(j)}\right)$, $j \geq 1$, as shown in Fig. 2. The ON periods represent the presence of the WiFi connectivity, while the OFF periods represent the interruption of the WiFi connectivity. During the latter periods data is either not transmitted (because the interface is idle) or it is transmitted only through the cellular network. The duration of each ON period $T_{\text{ON}}^{(j)}$ or OFF period $T_{\text{OFF}}^{(j)}$, is assumed to be an exponentially distributed random variable and independent of the duration of other ON or OFF periods [9].

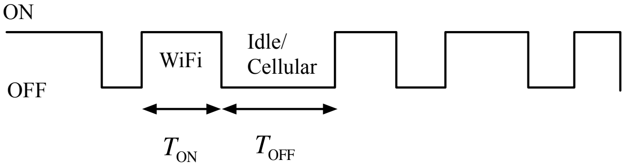


Figure 2. The WiFi network availability model

Accordingly, we build two different offloading strategies:

- **Uninterrupted Offloading Strategy**: we employ a single queue with two states to offload jobs to the cloud server. When there is a WiFi connection available, all jobs are sent over the WiFi network; otherwise, they are sent over the cellular interface as the cellular network is always available [15]. Therefore, the process of offloading jobs to the cloud cannot be interrupted.
- **Interrupted Offloading Strategy**: we assign jobs upon arrival to one of two parallel queues which describe cellular or WiFi transmission [5]. Offloading is interrupted during the periods when WiFi is disconnected. It is a dispatching problem where incoming jobs are splitted on arrival for service by two servers and join before departure. Only when all the jobs are transmitted and have rejoined can the cloud processing start.

We have two states (for the uninterrupted offloading strategy) or two parallel queues (for the interrupted offloading strategy). Upon arrival of a job an assignment decision is made and the job is placed into the corresponding queue (state).

A key assumption in our work is that each server (state) operates at a constant power p_i ($i \in \{1, 2\}$) whenever it is busy. We use $e_i \in \{0, 1\}$ to indicate whether Server (State) i is available or not. If $e_i = 1$, Server (State) i is available, otherwise it is unavailable. Since $P_i = \lambda_i \mathcal{E}_i$ is the consumed power, further by Little's Law, $\mathbb{E}[N_i] = \lambda_i \mathbb{E}[T_i]$, the mean energy consumption and mean response time due to offloading can be calculated as follows, respectively.

$$\begin{aligned}\mathbb{E}[\mathcal{E}_o] &= \frac{1}{\lambda_c}\sum_{i=1}^{2}\lambda_i \mathbb{E}[\mathcal{E}_i] = \frac{1}{\lambda_c}\sum_{i=1}^{2}\mathbb{E}[P_i] \\ &= \frac{1}{\lambda_c}\Big\{\sum_{i=1}^{2} p_i \Pr\{N_i > 0, e_i = 1\}\Big\}, \end{aligned} \tag{4}$$

$$\begin{aligned}\mathbb{E}[T_o] &= \frac{1}{\lambda_c}\sum_{i=1}^{2}\lambda_i \mathbb{E}[T_i] + \mathbb{E}[T_c] \\ &= \frac{1}{\lambda_c}\Big\{\sum_{i=1}^{2}\mathbb{E}[N_i] + \mathbb{E}[N_c]\Big\}, \end{aligned} \tag{5}$$

where λ_i is the mean rate of jobs arriving to Queue (State) i; $\mathbb{E}[\mathcal{E}_i]$ and $\mathbb{E}[T_i]$ are the mean energy consumption and mean response time in Queue (State) i, respectively, and $\mathbb{E}[T_c]$ is the expected execution time in the cloud server.

Since the probability that the corresponding server (state) is busy is equal to the load [14], we have $\Pr\{N_i > 0 | e_i = 1\} = \rho_i$ when the server (state) is always available. Further, we have:

$$\begin{aligned}\Pr\{N_i > 0, e_i = 1\} &= \Pr\{N_i > 0 | e_i = 1\}\cdot\Pr\{e_i = 1\} \\ &= \rho_i \cdot \Pr\{e_i = 1\}. \end{aligned} \tag{6}$$

D. Metrics

The general cost metric includes energy consumption related costs in addition to the usual performance metrics such as the response time. The response time is the time between the arrival of a job until it completes service and departs. The energy consumption is the energy spent on the mobile device in that period. We use queueing theory to model the offloading systems according to different response time energy metrics. We study the tradeoff between the mean energy consumption and mean response time, which is a non-trivial multi-objective optimization problem and define three different metrics in the following subsections.

1) ERWS: The Energy-Response time Weighted Sum (ERWS) is a metric to capture energy-performance tradeoffs and to compare different policies. It is defined by setting the cost function as the weighted sum of both average values, i.e. the ERWS metric:

$$ERWS = \omega\mathbb{E}[\mathcal{E}] + (1-\omega)\mathbb{E}[T], \tag{7}$$

where $\mathbb{E}[\mathcal{E}]$ and $\mathbb{E}[T]$ are the mean energy consumption and mean response time, respectively. ω (ranging between 0 and 1) is a weighting parameter that indicates the relative significance between the energy consumption and response time for the mobile device. Large ω favors energy consumption while small ω favors response time. In some special cases performance can be traded for power consumption and vice versa [16], therefore we can use the ω parameter to express such special cases preferences for different applications.

For Eq. (7), the mean time and energy are additive terms over time; therefore the ERWS metric has the advantage of being analytically tractable and can be optimized via Markov Decision Processes [17]. From the view of minimization, this metric allows comparing arbitrary offloading policies to the optimal offloading policy in our work. However, it has the disadvantage of a linear combination of two metrics on different scales.

2) ERP: The Energy-Response time Product (ERP) metric is widely accepted as a suitable metric to capture energy-performance tradeoffs. It is defined as:

$$ERP = \mathbb{E}[\mathcal{E}] \cdot \mathbb{E}[T]. \tag{8}$$

Minimizing ERP can be seen as maximizing the 'performance-per-joule', with performance being defined as the inverse of mean response time [17].

The energy response time product does not suffer from comparison of different scales. While the ERWS metric implies that a reduction in mean response time from 1000 sec to 999 sec is of the same value as a reduction from 2 sec to 1 sec, the ERP metric implies that a reduction in mean response time from 2 sec to 1 sec is much better than a reduction from 1000 sec to 999 sec, which is indeed a more realistic view on the actual system [17]. However, since ERP is the product of two expectations, it is a difficult metric to address analytically.

3) ERWP: To overcome the disadvantages mentioned above, we propose a new metric named Energy-Response time Weighted Product (ERWP), which combines the strengths of ERWS and ERP. It is defined as:

$$ERWP = \mathbb{E}[\mathcal{E}]^{\omega} \cdot \mathbb{E}[T]^{1-\omega}. \tag{9}$$

We can rewrite Eq. (9) as:

$$ERWP = e^{\omega\cdot\ln(\mathbb{E}[\mathcal{E}])+(1-w)\cdot\ln(\mathbb{E}[T])}. \tag{10}$$

Therefore, it inherits the characteristics of the ERWS metric that assigns different importance weights to energy and time, and has the advantage of being analytically tractable since the logarithmic expectation is additive over time.

Meanwhile, when $\omega = 0.5$, the mean energy consumption and mean response time have equal importance:

$$ERWP = \sqrt{\mathbb{E}[\mathcal{E}] \cdot \mathbb{E}[T]}, \tag{11}$$

which indicates that ERWP metric has the advantage of the ERP metric in being sensitive to difference of the scales of the component metrics.

To the best of our knowledge the ERWP metric has not been treated analytically before.

Based on the above analysis in the local execution and the remote execution for the offloadable jobs, we can derive the ERWP metric from Eq. (9) after combining the results in Eqs. (2)–(5), which is expressed by:

$$\begin{aligned} ERWP &= \Big\{\pi\mathbb{E}[\mathcal{E}_o] + (1-\pi)\mathbb{E}[\mathcal{E}_m]\Big\}^{\omega} \\ &\qquad\qquad \cdot \Big\{\pi\mathbb{E}[T_o] + (1-\pi)\mathbb{E}[T_m]\Big\}^{1-\omega} \\ &= \frac{1}{\lambda}\Big\{\sum_{i=1}^{N}\mathbb{E}[P_i] + \mathbb{E}[P_m]\Big\}^{\omega} \\ &\qquad\qquad \cdot \Big\{\sum_{i=1}^{N}\mathbb{E}[N_i] + \mathbb{E}[N_c] + \mathbb{E}[N_m]\Big\}^{1-\omega}. \end{aligned} \tag{12}$$

Similarly, we can derive the ERWS metric from Eq. (7) and ERP metric from Eq. (8) for the offloadable jobs.

III. Uninterrupted Offloading Strategy

In this section, we analyse the uninterrupted offloading strategy for remote execution. We formulate the queueing model based on the WiFi availability model, and then we use queueing analysis to derive the ERWP metric.

A. Problem Formulation

Figure 3 depicts an uninterrupted offloading strategy based on the WiFi network's availability model. The total cost for offloading a job is composed of the cost for sending the job to the cloud and idly waiting for the cloud to complete the job. We propose a Markov modulated queue for uplink transmission. A single-server queuing system that oscillates between two feasible states is denoted by f_{ON} and f_{OFF}. The persistence of the system at any state is governed by a random mechanism: if the system functions at state f_{ON} it tends 'to jump' to the other state with Poisson intensity ξ and if the system functions at state f_{OFF} it tends 'to jump' to the other state with Poisson intensity η [18].

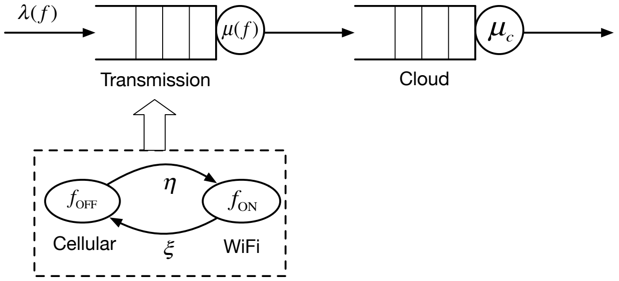


Figure 3. Uninterrupted offloading strategy with cellular and WiFi networks

From Fig. 3, the jobs are offloaded either via a cellular connection or a WiFi network to the cloud. We assume that the mean job size is $\mathbb{E}[X]$, the transmission speed of the cellular network is s_1 with service rate $\mu_1 = s_1/\mathbb{E}[X]$, and its operating power is p_1 when serving jobs and zero whenever idle. Similarly, WiFi runs at speed s_2, with service rate $\mu_2 = s_2/\mathbb{E}[X]$, and its operating power is p_2. We assume that jobs arrive according to a Poisson process with rate $\lambda(f)$, and the modulating process $f \in \{f_{\mathrm{ON}}, f_{\mathrm{OFF}}\}$ determines the transition rates.

$$\lambda(f) = \begin{cases} \lambda_1, & \text{if } f = f_{\mathrm{OFF}} \\ \lambda_2, & \text{if } f = f_{\mathrm{ON}} \end{cases} \text{ and } \mu(f) = \begin{cases} \mu_1, & \text{if } f = f_{\mathrm{OFF}} \\ \mu_2, & \text{if } f = f_{\mathrm{ON}} \end{cases}. \tag{13}$$

The original offloading jobs arrive at the system according to a Poisson process with rate λ_c, since the modulating process has two states, which results in independent Poisson processes with rate λ_i ($i \in \{1, 2\}$), we have $\lambda_1 + \lambda_2 = \lambda_c$.

It is well-known from the literature [19] that a product-form for the tandem queues between the modulated (the exponential queue) and the modulating processes exists if and only if the following condition holds:

$$\exists \rho \in R^+ \quad s.t. \, \forall f \in \{f_{\mathrm{ON}}, f_{\mathrm{OFF}}\}, \quad \frac{\lambda(f)}{\mu(f)} = \rho. \tag{14}$$

Then by substituting Eq. (13) into Eq. (14), we have:

$$\frac{\lambda_1}{\mu_1} = \frac{\lambda_2}{\mu_2}. \tag{15}$$

This is the case where the traffic intensities ρ_1 and ρ_2 are equal, though arrival intensities and service capacities need not be equal. Now this transition from one state to the second state will carry no influence on the random variable 'number of jobs present in the queue', since the traffic intensity $\lambda_i/\mu_i (= \rho)$ has not changed [20].

Figure 3 demonstrates that the uninterrupted offloading strategy is a conditional product-form model consisting of a tandem between a transmission queue (with two alternating states of cellular and WiFi) and an exponential queue representing the cloud. The former queue alternates its service by means of mutual resets according to the availability of WiFi, which is governed by an interrupted Poisson Process (IPP) with exponentially distributed ON-OFF periods. We model the intermittent availability of WiFi hotspots as a FCFS queue with occasional server break-down [7]. The queue works in mutual exclusive states and the reset occurs when the WiFi period alternates, either from ON-state to OFF-state or from OFF-state to ON-state. When WiFi becomes unavailable, the cellular network starts transmitting, otherwise when the WiFi connection becomes available, jobs are served only by the WiFi network. Specifically, offloading is not interrupted in this model, either in ON-state where the WiFi network is processing the existing jobs, or in the OFF-state during which the job is serving by the cellular network (the cellular connectivity is assumed to be always available). We assume that the sojourn time in a hotspot and the time to move from one hotspot to another are exponentially distributed with parameters ξ (failure rate), and η (recovery rate), respectively.

Since there is no waiting time before entering service, the $M/M/\infty$ queue of the cloud is occasionally referred to as a delay (sometimes pure delay) station, the probability distribution of the delay being that of the service time. Thus, the expected execution time taken in the cloud server can be calculated as $\mathbb{E}[T_c] = 1/\mu_c$. Since the application jobs are remotely executed in the cloud server rather than in the mobile device and we are only concerned with the energy consumption of the mobile device, we do not need to concern ourselves with calculating the energy consumption of the cloud server.

B. Metric-Based Analysis

We use queueing analysis to derive formulas for the average number of jobs in the $M/M/1$ queue. Given the previously stated assumptions, the uninterrupted offloading strategy can be modelled with a 2D Markov chain, as shown in Fig. 4.

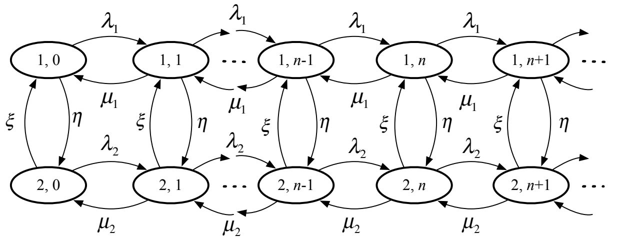


Figure 4. 2D Markov chain for the uninterrupted offloading strategy

The states with cellular network are denoted $\{1, n\}$, and the states with WiFi connectivity are denoted $\{2, n\}$. n corresponds to the number of jobs in the system (queuing and in service). Writing the balance equations for this chain gives:

$$\pi_{1,0}(\lambda_1 + \eta) = \pi_{1,1}\mu_1 + \pi_{2,0}\xi \tag{16a}$$

$$\pi_{2,0}(\lambda_2 + \xi) = \pi_{2,1}\mu_2 + \pi_{1,0}\eta \tag{16b}$$

$$\pi_{1,n}(\lambda_1 + \eta + \mu_1) = \pi_{1,n-1}\lambda_1 + \pi_{1,n+1}\mu_1 + \pi_{2,n}\xi \; (n > 0) \tag{16c}$$

$$\pi_{2,n}(\lambda_2 + \xi + \mu_2) = \pi_{2,n-1}\lambda_2 + \pi_{2,n+1}\mu_2 + \pi_{1,n}\eta \; (n > 0) \tag{16d}$$

The steady-state probability of finding the offloading system in some region with WiFi unavailability (with only cellular access) is $\pi_1 = \frac{\mathbb{E}[T_{\mathrm{OFF}}]}{\mathbb{E}[T_{\mathrm{ON}}]+\mathbb{E}[T_{\mathrm{OFF}}]} = \frac{\xi}{\eta+\xi}$. Similarly, the steady-state probability for the periods with WiFi availability is $\pi_2 = \frac{\mathbb{E}[T_{\mathrm{ON}}]}{\mathbb{E}[T_{\mathrm{ON}}]+\mathbb{E}[T_{\mathrm{OFF}}]} = \frac{\eta}{\eta+\xi}$.

Let two quantities λ^* and μ^* be defined as:

$$\lambda^* = \pi_1 \cdot \lambda_1 + \pi_2 \cdot \lambda_2, \tag{17}$$

$$\mu^* = \pi_1 \cdot \mu_1 + \pi_2 \cdot \mu_2 \,. \tag{18}$$

We define the probability generating functions for both Cellular and WiFi states as:

$$G_i(z) = \sum_{n=0}^{\infty} \pi_{i,n} z^n, \quad |z| \leq 1, \forall i = 1, 2. \tag{19}$$

After some calculation, yield:

$$(\lambda_1+\eta+\mu_1)G_1(z) = \lambda_1 z G_1(z) + \xi G_2(z) + \frac{\mu_1}{z}[G_1(z) - \pi_{1,0}] + \pi_{1,0}\mu_1, \tag{20}$$

$$(\lambda_2+\xi+\mu_2)G_2(z) = \lambda_2 z G_2(z) + \eta G_1(z) + \frac{\mu_2}{z}[G_2(z) - \pi_{2,0}] + \pi_{2,0}\mu_2. \tag{21}$$

After solving the Eqs. (20) and (21), we have:

$$g(z)G_1(z) = \pi_{2,0}\xi\mu_2 z + \pi_{1,0}\mu_1\big[\xi z + \lambda_2 z(1-z) - \mu_2(1-z)\big],$$

where $g(z) = \lambda_1\lambda_2 z^3 - (\eta\lambda_2 + \xi\lambda_1 + \lambda_1\lambda_2 + \lambda_1\mu_2 + \lambda_2\mu_1)z^2 + (\eta\mu_2 + \xi\mu_1 + \mu_1\mu_2 + \lambda_1\mu_2 + \lambda_2\mu_1)z - \mu_1\mu_2$, and it is proven that $g(z)$ has only one root z_0 in the interval $(0, 1)$ [20].

After some algebraic manipulations, we obtain:

$$\pi_{1,0} = \frac{\xi(\mu^* - \lambda^*)z_0}{\mu_1(1-z_0)(\mu_2 - \lambda_2 z_0)}, \tag{22}$$

$$\pi_{2,0} = \frac{\eta(\mu^* - \lambda^*)z_0}{\mu_2(1-z_0)(\mu_1 - \lambda_1 z_0)}. \tag{23}$$

Once the values of $\pi_{1,0}$ and $\pi_{2,0}$ have been established, the probability generating functions can be calculated as:

$$G_1(z) = \frac{\xi(\mu^* - \lambda^*)z + \pi_{1,0}\mu_1(1-z)(\lambda_2 z - \mu_2)}{g(z)}, \quad (24)$$

$$G_2(z) = \frac{\eta(\mu^* - \lambda^*)z + \pi_{2,0}\mu_2(1-z)(\lambda_1 z - \mu_1)}{g(z)}. \quad (25)$$

By using $\mathbb{E}[N_i] = \sum_{n=0}^{\infty} n\pi_{i,n} = dG_i(z)/dz|_{z=1}$, we get the average number of jobs in the system [20]:

$$\begin{aligned} \mathbb{E}[N] &= \mathbb{E}[N_1] + \mathbb{E}[N_2] \\ &= \frac{\lambda^*}{(\mu^* - \lambda^*)} + \frac{\mu_1(\mu_2 - \lambda_2)\pi_{1,0} + \mu_2(\mu_1 - \lambda_1)\pi_{2,0}}{(\xi + \eta)(\mu^* - \lambda^*)} \\ &\quad - \frac{(\mu_1 - \lambda_1)(\mu_2 - \lambda_2)}{(\xi + \eta)(\mu^* - \lambda^*)}, \end{aligned} \quad (26)$$

which contains a root of third order equation, and thus is difficult to calculate. However, when taking the balance traffic condition in Eq. (15) into account, it can be further simplified.

In such a situation, let $\mu_1/\lambda_1 = \mu_2/\lambda_2 = \theta$, and then we substitute them into $\pi_{1,0}$, $\pi_{2,0}$ and $g(z)$, it is easy to prove that $\pi_{1,0}/\pi_{2,0} = \pi_1/\pi_2$ and $g(\theta) = 0$.

Therefore, we have the decomposition:

$$g(z) = \lambda_1\lambda_2(z - \theta)(z^2 - kz + \theta), \quad (27)$$

where $k = \eta/\lambda_1 + \xi/\lambda_2 + 1 + \theta$. The root of interest z_0, which is located in the interval $(0, 1)$, is equal to $z_0 = (k - \sqrt{k^2 - 4\theta})/2$.

Finally, we get $\pi_{i,0} = \pi_i \cdot (1 - 1/\theta)$ $(\forall i = 1, 2)$, then $\rho = 1 - (\pi_{1,0} + \pi_{2,0}) = \lambda^*/\mu^*$. By using induction, it is easy to prove $\pi_{i,n} = \pi_i \cdot (1 - \rho)\rho^n$ [20].

Therefore, the partial generating functions are derived as:

$$G_i(z) = \pi_i(1-\rho)\sum_{n=0}^{\infty}(z\rho)^n = \pi_i \cdot \frac{\mu^* - \lambda^*}{\mu^* - \lambda^* z}, \quad \forall i = 1, 2, \quad (28)$$

and then by using $\mathbb{E}[N_i] = \sum_{n=0}^{\infty} n\pi_{i,n} = dG_i(z)/dz|_{z=1}$, we obtain:

$$\mathbb{E}[N_1] = \pi_1 \cdot \frac{\lambda^*}{\mu^* - \lambda^*}, \quad (29)$$

$$\mathbb{E}[N_2] = \pi_2 \cdot \frac{\lambda^*}{\mu^* - \lambda^*}. \quad (30)$$

The mean number of jobs in cloud queue is calculated as:

$$\mathbb{E}[N_c] = \frac{\lambda_c}{\mu_c}. \quad (31)$$

Since $\Pr\{e_i = 1\} = \pi_i$, according to Eq. (6), we have:

$$\Pr\{N_i > 0, e_i = 1\} = \rho_i \cdot \pi_i, \quad \forall i = 1, 2. \quad (32)$$

Further, by substituting Eqs. (29)–(32) into Eq. (12), we can formulate the explicit expressions of the ERWP metric for the uninterrupted offloading strategy.

IV. Interrupted Offloading Strategy

In this section, we discuss the interrupted offloading strategy for remote execution. First, we formulate the queueing model with cellular and WiFi networks, and then we use queueing analysis to derive the ERWP metric based on an $M/M/1$ queue and an $M/M/1$ queue with intermittent server availability.

A. Problem Formulation

The interface selection problem in offloading systems is modelled as the decision to which queue an arriving job should be assigned. In this decision the offloading dispatcher takes both performance and energy costs into account. This seems natural for heterogeneous servers where a job needs a different amount of energy and time to be served by different servers. For example, whereas assigning each job to a low power server would be beneficial from the energy consumption perspective at low loads, such a strategy may end up in difficulties at higher loads as the response time increases rapidly [21]. Indeed, the energy-performance tradeoff takes on different explicit expressions under different offloading strategies.

As shown in Fig. 5, we consider a queuing model that consists of two parallel queues of cellular and WiFi with work conserving queuing disciplines. λ_1 and λ_2 are the mean rates of jobs into Queue 1 and Queue 2. Since the original offloading jobs arrive at the system according to a Poisson process with rate λ_c, one strategy would be random assignment into Queue i $(i \in \{1, 2\})$, which results in independent Poisson processes with rate λ_i, and we have $\lambda_1 + \lambda_2 = \lambda_c$.

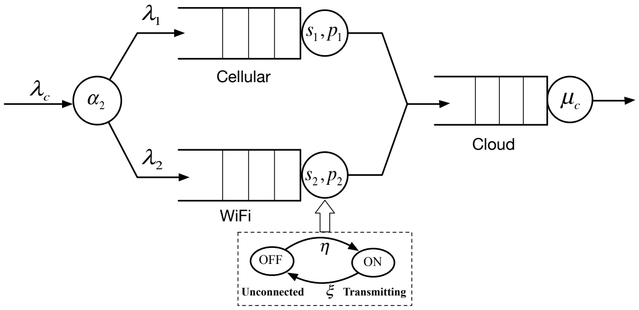


Figure 5. Interrupted offloading strategy with cellular and WiFi networks

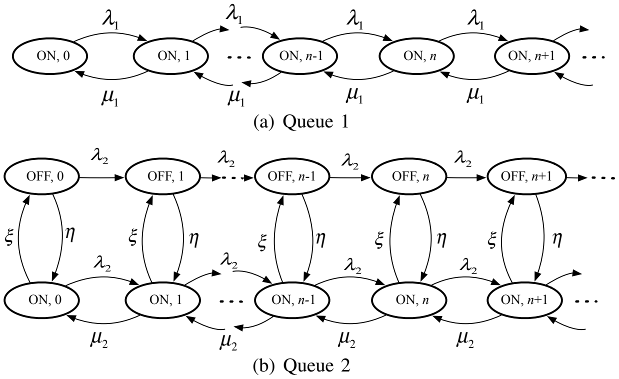
The two queues have the following behavior:

- **Queue 1**: When a job is offloaded to the cloud via a cellular network, there is queueing due to the transmission speed of the cellular link. Costs arise in terms of transmission delays (queueing and actual transmission time) and transmission energy consumption. Server 1 is always available since the cellular connection is always on.
- **Queue 2**: When a job is offloaded to the cloud via a WiFi network, there is queueing due to the transmission speed of the WiFi link. We model the intermittent availability of hotspots as a FCFS queue with occasional server breakdown. The availability of Server 2 is governed by an IPP with exponentially distributed ON-OFF periods. Specifically, the server is either in ON-state processing the existing jobs, or in OFF-state during which no job receives service. We can assume that the service station fails from time to time and resumes its operation after a random time. When a server recovers, it continues to serve the customer whose service has been interrupted, i.e. the work already completed is not lost (cf. data transfers with resume).

Similarly, the cloud queue is a pure delay station at which jobs spend an exponentially distributed amount of time with mean equal to $1/\mu_c$ time units.

B. Metric-Based Analysis

The Markov chains for the interrupted offloading strategy in Fig. 5 can be decomposed into two separate chains, as shown in Fig. 6.



(a) Queue 1

(b) Queue 2

Figure 6. Markov chains for interrupted offloading strategy with cellular and WiFi networks

1) Queue 1: the Markov chain is depicted in Fig. 6(a), and the expected number of jobs can be calculated as:

$$\mathbb{E}[N_1] = \frac{\rho_1}{1-\rho_1}, \tag{33}$$

where the workload for Queue 1 during the busy period is $\rho_1 = \lambda_1/\mu_1 = \lambda_1 \mathbb{E}[X]/s_1$.

2) Queue 2: refers to offloading jobs from the mobile device to the cloud via a WiFi network, which is modelled as an $M/M/1$-FCFS queue with intermittently available service. The Markov chain is depicted in Fig. 6(b), which is equivalent to assuming that $\lambda_1 = \lambda_2$ and $\mu_1 = 0$ in Fig. 4.

Writing the balance equations for this chain gives:

$$\pi_{\text{OFF},0}(\lambda_2 + \eta) = \pi_{\text{ON},0}\xi \tag{34a}$$

$$\pi_{\text{ON},0}(\lambda_2 + \xi) = \pi_{\text{OFF},0}\eta + \pi_{\text{ON},1}\mu_2 \tag{34b}$$

$$\pi_{\text{OFF},n}(\lambda_2 + \eta) = \pi_{\text{OFF},n-1}\lambda_2 + \pi_{\text{ON},n}\xi \tag{34c}$$

$$\pi_{\text{ON},n}(\lambda_2 + \xi + \mu_2) = \pi_{\text{ON},n-1}\lambda_2 + \pi_{\text{ON},n+1}\mu_2 + \pi_{\text{OFF},n}\eta \tag{34d}$$

After simple algebraic operations of equations in Eq. (34) yields:

$$(\pi_{\text{OFF},n} + \pi_{\text{ON},n}) \cdot \lambda_2 = \pi_{\text{ON},n+1}\mu_2 \ (n = 0, 1, 2, \dots). \tag{35}$$

Summation of Eq. (35) over all n, we obtain:

$$\pi_{\text{ON},0} = \pi_{\text{ON}} - \frac{\lambda_2}{\mu_2}, \tag{36}$$

where $\pi_{\text{ON},0} = \pi_{2,0}$, $\pi_{\text{ON}} = \pi_2$ and $\pi_{\text{OFF}} = \pi_1$.

According to Eqs. (17) and (18), we have $\lambda^* = \lambda_2$ and $\mu^* = \pi_2\mu_2$. After substituting the above values in Eq. (26), we derive the mean number of jobs in Queue 2 as:

$$\begin{aligned}\mathbb{E}[N_2] &= \mathbb{E}[N_{\text{OFF}}] + \mathbb{E}[N_{\text{ON}}] \\ &= \frac{\lambda^*}{(\mu^* - \lambda^*)} + \frac{\mu_2\lambda_2\pi_{2,0} - (-\lambda_2)(\mu_2 - \lambda_2)}{(\xi + \eta)(\mu^* - \lambda^*)} \\ &= \frac{\lambda_2}{\pi_2\mu_2 - \lambda_2} + \frac{\pi_1\lambda_2\mu_2}{(\pi_2\mu_2 - \lambda_2)(\xi + \eta)}. \end{aligned} \tag{37}$$

3) Optimization: Since Server 1 is always available, we have $\Pr\{e_1 = 1\} = 1$ and the fraction of time that Server 2 is available to process jobs is: $\Pr\{e_2 = 1\} = \frac{\eta}{\xi+\eta} = \pi_2$, where as the recovery rate $\eta \to \infty$, the availability of Server 2 tends to be 1. Then we have:

$$\Pr\{N_1 > 0, e_1 = 1\} = \rho_1, \tag{38}$$

$$\Pr\{N_2 > 0, e_2 = 1\} = \rho_2 \cdot \pi_2. \tag{39}$$

Further, by substituting Eqs. (33), (37)–(39) into Eq. (12), we can formulate the optimization of the ERWP metric for the offloading assignment as:

$$\lambda_i^{\min} = \arg\min_{\lambda_i} ERWP. \tag{40}$$

We seek the arrival rate $\lambda_i^{\min}$ to Queue i such that ERWP is minimised when both queues are in operation. We can apply Newton's method to find $\lambda_i^{\min}$ iteratively [22].

In other words, arriving jobs are assigned to Queue 1 and Queue 2 according to the optimized objective defined in Eq. (40), minimizing the ERWP metric.

V. Performance Evaluation

A. A Realistic Offloading Scenario

Different wireless network interfaces vary in many ways, which we have to capture in simplified form in just few parameters. Cellular networks such as EDGE and 3G, usually have much higher availability than WiFi, but the transmission rate of WiFi is higher. Besides, the WiFi interface is more energy-efficient than the cellular interface [8]. These imply that WiFi is much faster and more energy-efficient than the cellular interface for transmitting the same quantity of data. Therefore, we consider here a simple scenario where the transmission rate of the cellular network is smaller than that of WiFi, i.e. $s_1 < s_2$ and the power consumption when transmitting jobs via the cellular link is larger than the WiFi link, i.e. $p_1 > p_2$.

Using measurements from real traces in [15], the average data rates for the cellular network and WiFi are set as $s_1 = 800$ Kbps and $s_2 = 2$ Mbps, respectively. The mean job size is 125 KB. According to the power models developed in [23], we set the power coefficients $p_1 = 2.5$ W, $p_2 = 0.7$ W and $p_m = 2$ W, respectively. Besides, suppose that the total job arrival rate for offloading is $\lambda = 0.6$ packet/s, the mobile service rate $\mu_m = 2$ tasks/s, the cloud service rate $\mu_c = 5$ tasks/s and both the failure rate ξ and recovery rate η of Server 2 are equal to 1.

B. Performance Analysis

We first analyse the case when the offloading probability $\pi = 0.5$, indicating that half of the offloadable jobs are offloaded to the cloud, while the rest are executed on the mobile device.

From Fig. 7, we can observe that the uninterrupted offloading strategy performs significantly better than the interrupted one when ω is small, but as ω approaches to 1, the interrupted strategy performs much better. This means that when considering energy consumption more important than response time (for delay-tolerance applications), it is better to use the interrupted strategy; otherwise when considering response time more important (for delay-sensitive applications), the uninterrupted strategy is much more preferred, which fully uses the unavailable periods of WiFi by transmitting with a cellular network. Since energy

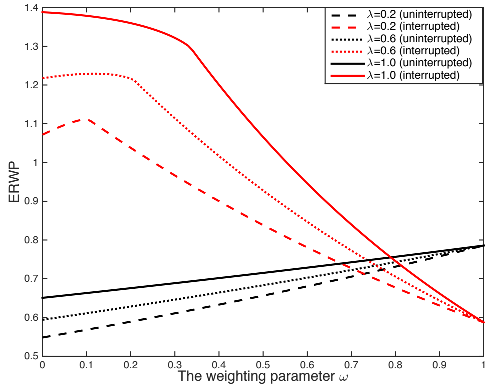


Figure 7. Comparison of two offloading strategies under arrival rate λ

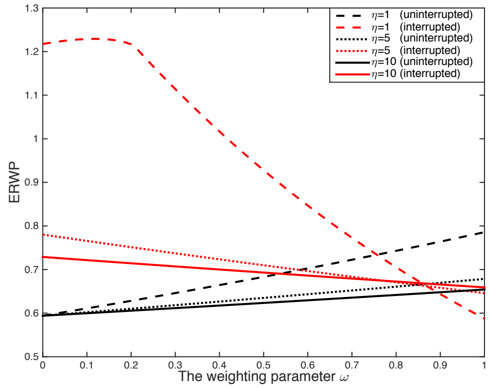


Figure 8. Comparison of two offloading strategies under recovery rate η

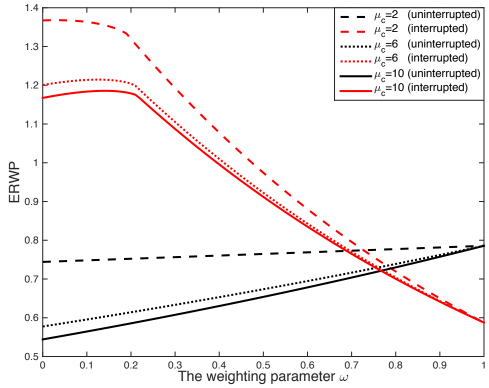


Figure 9. Comparison of two offloading strategies under cloud service rate μ_c

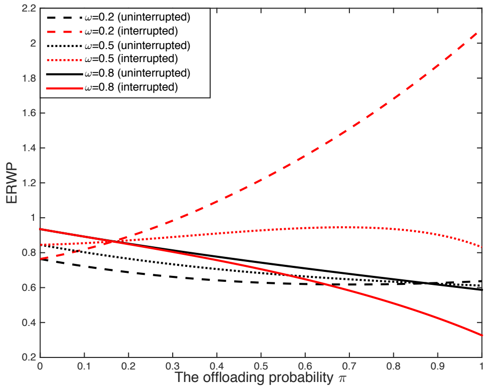


Figure 10. Comparison of two offloading strategies under ω parameter

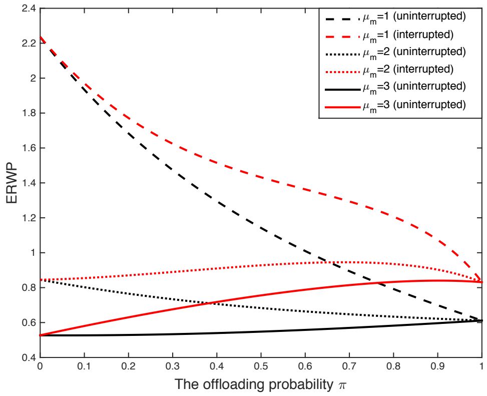
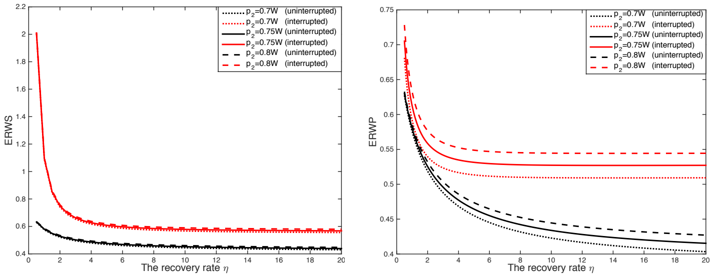


Figure 11. Comparison of two offloading strategies under different mobile service rate μ_m



(a) ERWS

(b) ERWP

Figure 12. Comparison of of two offloading strategies based on ERWS and ERWP metrics

is measured per job, when the weight is on energy consumption only the metric is for both strategies insensitive to the job arrival rate. As the arrival rate of the offloadable jobs λ increases, none of the offloading strategies can achieve a low ERWP value. However, the uninterrupted strategy varies less. The interrupted strategy is more sensitive to the job arrival rate.

Similar observations can be made from Fig. 8 or sensitivity to the recovery rate η. The interrupted strategy suffers more from long repair times than the uninterrupted strategy. This is reasonable, due to the lower WiFi availability, resulting in most of the jobs being offloaded through the slower and more energy consuming cellular network interface. When $\omega < 0.85$, (i.e. response time is more important) the interrupted strategy also performs much better as η increases. The reason is that arriving jobs to the WiFi queue have a higher probability to be offloaded to the cloud. However, with more importance being given to the energy consumption, this strategy performs much worse as η increases and the down-times become less.

In Fig. 9, it is observed that the faster the cloud serves, the lower the ERWP value is. The cloud service rate μ_c has a great influence on the mean response time since we have to wait for the cloud service to be finished. But the cloud service rate has little influence on the energy consumption since the jobs are processed remotely rather than locally on the mobile device.

Then, we dynamically change the offloading probability π to find the optimal offloading decision.

As shown in Fig. 10, when ω is small ($\omega = 0.2$), it is not worth offloading any job in the interrupted strategy, while it is better to offload half of the offloadable jobs to the cloud in the uninterrupted strategy. However, with more focus on the energy consumption (ω approaches to 1), it is better to offload all the jobs for both strategies; meanwhile the interrupted offloading strategy obtains lower values for the metrics than the uninterrupted one.

The ERWP metric can be treated as the ERP metric when we set the weighting parameter $\omega = 0.5$, i.e. when both the energy consumption and the response time have the equal importance. From Fig. 11, it is observed that the uninterrupted strategy should always be preferred. When the mobile service rate μ_m is very small, it is worth offloading all the jobs to the cloud with both offloading strategies. Since the local execution is very slow it is beneficial to offload all jobs. However, when μ_m reaches some value, it is better not to offload since the cost saved from remote execution is not enough to cover the extra cost for offloading.

In Fig. 12 we compare the ERWP metric with the ERWS metric. When the power p_2 changes slightly, e.g. from 0.7 W to 0.8 W, it is difficult to tell the difference between the two offloading strategies according to the ERWS metric depicted in Fig. 12(a), while it is very clear to see the difference according to the ERWP metric shown in Fig. 12(b). It seems that the ERWP metric is much more sensitive to the large scale and can capture small changes when the ERWS metric has the disadvantage of a linear combination of two criteria on different scales.

VI. Conclusions

In this paper, we have developed two offloading strategies to leverage the complementary strength of WiFi and cellular networks. We have formulated queueing models to carry out optimality analysis of the energy-performance tradeoff for mobile cloud offloading systems based on the ERWP metric, which captures both energy and performance metrics and also intermittently available access links. The ERWP metric combines the advantages of both an additive metric (ERWS) and a product metric (ERP), i.e. it not only assigns different importance weights to energy consumption and response time, but also is insensitive to criteria on different scales.

We find that for delay-tolerant applications, it is better to use the interrupted strategy instead of the uninterrupted one, while for delay-sensitive applications, the uninterrupted strategy shows very good results and outperforms the interrupted offloading one by a significant margin. The offloading probability closely depends on the mobile service rate, the cloud service rate and the weighting parameter. Slow mobile service rate and fast cloud service rate will result in more jobs to be offloaded to the cloud. We can thus judiciously make the offloading decisions of whether to offload or not and how much to offload that optimise the ERWP metric.

The proposed queueing model can be used to describe complex and realistic offloading systems. So far, the assumption that the WiFi data rate and power consumption remain constant in all the regions within WiFi coverage is somehow unrealistic. Therefore, it is worth considering scenarios where they might be different at each connected access point in the future.

References

[1] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: architecture, applications, and approaches," *Wireless communications and mobile computing*, vol. 13, no. 18, pp. 1587–1611, 2013.

[2] M.-R. Ra, J. Paek, A. B. Sharma, R. Govindan, M. H. Krieger, and M. J. Neely, "Energy-delay tradeoffs in smartphone applications," in *Proceedings of the 8th international conference on Mobile systems, applications, and services*, pp. 255–270, ACM, 2010.

[3] H. Wu, Q. Wang, and K. Wolter, "Tradeoff between performance improvement and energy saving in mobile cloud offloading systems," in *Communications Workshops (ICC), 2013 IEEE International Conference on*, pp. 728–732, IEEE, 2013.

[4] H. Wu and K. Wolter, "Dynamic transmission scheduling and link selection in mobile cloud computing," in *Analytical and Stochastic Modeling Techniques and Applications*, pp. 61–79, Springer, 2014.

[5] H. Wu and K. Wolter, "Tradeoff analysis for mobile cloud offloading based on an additive energy-performance metric," in *Performance Evaluation Methodologies and Tools (VALUETOOLS), 2014 8th International Conference on*, ACM, 2014.

[6] A. Rahmati and L. Zhong, "Context-for-wireless: context-sensitive energy-efficient wireless data transfer," in *Proceedings of the 5th international conference on Mobile systems, applications and services*, pp. 165–178, ACM, 2007.

[7] E. Hyytiä, T. Spyropoulos, and J. Ott, "Offload (only) the right jobs: Robust offloading using the markov decision processes," in *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, (Boston, MA, USA), Jun. 2015. to appear.

[8] P. Shu, F. Liu, H. Jin, M. Chen, F. Wen, Y. Qu, and B. Li, "etime: energy-efficient transmission between cloud and mobile devices," in *INFOCOM, 2013 Proceedings IEEE*, pp. 195–199, IEEE, 2013.

[9] F. Mehmeti and T. Spyropoulos, "Performance analysis of "on-the-spot" mobile data offloading," in *Global Communications Conference (GLOBECOM), 2013 IEEE*, pp. 1577–1583, IEEE, 2013.

[10] E. Cuervo, A. Balasubramanian, D.-k. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "Maui: making smartphones last longer with code offload," in *Proceedings of the 8th international conference on Mobile systems, applications, and services*, pp. 49–62, ACM, 2010.

[11] K. Kumar, J. Liu, Y.-H. Lu, and B. Bhargava, "A survey of computation offloading for mobile systems," *Mobile Networks and Applications*, vol. 18, no. 1, pp. 129–140, 2013.

[12] A. Papoulis and S. U. Pillai, *Probability, random variables, and stochastic processes*. Tata McGraw-Hill Education, 2002.

[13] V. Cardellini, V. De Nito Personé, V. Di Valerio, F. Facchinei, V. Grassi, F. Lo Presti, and V. Piccialli, "A game-theoretic approach to computation offloading in mobile cloud computing," tech. rep., Technical Report, available online at http://www. optimizationonline. org/DB HTML/2013/08/3981. html, 2013.

[14] A. Wierman, L. L. Andrew, and A. Tang, "Power-aware speed scaling in processor sharing systems," in *INFOCOM 2009, IEEE*, pp. 2007–2015, IEEE, 2009.

[15] K. Lee, J. Lee, Y. Yi, I. Rhee, and S. Chong, "Mobile data offloading: How much can wifi deliver?," *Networking, IEEE/ACM Transactions on*, vol. 21, no. 2, pp. 536–550, 2013.

[16] Y.-W. Kwon and E. Tilevich, "Energy-efficient and fault-tolerant distributed mobile execution," in *Distributed Computing Systems (ICDCS), 2012 IEEE 32nd International Conference on*, pp. 586–595, IEEE, 2012.

[17] A. Gandhi, V. Gupta, M. Harchol-Balter, and M. A. Kozuch, "Optimality analysis of energy-performance trade-off for server farm management," *Performance Evaluation*, vol. 67, no. 11, pp. 1155–1171, 2010.

[18] S. Balsamo, G.-L. dei Rossi, and A. Marin, "Queueing networks and conditional product-forms," in *Proceedings of the 7th International Conference on Performance Evaluation Methodologies and Tools*, pp. 204–213, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2013.

[19] J.-M. Fourneau, B. Plateau, and W. Stewart, "Product form for stochastic automata networks," in *Proceedings of the 2nd international conference on Performance evaluation methodologies and tools*, pp. 32–, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007.

[20] U. Yechiali and P. Naor, "Queuing problems with heterogeneous arrivals and service," *Operations Research*, vol. 19, no. 3, pp. 722–734, 1971.

[21] A. Penttinen, E. Hyytiä, and S. Aalto, "Energy-aware dispatching in parallel queues with on-off energy consumption," in *Performance Computing and Communications Conference (IPCCC), 2011 IEEE 30th International*, pp. 1–8, IEEE, 2011.

[22] I. Tsimashenka and W. J. Knottenbelt, "Trading off subtask dispersion and response time in split-merge systems," in *Analytical and Stochastic Modeling Techniques and Applications*, pp. 431–442, Springer, 2013.

[23] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani, "Energy consumption in mobile phones: a measurement study and implications for network applications," in *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*, pp. 280–293, ACM, 2009.