Energy-Optimal Execution Policy for A Cloud-Assisted Mobile Application Platform

Yonggang Wen[a], Weiwen Zhang[a], Kyle Guan[b], Dan Kilper[b], Haiyun Luo[c]

[a] *School of Computer Engineering, Nanyang Technological Unviersity, Nanyang Avenue, Singapore 639798*
Email: {ygwen, wzhang9}@ntu.edu.sg
[b] *Bell Laboratories, Alcatel-Lucent, Holmdel, NJ 07733*
Email: {kyle.guan, dan.kilper}@alcatel-lucent.com
[c] *US Research Center, China Mobile, Milpitas, CA 95035*
Email: {haiyunluo}@chinamobile.com

Abstract

In this paper, we derive strategies for the energy-optimal execution decision on a cloud-assisted mobile application platform. Specifically, in the platform, mobile applications can either be executed in the mobile device (i.e., mobile execution) or offloaded to the cloud clone for execution (i.e., cloud execution), with an objective to conserve energy for the mobile device. The design trade-off is between the computation energy for the mobile execution and the transmission energy for the cloud execution. The optimal execution policy can be identified by solving two energy-aware scheduling problems. The first one is to minimize the computation energy to complete all CPU cycles required by the application, by optimally configuring the clock frequency in the mobile device. The second one is to minimize the transmission energy of transferring all the data within a time deadline, by optimally scheduling the data rate over a stochastic wireless channel. We formulate both problems as constrained optimization problems, and obtain closed-form solutions for the optimal scheduling policy and the minimum energy consumed in both cases. Further theoretical analysis for both execution modes indicates that the optimal policy depends on not only the application profile (i.e., the data volume and the delay deadline), but also the wireless transmission model (i.e., the monomial order for the energy consumption model). For the mobile execution, the optimal energy scales cubically with the data transmission size, while for the cloud execution the monomial order of the transmission model has significant influence on the energy consumption. These analytical results enable us to determine the optimal condition under which the mobile execution or the cloud execution is more energy-

efficient for the mobile device. Moreover, numerical results suggest that a significant amount of energy can be saved by optimally offloading the mobile application to the cloud clone for execution.

Keywords:
Energy-Optimal Execution, Mobile Application, Cloud Computing, Lagrangian Multiplier Method

1. Introduction

The tension between resource-hungry applications and resource-poor mobile devices is considered as one of the driving forces for the evolution of mobile platforms. Due to the limited physical size, mobile devices are inherently resource-constrained [1], equipped with a limited supply of resources in computation, energy, bandwidth and storage. In particular, the energy supply from the limited battery capacity [2] has been one of the most challenging design issues for mobile devices. Indeed, the limited battery life has been found by market research as the biggest complaint for smart phones [3]. Therefore, design decisions for mobile applications have to take consideration of the resource limitations in the mobile devices.

Emerging cloud-computing technology[4], owing to the nature of elastic resource pooling, offers an opportunity to extend the capabilities of mobile devices for energy-hungry applications. Various cloud-assisted mobile platforms have been proposed, such as Cloudlet [5] and Cloud Clone [6]. In these proposed platforms, each mobile device is associated with a system-level clone in the cloud infrastructure. The mobile clone, which runs on a virtual machine (VM), can execute mobile applications on behalf of the mobile device - this is commonly referred as *application offloading*. This architecture requires both a mechanism to implement task offloading and a policy to decide when to offload applications. On one hand, existing research [5, 6, 8, 9] has proposed various architectures and mechanisms of offloading applications to the cloud. On the other hand, the research on optimal energy policies for application offloading to cloud execution is rather inadequate (cf. Section 6 on related work).

We illustrate a generic architecture of the cloud-assisted mobile application platform in Figure 1. Each mobile device is replicated by a system-level clone that runs on a virtual machine (VM). The VM is located in a nearby cloud infrastructure and can migrate in response to the user's location. Moreover, the mobile clone regularly synchronizes its state with the physical device. With the scheme of application offloading, the mobile

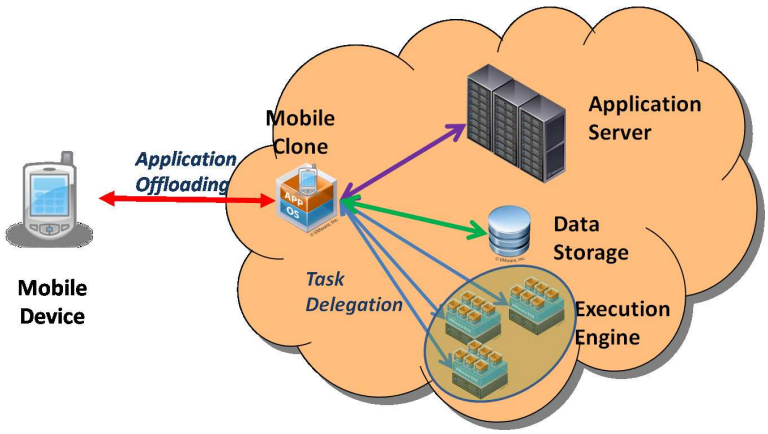


Figure 1: A cloud-assisted mobile application platform: the mobile device is cloned by a system-level virtual machine, which extends the capabilities of the mobile devices via different functionalities, including application offloading, task delegation, data staging and mobile P2P.

clone not only provides computing and storage in its local VM environment, but also harnesses computing and storage resources from a remote cloud, denoted as task delegation in Figure 1.

On this platform, a mobile application can be executed either on the mobile device (i.e., *mobile execution*) or on the cloud clone (i.e., *cloud execution*). The design objective is to develop an optimal application-execution policy, minimizing the energy consumed by the mobile device. When the application is executed in the mobile device, the computation energy can be minimized by optimally scheduling the clock frequency of the mobile device via the Dynamic Voltage Scaling (DVS) technology [10]. When the application is executed in the cloud clone, the transmission energy can be minimized by optimally scheduling the transmission data rate in a stochastic wireless channel. For both scheduling problems, we formulate them as convex optimization problems, with a constraint that the application should be completed within a time deadline. We solve both optimization problems analytically and obtain closed-form solutions for the optimal scheduler and the minimum energy consumed by the mobile device. Our analytical solutions are applied to decide the optimal condition for energy-efficient application execution.

The rest of this paper is organized as follows. In Section 2, we present a model for energy consumption in the mobile execution and the cloud execution. In Section 3 and 4, we solve the optimization problems for the optimal

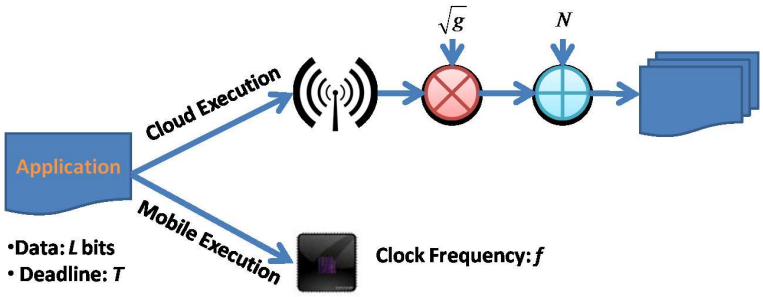


Figure 2: Mobile application executed in two alternative modes: the mobile execution and the cloud execution.

CPU clock-frequency scheduling in the mobile execution and the optimal transmission data-rate scheduling in the cloud execution. Closed-forms solutions are derived for both optimization problems. In Section 5, analytical results from previous two sections are applied to develop optimal execution strategies for mobile applications. In Section 6, the review of related work is presented. Section 7 summarizes this paper and provides future directions.

2. System Model and Problem Formulation

In this section, we present a mathematical model for application execution on the cloud-assisted mobile application platform. First, we define a mobile application profile. Following that, we introduce an energy consumption model for application execution, including a computation energy model for the mobile execution and a transmission energy model for the cloud execution.

2.1. Mobile Application Model

A mobile application is characterized by two parameters, including:

- Input data size L: the number of data bits as the input to the application;
- Application completion deadline T: the delay deadline before which the application should be completed.

We denote the application profile as $A(L, T)$.

In this research, we are interested in the problem of energy-optimal application execution. The mobile execution and cloud execution impose different energy consumption on the mobile device, which will be detailed in the next two subsections.

2.2. Mobile Execution Energy Model

When the application is executed on the mobile device, the energy consumption is determined by CPU workload. The workload is measured by the number of CPU cycles required by the application, denoted as W, which depends on the input data size and the algorithm in the application. Typically, W is modeled as a random variable, which we elaborate on and describe in Section 3.

For the mobile execution, its computation energy can be minimized by optimally configuring the clock frequency of the chip, via the dynamic voltage scaling (DVS) technology [10]. In CMOS circuits [11], the energy per operation $\mathcal{E}_{op}$ is proportional to V^2, where V is the supply voltage to the chip. Moreover, it has been observed that, when operating at low voltage limits, the clock frequency of the chip, f, is approximately linear proportional to the voltage supply, V [11]. As a result, the energy per operation can be expressed as,

$$\mathcal{E}_{op} = \kappa f^2, \tag{1}$$

where κ is the effective switched capacitance depending on the chip architecture. Note that a CPU can reduce its energy consumption substantially by running more slowly. However, the application has to meet a delay deadline of T, which suggests that the clock frequency cannot remain low. As such, one would like to configure the clock frequency to minimize the total energy consumption, while meeting the application delay deadline. The optimization problem can be formulated as,

$$\mathcal{E}_m^* = \min_{\psi \in \Psi} \{\mathcal{E}_m(L, T, \psi)\}, \tag{2}$$

where $\psi = \{f_1, f_2, ...f_W\}$ is any clock-frequency vector that meets the delay deadline, Ψ is the set of all feasible clock-frequency vectors, and $\mathcal{E}_m(L, T, \psi)$ is the total energy consumed by the mobile device. This optimization problem will be solved in Section 3.

2.3. Cloud Execution Energy Model

When the application is executed by the cloud clone, the energy consumed by the mobile device depends on the amount of data to be transmitted from the mobile device to the cloud clone and the wireless channel model. For any mobile application $A(L, T)$, L bits of data needs to be transmitted to the cloud clone. Note that we assume the binary executable file for the application has been replicated on the cloud clone initially. As such, it does not incur additional energy cost. We assume a stochastic fading model

for the wireless channel between the mobile device and the cloud clone. As illustrated in Figure 2, it is characterized by a channel gain of g and a noise power of N. Specific models (i.e., an i.i.d model and a Gilbert-Elliott model) for the channel gain will be presented in Section 4.1.

In this research, we adopt an empirical transmission energy model as in [12, 13, 14, 15]. Specifically, for a wireless fading channel with a gain of g, the energy consumed to transfer s bits of data over the channel within a time slot is governed by a convex monomial function, i.e.,

$$\mathcal{E}_t(s, g, n) = \lambda \frac{s^n}{g}, \tag{3}$$

where n denotes the monomial order, and λ denotes the energy coefficient. It has been shown that some practical modulation scheme exhibits an energy-bit relation that can be well approximated by a monomial. It is normally assumed that $2 \leq n \leq 5$, depending on the modulation scheme.

In the cloud execution, it is possible to minimize the total transmission energy by optimally varying the data rate (the number of transmitted bits in a given time slot), in response to a stochastic channel. Since the energy cost per time slot is a convex function of bits transmitted, it is ideal to transmit as few bits as possible [25]. However, reducing the number of bits transmitted per time slot increases the total delay for the application. Therefore, there exists an optimal transmission data-rate schedule to minimize the total transmission energy, while satisfying the delay requirement. Under the optimal transmission scheduling, the minimum amount of transmission energy for the cloud execution is given by

$$\mathcal{E}_c^* = \min_{\phi \in \Phi} \mathbb{E}\{\mathcal{E}_c(L, T, \phi)\}, \tag{4}$$

where $\phi = \{s_1, s_2, ...s_T\}$ denotes a data transmission schedule (s_i for the number of bits transmitted in time slot i) that meets the delay deadline (T time slots), Φ is the set of all feasible data schedules, and $\mathcal{E}_c(L, T, \phi)$ denotes the transmission energy. It should be noted that the expectation of energy consumption is taken for different channel states. This optimization problem will be solved in Section 4.

2.4. Optimal Application Execution Policy

The decision for energy-optimal application execution, is to choose where to execute the application, with an objective to minimize the total energy

consumed on the mobile device. Specifically, the optimal policy is determined by the following decision rule,

$$\begin{cases} \text{Mobile Execution} & \text{if} \quad \mathcal{E}_m^* \leq \mathcal{E}_c^* \\ \text{Cloud Execution} & \text{if} \quad \mathcal{E}_m^* > \mathcal{E}_c^*. \end{cases} \tag{5}$$

As shown in Eq. (1) and Eq. (3), $\mathcal{E}_m^*$ is proportional to κ and $\mathcal{E}_c^*$ is proportional to λ. Hence, the absolute values of κ and λ are not critical, but the ratio between these two constant energy coefficients, κ/λ, could affect the determination of the optimal execution policy.

Moreover, as specified in Eq. (2) and Eq. (4), the optimal clock-frequency vector ψ and data transmission scheduling vector ϕ have critical effects on the energy consumption of mobile execution and cloud execution, respectively. In order to decide an optimal application execution strategy, we will first solve these two optimization problems to find the optimal scheduling vectors.

3. Optimal Computation Energy under Mobile Execution

In this section, we investigate the problem of minimizing the computation energy for executing an application in the mobile device, by optimally setting the clock frequency of the chip.

3.1. Probabilistic Task Execution in Mobile Device

Let W indicate the number of CPU cycles needed for an application. For a given input data size, L, it can be expressed as [2]

$$W = LX, \tag{6}$$

where X has been shown to be a random variable with an empirical distribution[16]. The estimation of this distribution, which depends on the nature of the application, has been treated in [17, 18], and is thus beyond the scope of this paper. In this paper, we assume that the probability distribution function (PDF) of X is $P(x)$, and its cumulative distribution function (CDF) is defined as

$$F_X(x) = \Pr[X \leq x], \tag{7}$$

and its complementary cumulative distribution function (CCDF), denoted as $F_X^c(w)$, is defined as

$$F_X^c(x) = 1 - F_X(x). \tag{8}$$

Therefore, the CDF of the workload W is given by $F_W(w) = F_X(w/L)$, and its CCDF is given by $F_W^c(w) = F_X^c(w/L)$.

As shown in [16, 17, 18], the number of CPU cycles per bit can be modeled by a Gamma distribution. The PDF of the Gamma distribution is given by

$$p_X(x) = \frac{1}{\beta\Gamma(\alpha)}(\frac{x}{\beta})^{\alpha-1}e^{-\frac{x}{\beta}}, \quad \text{for} \quad x > 0, \tag{9}$$

which depends on two parameters(the shape α and the scale β).

In this paper, we adopt a probabilistic performance requirement. We assume that the jobs should satisfy the soft real-time requirements. Specifically, each application will meet its deadline with a probably of ρ by allocating W_ρ CPU cycles. The parameter ρ is called the application completion probability (ACP). When the application execution fails to meet its deadline, it will continue to execute at the maximum clock frequency to completion. The additional computation energy is negligible when the task completion probability is very close to 1. As a result, we focus on $\mathcal{E}_\rho$ under the assumption that the task completion probability is close to 1.

The probability that each job requires no more than the allocated W_ρ cycles is at least ρ, i.e.,

$$F_W(W_\rho) = \Pr[W \leq W_\rho] \geq \rho. \tag{10}$$

Using Eq. (7), we can obtain the number of CPU cycles, for a given ρ, as

$$W_\rho = F_W^{-1}(\rho) = LF_X^{-1}(\rho), \tag{11}$$

which is the ρ^{th} quantile for the distribution of W.

3.2. Energy-Efficient Clock-Frequency Configuration

In this subsection, we aim to minimize the expected energy consumption of the application execution, by optimally setting the clock frequency of the mobile device. Specifically, for each application, the problem is to find a clock frequency scheduling for each of its allocated cycles, such that the total computation energy for these allocated cycles is minimized while their total execution time is less than the application deadline.

We assume that $f(w)$ is a clock-frequency schedule vector, where w is the number of CPU cycles it has completed previously. Therefore, the energy consumption is given by

$$\mathcal{E}_m = \kappa \sum_{w=1}^{W_\rho} F_W^c(w)[f(w)]^2, \tag{12}$$

where $F_X^c(w)$ is the probability that the application has not completed after w CPU cycles. The optimization problem in Eq. (2) can be rewritten as,

$$\min_{f(w)} \quad \kappa \sum_{w=1}^{W_\rho} F_W^c(w)[f(w)]^2, \tag{13}$$

$$\text{s.t.} \quad \sum_{w=1}^{W_\rho} \frac{1}{f(w)} \leq T, \tag{14}$$

$$f(w) > 0 \tag{15}$$

where Eq. (14) corresponds to the delay constraint.

The optimization problem, denoted in Eq. (13) can be solved analytically. The results are summarized in Theorem 3.1.

Theorem 3.1. *For the optimal CPU scheduling problem in Eq.* (13)*, the optimal clock scheduling vector is given by*

$$f^*(w) = \frac{\theta}{T[F_W^c(w)]^{1/3}}, \quad 1 \leq w \leq W_\rho, \tag{16}$$

where $\theta = \sum_{i=1}^{W_\rho} [F_W^c(i)]^{1/3}$*. The optimal computation energy is*

$$\mathcal{E}_m^* = \frac{\kappa}{T^2}\{\sum_{w=1}^{W_\rho} [F_W^c(w)]^{1/3}\}^3. \tag{17}$$

Proof 3.1. *See Appendix A.*

Proposition 3.1. *For a task load with an exponentially-tailed CCDF (i.e.,* $F^c(w) \sim \mu e^{-\nu w}$ *as* $w \to \infty$ *for some constant* $\mu > 0$ *and* $\nu > 0$*), the minimum energy consumption converges monotonically to a finite value, as the target completion probability increases to 1.*

Proof 3.2. *See Appendix B.*

In Figure 3, we plot the minimum computation energy, $\mathcal{E}_m^*$, as a function of the target completion probability, ρ. Notice that the Gamma distribution is exponentially tailed. As a result, it can been that, as the application completion probability of ρ increases, the minimum computation energy increases monotonically and converges to a finite value.

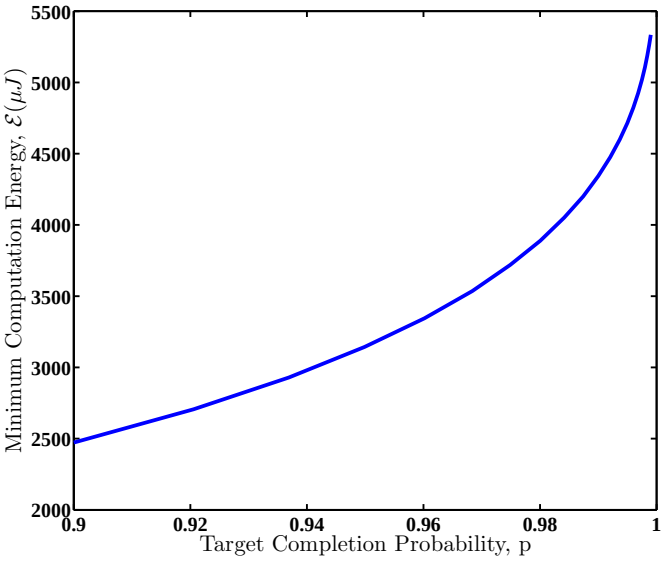


Figure 3: The minimum computation energy, $\mathcal{E}_m^*$, is plotted as a function of the target completion probability. In this graph, the task load is modeled as the Gamma distribution, with $\alpha = 4$, $\beta = 200$, $L = 800bits$, $T = 50ms$.

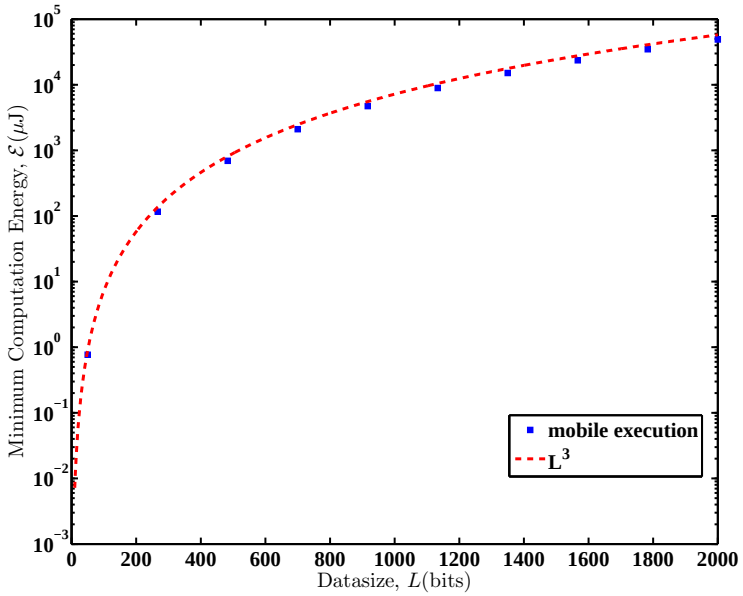


Figure 4: The minimum computation energy, $\mathcal{E}_m^*$, is plotted as a function of the input data size L. In this graph, the task load is modeled as the Gamma distribution, with $\alpha = 4$, $\beta = 200$, $T = 50ms$.

Proposition 3.2. *For the optimal CPU scheduling in Eq.* (17)*, the optimal computation energy is proportional to negative quadratic of the delay deadline.*

$$\mathcal{E}_m^* \sim T^{-2}. \tag{18}$$

Proposition 3.3. *For the optimal CPU scheduling problem in Eq.* (17)*, the optimal computation energy is proportional to cube of the data size.*

$$\mathcal{E}_m^* \sim L^3 \tag{19}$$

Proof 3.3. *See Appendix C.*

In Figure 4, we plot the minimum computation energy as a function of the input data size, and compare it with a scaling law of L^3. It shows that $\mathcal{E}_m^*$ scales at L^3.

4. Optimal Transmission Energy under Cloud Execution

In this section, we consider the problem of scheduling data transmission via rate adaptation to wireless (fading) channel variations, under a deadline

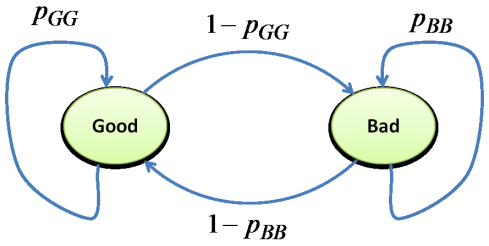


Figure 5: The Gilbert-Elliott (GE) channel model.

constraint. As such, we first briefly describe the channel model. Next, we derive the minimum expected energy expenditure for transmission under different channel models.

4.1. Wireless Channel Models

As shown in Fig. 2, we consider the scheduling of L bits of data with a deadline in T discrete time slots. The channel state at time slot t is denoted as g_t. We assume that only causal knowledge of the channel state is available. In this work, we consider two types of channel state models:

1) i.i.d. Channel Model: The channel states $\{g_t\}$ are independently and identically distributed (i.i.d). The i.i.d channel state model is self-explanatory. For instance, in [12, 13] channel states are modeled as truncated exponential random variables.

2) Gilbert-Elliott Channel Model: The channel states $\{g_t\}$ are determined by a discrete state space Markov model. For Markovian channel states, we consider the Gilbert-Elliott (GE) channel model [14, 15] in which there are two states: "good" and "bad" channel conditions, denoted as G and B, respectively. The two states correspond to a two-level quantization of the channel gain. If the measured channel gain is above some value, the channel is labeled as good. Otherwise, the channel is labeled as bad. Let the (average) channel gains of the good and bad states be g_G and g_B, respectively.

In this model, as illustrated in Figure 5, the state transition matrix is completely determined by the values p_{GG} (for the probability that the next state is the good state, given that the current state is also the good state) and p_{BB} (for the probability that the next state is the bad state, given that the current state is also the bad state). Accordingly, we have

$$p_{GB} = 1 - p_{GG}, \tag{20}$$

$$p_{BG} = 1 - p_{BB}, \tag{21}$$

where p_{GB} denotes the probability in which channel will transit from the good state to the bad state in the next time slot and p_{BG} denotes the probability in which channel will transit from the bad state to the good state in the next time slot. The state sojourn time is geometrically distributed. As such, the mean state sojourn time (duration of being in a state), measured in number of time slots in this state, is given by

$$T_G = \frac{1}{1-p_{GG}}, \tag{22}$$

$$T_B = \frac{1}{1-p_{BB}}. \tag{23}$$

4.2. Optimal Data Transmission Scheduling

We consider a discrete time model as in [12, 13]. We denote t as discrete time index in descending order (from $t = T$ to $t = 1$). In time slot t, if the number of bits transmitted is s_t, the transmission energy cost is $\mathcal{E}_t(s_t, g_t) = \lambda \frac{s_t^n}{g_t}$. Therefore, the optimization problem in Eq. (4) for the optimal data-transmission schedule can be rewritten as,

$$\begin{aligned} \min_{s_t}: \quad & \mathbb{E}\left[\sum_{t=1}^{T} \mathcal{E}_t(s_t, g_t)\right] \\ \text{s.t.:} \quad & \sum_{t=1}^{T} s_t = L, \\ & s_t \geq 0, \forall t. \end{aligned} \tag{24}$$

This optimization problem will be solved under the aforementioned channel models, including

1) i.i.d. Channel States: With an optimal scheduling, the minimum expected energy of the i.i.d. channel model is given by [12]:

$$\mathcal{E}_t(L) = \lambda L^n \zeta_t \tag{25}$$

where ζ_t can be solved recursively by:

$$\zeta_t = \begin{cases} \mathbb{E}[(\frac{1}{(g_t)^{\frac{1}{n-1}} + (\frac{1}{\zeta_{t-1}})^{\frac{1}{n-1}}})^{n-1}], & t \geq 2; \\ \mathbb{E}[\frac{1}{g_t}], & t = 1, \end{cases} \tag{26}$$

where the expectation is conduct over the distribution of the channel state g_t.

Proposition 4.1. *As stated in reference [12], when the application completion deadline of T increases, the minimum transmission energy decreases monotonically and scales with a factor of $T^{-(n-1)}$, where n is the monomial order in Eq.* (3)*:*

$$\mathcal{E}_t(L) \sim T^{-(n-1)} \tag{27}$$

2) GE Channel Model: The derivation for the optimal scheduler and the minimum transmission energy for the GE channel model is provided in the Appendix D. The minimum expected energy depends on the channel state at $t = T + 1$. If, at $t = T + 1$, the channel is in the good state, the optimal number of data bits transmitted in each time slot is given by

$$s_t^*(l_t, g_t) = \begin{cases} l_t \left(\frac{(g_t)^{\frac{1}{n-1}}}{(g_t)^{\frac{1}{n-1}} + (\frac{1}{\zeta_{t-1,G}})^{\frac{1}{n-1}}} \right), & t \geq 2; \\ l_1, & t = 1, \end{cases} \tag{28}$$

where l_t denotes the number of unfinished bits at time slot t, and

$$\zeta_{t;G} = \begin{cases} p_{GG} \left[\left(\frac{1}{(g_G)^{\frac{1}{n-1}} + (\frac{1}{\zeta_{t-1;G}})^{\frac{1}{n-1}}} \right)^{n-1} \right] \\ + p_{GB} \left[\left(\frac{1}{(g_B)^{\frac{1}{n-1}} + (\frac{1}{\zeta_{t-1;G}})^{\frac{1}{n-1}}} \right)^{n-1} \right], & t \geq 2; \\ p_{GG} \left[\frac{1}{g_G} \right] + p_{GB} \left[\frac{1}{g_B} \right], & t = 1. \end{cases} \tag{29}$$

With this optimal scheduling, the minimum expected energy is given by:

$$\mathcal{E}_t(L; G) = \lambda L^n \zeta_{t;G}. \tag{30}$$

If, at $t = T + 1$, the channel is in the bad state, the optimal number of data bits transmitted in each time slot is given by

$$s_t^*(l_t, g_t) = \begin{cases} l_t \left(\frac{(g_t)^{\frac{1}{n-1}}}{(g_t)^{\frac{1}{n-1}} + (\frac{1}{\zeta_{t-1,B}})^{\frac{1}{n-1}}} \right), & t \geq 2; \\ l_1, & t = 1, \end{cases} \tag{31}$$

where

$$\zeta_{t;B} = \begin{cases} p_{BB}\left[\left(\frac{1}{(g_B)^{\frac{1}{n-1}}+(\frac{1}{\zeta_{t-1;B}})^{\frac{1}{n-1}}}\right)^{n-1}\right] \\ +p_{BG}\left[\left(\frac{1}{(g_G)^{\frac{1}{n-1}}+(\frac{1}{\zeta_{t-1;B}})^{\frac{1}{n-1}}}\right)^{n-1}\right], & t \geq 2; \\ p_{BB}\left[\frac{1}{g_B}\right] + p_{BG}\left[\frac{1}{g_G}\right], & t = 1. \end{cases} \tag{32}$$

With this optimal scheduling, the minimum expected energy is given by:

$$\mathcal{E}_t(L;B) = \lambda L^n \zeta_{t;B}. \tag{33}$$

Given that, at steady state, the probability that a channel is in good or bad state is $\frac{T_G}{T_G+T_B}$ and $\frac{T_B}{T_G+T_B}$, respectively, the minimum expected transmission energy $\mathcal{E}_c^*$ is:

$$\begin{aligned} \mathcal{E}_c^*(L,T) &= \frac{T_G}{T_G+T_B}\mathcal{E}_t(L;G) \\ &+ \frac{T_B}{T_G+T_B}\mathcal{E}_t(L;B). \end{aligned} \tag{34}$$

Proposition 4.2. *As the data size of L increases, the minimum transmission energy increases monotonically and scales with a factor of L^n, where n is the monomial order in Eq.* (3)

$$\mathcal{E}_c^* \sim L^n \tag{35}$$

Proposition 4.3. *As the application completion deadline of T increases, the minimum transmission energy decreases monotonically and scales with a factor of $T^{-(n-1)}$, where n is the monomial order in Eq.* (3)

$$\mathcal{E}_c^* \sim T^{-(n-1)} \tag{36}$$

Proof 4.1. *See Appendix E.*

Proposition 4.4. *In Eq.* (29) *and* (32)*, neither $\zeta_{t;G}$ nor $\zeta_{t;B}$ scales to infinity.*

Proof 4.2. *See Eq.* (E.13) *and* (E.14) *in Appendix E.*

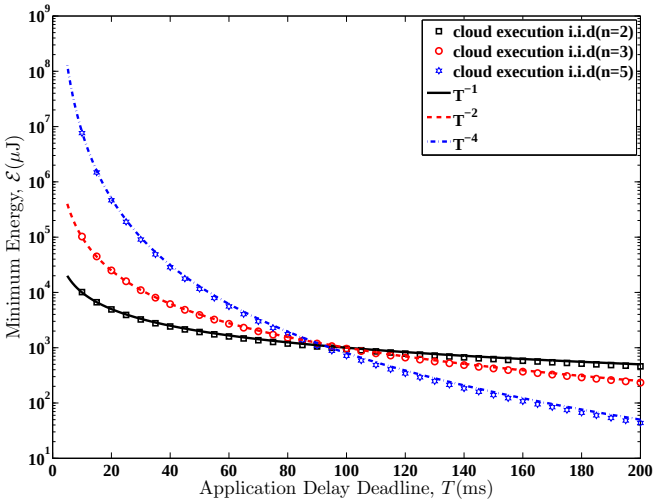


Figure 6: Expected transmission energy is plotted as a function of deadline T for the i.i.d channel model ($L = 800bits$).

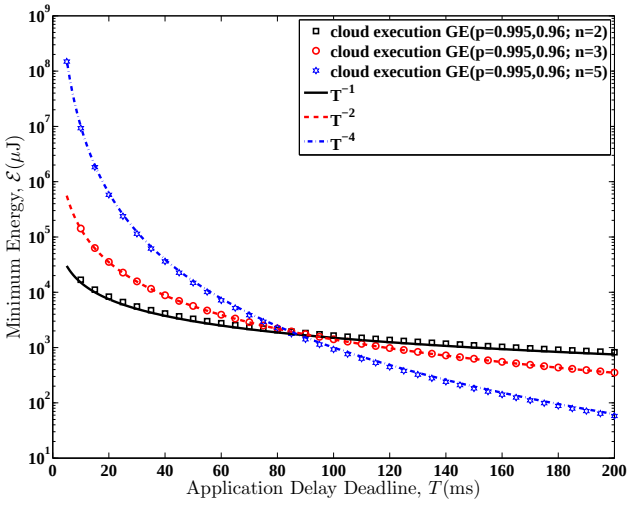


Figure 7: Expected transmission energy is plotted as a function of deadline T for the GE channel model. In this graph, $L = 800bits$, $p_{GG} = 0.995$, $p_{BB} = 0.96$, $g_G = 1$ and $g_B = 0.1$.

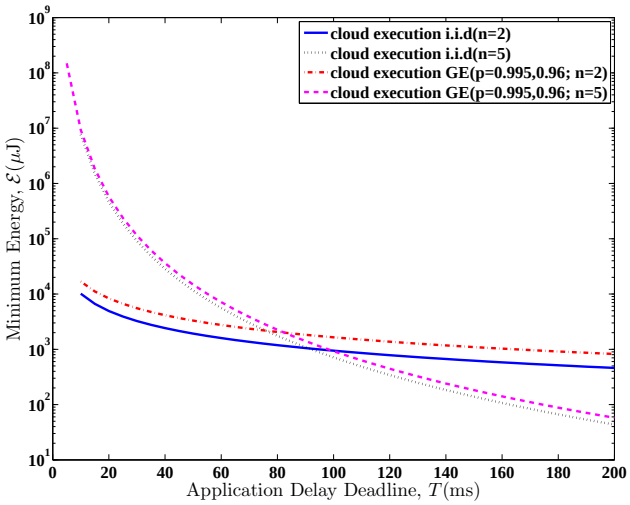


Figure 8: The expected transmission energy is plotted as a function of deadline T for the i.i.d channel model and the GE channel model. In this graph, $L = 800bits$.

In Figure 6 and 7, we plot the expected transmission energy under both i.i.d model and GE model as a function of the deadline T with different n, and compare them with a scaling factor of $T^{-(n-1)}$. For the i.i.d model, we use the truncated exponential random variable g with threshold 0.001. That is, $f(g) = e^{-(g-0.001)}$ for $g \geq 0.001$; $f(g) = 0$ otherwise. For the GE model, we set the parameters as $p_{GG} = 0.995$, $p_{BB} = 0.96$, $g_G = 1$ and $g_B = 0.1$. Note that, the scaling factor matches the numerical results well for both the cases of the i.i.d model and the GE model, which is consistent with Proposition 4.1 and Proposition 4.3. Moreover, as the application delay deadline T becomes smaller, the notation $T^{-(n-1)}$ will be larger, which results in more energy consumption. It also suggests that as it gets closer to the execution deadline, the chip clock frequency will be accelerated to meet the deadline.

In Figure 8, we plot the expected minimum transmission energy as a function of the deadline T. It can be observed that the energy consumed in the i.i.d. model is slightly smaller than that in the GE model, for the same monomial order n. In Section 5, the GE model is chosen to compare with the mobile execution so as to achieve the optimal consumed energy.

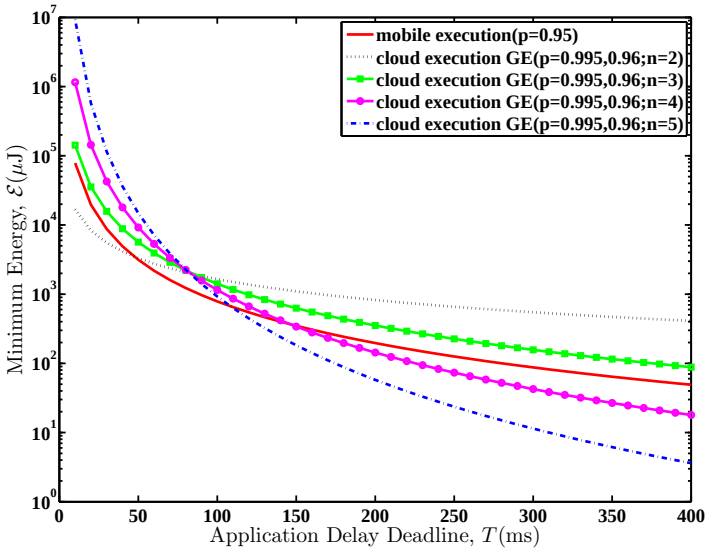


Figure 9: The minimum energy, $\mathcal{E}^*$, is plotted as a function of the application delay deadline T, for the mobile execution and the cloud execution. The task load is modeled as the Gamma distribution, with $\alpha = 4$, $\beta = 200$, and $L = 800bits$. The channel is a GE model with $p_{GG} = 0.995$, $p_{BB} = 0.96$, $g_G = 1$ and $g_B = 0.1$.

5. Optimal Application Execution Policy

In this section, we develop the optimal application execution policy, based on the analytical results obtained in Section 3 and Section 4. In particular, for a given application profile of $A(L, T)$, we compare the minimum computation energy for the mobile execution and the minimum transmission energy for the cloud execution. The optimal application execution policy is to choose whichever consumes less energy on the mobile device, in order to extend the battery life.

As proved previously in Section 3 and Section 4, there are some interesting relationships between the energy consumption and the application profile (i.e., data size and deadline delay): $\mathcal{E}_m^* \sim L^3$ and $\mathcal{E}_m^* \sim T^{-2}$ for the mobile execution, and $\mathcal{E}_c^* \sim L^n$ and $\mathcal{E}_c^* \sim T^{-(n-1)}$ for the cloud execution, respectively. Thus, when $n < 3$, the cloud execution consumes less energy for large data, while when $n > 3$, it is also encouraged to offload the application to the cloud for relatively long delay deadline.

As an example, we use the same application parameters of the mobile execution and the cloud execution as those described in Section 3 and 4, to

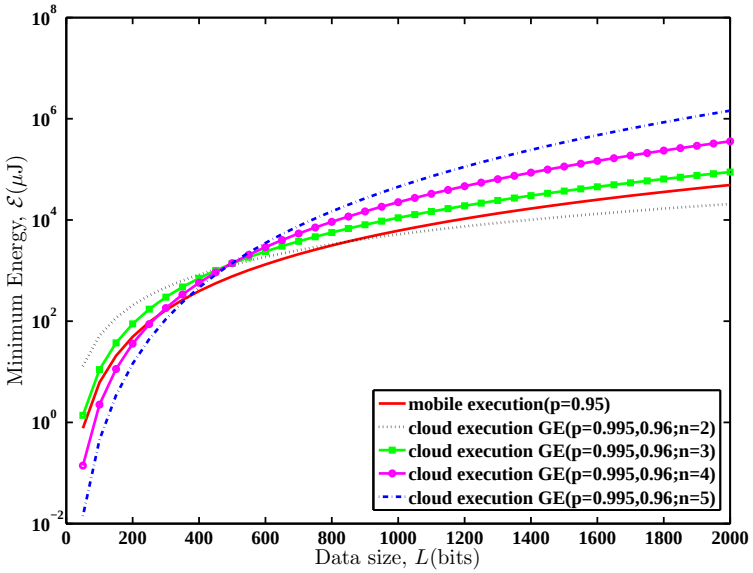
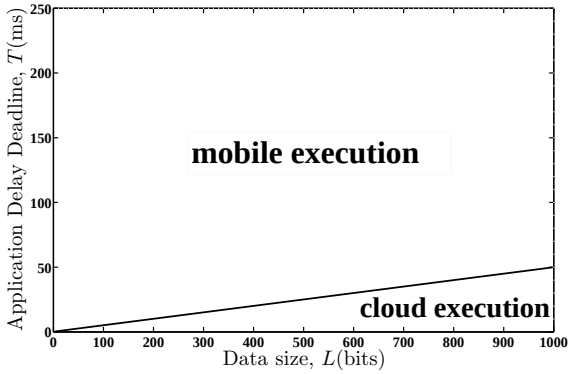


Figure 10: The minimum energy, $\mathcal{E}^*$, is plotted as a function of the data size L. The task load is modeled as the Gamma distribution, with $\alpha = 4$, $\beta = 200$, and $T = 50ms$. The channel is assumed as the GE model with $p_{GG} = 0.995$, $p_{BB} = 0.96$, $g_G = 1$ and $g_B = 0.1$.

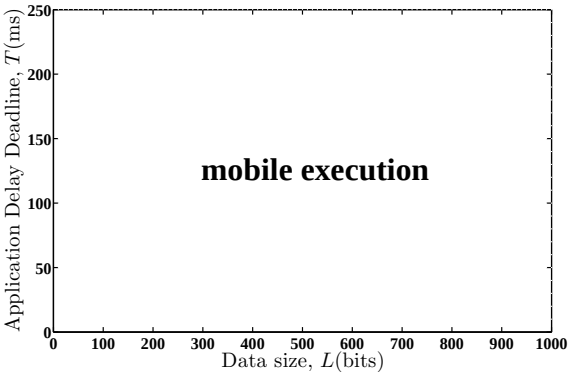
compare the energy consumptions. In Figure 9, we plot the minimum energy consumed by the mobile device for the mobile execution and the cloud execution, as a function of the application completion deadline of T. The optimal execution strategy depends on the monomial order of n. On one hand, when n is smaller than 3, the cloud execution is more energy-efficient when the delay deadline is below a threshold. This is because, when $n < 3$, the scaling factor for the cloud execution is slower than T^{-2}, the scaling factor for the mobile execution. On the other hand, when n is larger than 3, the cloud execution is more energy-efficient when the delay deadline is beyond a threshold. This is because, in this case, the scaling factor for the cloud execution is faster than T^{-2}, the scaling factor for the mobile execution. Moreover, by optimally deciding where to execute the application, a significant amount of energy can be saved on the mobile devices. For example, for an application profile of $A(800bits, 400ms)$, the mobile execution consumes 13 times energy more than the cloud execution for $n = 5$.

In Figure 10, we plot the minimum energy consumed by the mobile device for the mobile execution and the cloud execution, as a function of the input data size of L. The optimal execution strategy depends on the monomial order of n in Eq. (3). On one hand, when n is smaller than 3, the cloud execution is more energy-efficient when the data size is beyond a threshold. This is because, when $n < 3$, the scaling factor for the cloud execution is slower than L^3, the scaling factor for the mobile execution. On the other hand, when n is larger than 3, the cloud execution is more energy-efficient when the input data size is below a threshold. This is because, when $n > 3$, the scaling factor for the cloud execution is faster than L^3, the scaling factor for the mobile execution.

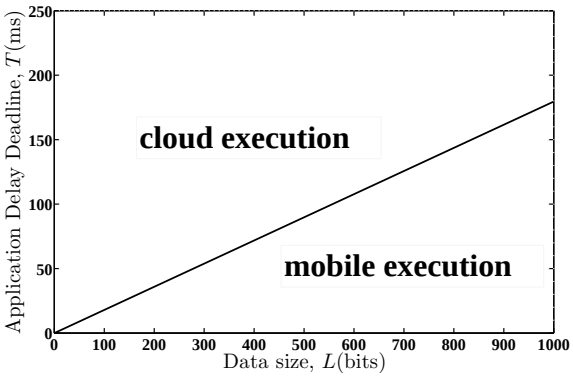
Moreover, for the specific application profile $A(L, T) = A(1000bits, 250ms)$ with different n, we plot in Figure 11 regions where the mobile execution or the cloud execution is more energy efficient. Specifically, for $n = 2$, the boundary between the two optimal operational regions is a line (i.e., $L/T = const$), where L/T can be considered as an effective data transmission rate. In this case, when the effective transmission rate is larger than a threshold, the cloud execution is optimal; otherwise, the mobile execution is optimal. All cases of $n = 3$ should be executed in the mobile device. This can be derived from Figures 9 and 10, in which for $n = 3$, the curve of the cloud execution is always above the curve of the mobile execution, indicating that energy consumption of mobile execution is smaller. In the case of $n = 4$, the boundary between the two optimal operating regions is a line (i.e., $L/T = \text{const}$). However, in this case, compared to the case of $n = 2$, when the effective transmission rate is larger than a threshold, the



(a) n=2



(b) n=3



(c) n=4

Figure 11: Optimal Energy Decision for a typical application profile $A(L,T) = A(1000bits, 250ms)$.

mobile execution is optimal; otherwise, the cloud execution is optimal..

The same approach can be adopt for different application parameters (i.e., the scale parameter and the shape parameter in the Gamma distribution of the number of CPU cycles, and monomial order of transmission model) to obtain the optimal policy for application execution. The thresholds can be identified similarly and the optimal operational regions for the mobile execution and the loud execution can be found respectively.

In summary, the optimal application execution policy depends on the application profile (i.e., the input data size of L and the application completion deadline of T), the wireless transmission model(i.e., the monomial order n in its energy consumption formula) and the ratio of energy coefficients (i.e., effective switched capacitance κ on the chip system of the device and energy coefficient λ in the wireless channel model). Moreover, energy consumed by the mobile device can be saved significantly by optimally deciding where to execute the application.

6. Related Work

Energy efficiency is a critical aspect for mobile platform, and it has been the topic of a number of studies. [18] presents the design and implementation of GRACE-OS for energy-efficient CPU scheduling on a stand-alone mobile device. Based on the probability distribution of the cycle demand, it finds out a schedule for each process and DVS algorithms are implemented in the CPU scheduler. [19] considers the energy-aware scheduling for embedded systems with multiprocessors as a probability-based load balancing problem. In this work, tasks are partitioned and assigned to processors based on their utilization in order for better energy reduction.

Computation offloading is a major concern for saving energy of the battery powered devices. Previous work in [22, 23, 24, 7, 20, 21] have investigated using remote process execution to extend battery life for mobile applications. Rudenko et al., in [22, 23] describe experiments using portable machines with WaveLAN radio devices, using experimental methods. They show that significant power can be saved through remote processing for several realistic tasks (up to 50% of battery life). Othman [24] uses simulation to show that battery life can be extended through process migration. The authors offer some decision-making algorithms such as History and Adaptive Load Sharing (ALS), which learns and adapts its decision based on previous CPU time measurements for a particular process. The paper does not discuss how the algorithm adapts to changes in noisy communication channels, which can have a significant impact on power consumption. Changjiu

Xian et al., in [20] find the optimal timeout for local execution and propose an adaptive approach for computation offloading to save energy on battery-powered systems. Computation instance is initially executed on the portable device with a timeout and it is off loaded to the server if the computation is not completed after the timeout. Peng Rong et al., in [21] adopt a two-state continuous-time Markov process to model the slow fading effect in the wireless channel, and provides off-line and on-line optimal policy to minimize the power consumption of the mobile device by remote processing under time constraints.

Moreover, some literatures have studied the energy issues for mobile applications in the cloud infrastructure. [3] presents a simple energy model to decide whether to offload applications to cloud. The trade-off is to determine the energy consumed by the computation in the mobile and communication for offloading, as well as the potential energy due to other additional operations, e.g., encryption for security. [2] demonstrates that workload, data communication patterns and technologies used (i.e., WLAN and 3G) are the main factors that highly affect the energy consumption of mobile applications in cloud computing. But its analysis is roughly based on statistical measurements and investigations. Also, [2, 3] mostly consider a fixed computation scheduling in the mobile device and a fixed bandwidth model for the wireless channel. Realistic models are needed to understand of the trade-off between computation and communication in the cloud-assisted mobile platform.

Compared to these previous efforts, this paper has several major contributions. First, the difference from the traditional computation offloading to a remote server or a fixed number of machines is that, in this architecture the cloud clone can offer services to the mobile users, e.g., the data back-up in the mobile devices. Second, the cloud clones can form a stable P2P network for more content sharing between machines in the cloud. Third, we consider a realistic wireless channel mode for the cloud execution, coupled with a realistic computing model in the mobile execution. In addition, we carefully examine the mobile execution and cloud execution in terms of energy conservation within an execution deadline, and finally we propose the energy-optimal execution policy. Our analysis of energy-optimal execution policy is based on the assumptions that the time of cloud execution is significantly short such that the execution time is negligible and the consumed energy of mobile device being idle before receiving the results is neglected. However, we have not considered any security issue on the cloud-assisted platform, thus the extra energy caused by additional operations concerning security, e.g., encryption and trust checking, is not taken into account.

7. Summary and Future Research

In this paper we investigated the problem of how to conserve energy for the resource-constrained mobile device, by optimally executing mobile applications in either the mobile device or the cloud clone. We proposed an optimization framework for energy-optimal application execution in the cloud-assisted mobile application platform. For the mobile execution, we minimize the computation energy by dynamically configuring the clock frequency of the chip, according to the workload distribution. For the cloud execution, we minimize the transmission energy by optimally scheduling data transmission across a stochastic wireless channel (i.e., the i.i.d model and the Gilbert-Elliott model). Closed-form solutions were obtained for both scheduling problems and are applied to decide the optimal application-execution condition under which either the mobile execution or the cloud execution is more energy-efficient for the mobile device. We also figure out the relationships between optimal energy and data size as well as delay deadline mathematically. Numerical results indicate that the optimal execution policy depends on the application profile, the wireless transmission model and the ratio of energy coefficients.

This paper only pays attention to energy issue related to mobile devices. For future work, the energy consumption in the cloud side will be taken into consideration, and we plan to minimize the energy consumption on both sides. Also, security mechanisms will be established on the mobile platform.

Acknowledgements

Part of this research with some preliminary results has been submitted to Infocom 2012. The authors would like to thank Dr. Xinwen Zhang, Dr. Xiaoqing Zhu, Dr. Shivkumar Kalyanaraman at IBM Research - India, and Prof. Changwen Chen at the University at Buffalo, State University of New York for their insightful discussions. Moreover, Yonggang Wen would like to thank Singapore Nanyang Technological University for the start-up grant support of this research.

Appendix A. Proof of Theorem 3.1

In this section, we will use the Lagrangian multiplier method to solve the optimization problem in Eq. (13).

$$\begin{aligned} L(f(w),\lambda) &= \sum_{w=1}^{W_\rho} F_W^c(w)[f(w)]^2 + \lambda(\sum_{w=1}^{W_\rho} \frac{1}{f(w)} - T) \\ &= \sum_{w=1}^{W_\rho} \{F_W^c(w)[f(w)]^2 + \frac{\lambda}{f(w)}\} - \lambda T \end{aligned}$$

The optimal clock schedule policy must satisfy the following conditions,

$$\frac{\partial L(f(w),\lambda)}{\partial f(w)} = 2F_W^c(w)f(w) - \frac{\lambda}{[f(w)]^2} = 0 \quad \text{(A.1)}$$

$$\frac{\partial L(f(w),\lambda)}{\partial \lambda} = \sum_{w=1}^{W_\rho} \frac{1}{f(w)} - T = 0. \quad \text{(A.2)}$$

Solving Eq. (A.1), we obtain that, for $1 \leq w \leq W_\rho$,

$$f^*(w) = \{\frac{\lambda}{2F_W^c(w)}\}^{1/3}. \quad \text{(A.3)}$$

Plugging Eq. (A.3) into Eq. (A.2), we obtain

$$\sum_{w=1}^{W_\rho} [F_W^c(w)]^{1/3} = T(\frac{\lambda}{2})^{1/3}. \quad \text{(A.4)}$$

Therefore, the optimal clock schedule policy is given by

$$f^*(w) = \frac{\theta}{T[F_W^c(w)]^{1/3}}, \quad \text{(A.5)}$$

where $\theta = \sum_{i=1}^{W_\rho} [F^c(i)]^{1/3}$. Substituting Eq. (A.5) into Eq. (12), we obtain the optimal computation energy as

$$\mathcal{E}_m^* = \frac{\kappa}{T^2}\{\sum_{w=1}^{W_\rho} [F_W^c(w)]^{1/3}\}^3. \quad \text{(A.6)}$$

Appendix B. Proof of Proposition 3.1

The minimum computation energy is given by

$$\mathcal{E}_m^* = \frac{\kappa}{T^2}\{\sum_{w=1}^{W_\rho}[F_W^c(w)]^{1/3}\}^3. \tag{B.1}$$

It is equivalent to show that, as $W_\rho \to \infty$,

$$\theta = \sum_{w=1}^{W_\rho}[F_W^c(w)]^{1/3} \tag{B.2}$$

converges to a fixed value.

For exponentially-tailed distribution, we have $F_W^c(w) \sim \mu e^{-\nu w}$ as $w \to \infty$ for some constant $\mu > 0$ and $\nu > 0$. Formally, for $\forall \epsilon > 0$, there exits a W, such that $|F_W^c(w) - \mu e^{-\nu w}| < \epsilon$. Using this fact, we can rewrite the energy factor θ as

$$\theta = \sum_{w=1}^{W}[F_W^c(w)]^{1/3} + \sum_{w=W+1}^{W_\rho}\mu^{\frac{1}{3}}e^{-\frac{\nu}{3}w}. \tag{B.3}$$

The first term is a constant, and the second term converges to a constant as W_ρ increases to ∞.

Appendix C. Proof of Proposition 3.3

In this section, we show the relationship of optimal energy and data size.

$$\mathcal{E}_m^* = \frac{\kappa}{T^2}\{\sum_{w=1}^{W_\rho}[F_W^c(w)]^{1/3}\}^3. \tag{C.1}$$

We just need to compute

$$\sum_{w=1}^{W_\rho}[F_W^c(w)]^{1/3}\}, \tag{C.2}$$

where

$$F_W^c(w) = F_X^c(\frac{w}{L}). \tag{C.3}$$

Assume $W_\rho = LT_\rho$, then

$$\begin{aligned}\sum_{w=1}^{W_\rho}[F_W^c(w)]^{1/3} &= \sum_{t=0}^{T_\rho-1}(\sum_{i=1}^{L}[F_W^c(Lt+i)]^{1/3}) \\ &= \sum_{t=0}^{T_\rho-1}(\sum_{i=1}^{L}[F_X^c(t+\frac{i}{L})]^{1/3}),\end{aligned}$$

According to the mean value theorem, there exists η ($\frac{1}{L} < \eta < 1$), such that

$$\begin{aligned}\sum_{t=0}^{T_\rho-1}(\sum_{i=1}^{L}[F_X^c(t+\frac{i}{L})]^{1/3}) &= \sum_{t=0}^{T_\rho-1}(L[F_X^c(t+\eta)]^{1/3}) \\ &= L\sum_{t=0}^{T_\rho-1}([F_X^c(t+\eta)]^{1/3}).\end{aligned}$$

Hence,

$$\sum_{w=1}^{W_\rho}[F_W^c(w)]^{1/3} = L\sum_{t=0}^{T_\rho-1}([F_X^c(t+\eta)]^{1/3}).$$

For the Gamma distribution, the complementary cumulative distribution function(CCDF) is

$$\sum_{i=0}^{\alpha-1}\frac{(\beta x)^i}{i!}e^{-\beta x}. \tag{C.4}$$

For this exponentially-tailed distribution, we have $F_X^c(t+\eta) \sim \mu e^{-\nu(t+\eta)}$ as $t \to \infty$ for some constant $\mu > 0$ and $\nu > 0$. Formally, for $\forall \epsilon > 0$, there exits a T_N, such that for $t > T_N$, we have $|F_X^c(t+\eta) - \mu e^{-\nu(t+\eta)}| < \epsilon$. Using this fact, we get

$$\begin{aligned}\sum_{t=0}^{T_\rho-1}([F_X^c(t+\eta)]^{1/3}) &= \sum_{t=0}^{T_N}([F_X^c(t+\eta)]^{1/3}) \\ &+ \sum_{t=T_{N+1}}^{T_\rho-1}\mu^{\frac{1}{3}}e^{-\frac{\nu}{3}(t+\eta)}.\end{aligned} \tag{C.5}$$

The first term is a constant, and in the following we examine the second term

$$\sum_{t=T_{N+1}}^{T_\rho-1} \mu^{\frac{1}{3}} e^{-\frac{\nu}{3}(t+\eta)} = \frac{\mu^{\frac{1}{3}} e^{-\frac{\nu}{3}[T_N+1+\eta]}[1-(e^{-\frac{\nu}{3}})^{T_\rho-T_N-1}]}{1-e^{-\frac{\nu}{3}}} \tag{C.6}$$

As $T_\rho \to \infty$, the second term converges to a constant. Thus, $\sum_{t=0}^{T_\rho-1}([F_X^c(t+\eta)]^{1/3})$ converges to a constant and would not scale to ∞.

Hence,

$$\sum_{w=1}^{W_\rho} [F_W^c(w)]^{1/3} \sim L \tag{C.7}$$

Combining the Eq.(C.1) and Eq.(C.7), we have

$$\mathcal{E}_m^* \sim L^3. \tag{C.8}$$

Appendix D. Optimal Transmission Energy for the GE model

In this section, we provide the proof for the results of optimal scheduling under the GE channel mode.

Using the dynamic programming (DP) approach, the optimization problem in Eq. (24) can be rewritten as

$$J_t(l_t, g_t) = \begin{cases} \min\limits_{0\le s_t\le l_t} \left(\frac{s_t^n}{g_t} + \mathbb{E}(J(l_t - s_t, g))\right), & t \ge 2 \\ \frac{l_1^n}{g_1}, & t = 1. \end{cases} \tag{D.1}$$

Here, l_t denotes the number of remaining (un-transmitted) bits at t, with $l_{t-1} = l_t - s_t$.

We use the induction approach. We first consider the case that at $t = T+1$, the channel is in the good state. At $t = 1$, all the remaining l_1 bits have to be transmitted to meet the deadline constraint. Given the channel state at $t = 2$ is in the good state, the expected minimum energy is given by

$$\begin{aligned} \bar{J}_1(l_1) &= \mathbb{E}\left[\frac{l_1^n}{g_1}\right] \\ &= l_1^n \left(p_{GG}\left[\frac{1}{g_G}\right] + p_{GB}\left[\frac{1}{g_B}\right]\right). \end{aligned} \tag{D.2}$$

Now suppose Eq. (29) is true for $t-1$, the DP problem stated in Eq. (D.1) becomes

$$J_t(l_t, g_t) = \min_{0 \le s_t \le l_t} \left(\frac{s_t^n}{g_t} + (l_t - s_t)^n \zeta_{t-1;G} \right). \tag{D.3}$$

The optimal s_t, denoted as s_t^*, can be solved as:

$$s_t^* = \frac{l_t g_t^{\frac{1}{n-1}}}{g_t^{\frac{1}{n-1}} + \left(\frac{1}{\zeta_{t-1;G}}\right)^{\frac{1}{n-1}}}. \tag{D.4}$$

Substituting (D.4) to (D.3), we have:

$$J_t(l_t, g_t) = l_t^n \left[\left(\frac{1}{(g_t)^{\frac{1}{n-1}} + (\frac{1}{\zeta_{t-1,G}})^{\frac{1}{n-1}}} \right)^{n-1} \right] \tag{D.5}$$

By taking expectation of $J_t(l_t, g_t)$ with respect to g_t, we have:

$$\begin{aligned} \zeta_{t,G} &= \mathbb{E}\left[\left(\frac{1}{(g_t)^{\frac{1}{n-1}} + (\frac{1}{\zeta_{t-1;G}})^{\frac{1}{n-1}}} \right)^{n-1} \right] \\ &= p_{GG} \left[\left(\frac{1}{(g_G)^{\frac{1}{n-1}} + (\frac{1}{\zeta_{t-1;G}})^{\frac{1}{n-1}}} \right)^{n-1} \right] \\ &+ p_{GB} \left[\left(\frac{1}{(g_B)^{\frac{1}{n-1}} + (\frac{1}{\zeta_{t-1;G}})^{\frac{1}{n-1}}} \right)^{n-1} \right]. \end{aligned} \tag{D.6}$$

Therefore, the result in Eq. (29) follows by induction. The proof for the results in Eq. (32) follows the exact same rationale, thus is omitted here for brevity.

Appendix E. Proof of Proposition 4.3

In this section, we present the relationship of optimal transmission energy and delay deadline. Since,

$$\zeta_{t;G} = \begin{cases} p_{GG}\left[\left(\frac{1}{(g_G)^{\frac{1}{n-1}}+(\frac{1}{\zeta_{t-1;G}})^{\frac{1}{n-1}}}\right)^{n-1}\right] \\ +p_{GB}\left[\left(\frac{1}{(g_B)^{\frac{1}{n-1}}+(\frac{1}{\zeta_{t-1;G}})^{\frac{1}{n-1}}}\right)^{n-1}\right], & t \geq 2; \\ p_{GG}\left[\frac{1}{g_G}\right] + p_{GB}\left[\frac{1}{g_B}\right], & t = 1. \end{cases} \tag{E.1}$$

and

$$g_G > g_B, \tag{E.2}$$

we have

$$\zeta_{t;G} < \zeta_{t;g_B} \tag{E.3}$$

and

$$\zeta_{t;G} > \zeta_{t;g_G}, \tag{E.4}$$

in which

$$\zeta_{t;g_B} = \begin{cases} p_{GG}\left[\left(\frac{1}{(g_B)^{\frac{1}{n-1}}+(\frac{1}{\zeta_{t-1;g_B}})^{\frac{1}{n-1}}}\right)^{n-1}\right] \\ +p_{GB}\left[\left(\frac{1}{(g_B)^{\frac{1}{n-1}}+(\frac{1}{\zeta_{t-1;g_B}})^{\frac{1}{n-1}}}\right)^{n-1}\right], & t \geq 2; \\ p_{GG}\left[\frac{1}{g_B}\right] + p_{GB}\left[\frac{1}{g_B}\right], & t = 1. \end{cases} \tag{E.5}$$

and

$$\zeta_{t;g_G} = \begin{cases} p_{GG}\left[\left(\frac{1}{(g_G)^{\frac{1}{n-1}}+(\frac{1}{\zeta_{t-1;g_G}})^{\frac{1}{n-1}}}\right)^{n-1}\right] \\ +p_{GB}\left[\left(\frac{1}{(g_G)^{\frac{1}{n-1}}+(\frac{1}{\zeta_{t-1;g_G}})^{\frac{1}{n-1}}}\right)^{n-1}\right], & t \geq 2; \\ p_{GG}\left[\frac{1}{g_G}\right] + p_{GB}\left[\frac{1}{g_G}\right], & t = 1. \end{cases} \tag{E.6}$$

Also,

$$P_{GB} + P_{GG} = 1 \tag{E.7}$$

such that

$$\zeta_{t;g_B} = \begin{cases} \left[\left(\frac{1}{(g_B)^{\frac{1}{n-1}}+(\frac{1}{\zeta_{t-1;g_B}})^{\frac{1}{n-1}}}\right)^{n-1}\right], & t \geq 2; \\ \frac{1}{g_B}, & t = 1. \end{cases} \tag{E.8}$$

When t is equal to or greater than 2,

$$\left[\frac{1}{\zeta_{t;g_B}}\right]^{\frac{1}{n-1}} = \left[\frac{1}{\zeta_{t-1;g_B}}\right]^{\frac{1}{n-1}} + (g_B)^{\frac{1}{n-1}} \tag{E.9}$$

Hence,

$$\begin{aligned} \left[\frac{1}{\zeta_{t;g_B}}\right]^{\frac{1}{n-1}} &= \left[\frac{1}{\zeta_{1;g_B}}\right]^{\frac{1}{n-1}} + (t-1)(g_B)^{\frac{1}{n-1}} \\ &= (g_B)^{-(n-1)}t, \end{aligned} \tag{E.10}$$

$$\zeta_{t;g_B} = \left[\frac{1}{g_B}\right] t^{-(n-1)}. \tag{E.11}$$

Similarly,

$$\zeta_{t;g_G} = \left[\frac{1}{g_G}\right] t^{-(n-1)}. \tag{E.12}$$

In that case,

$$\zeta_{t;G} \sim t^{-(n-1)} \tag{E.13}$$

Similarly,

$$\zeta_{t;B} \sim t^{-(n-1)} \tag{E.14}$$

Since,

$$\mathcal{E}_t(L;G) = L^n \zeta_{t;G} \tag{E.15}$$

and

$$\mathcal{E}_t(L;B) = L^n \zeta_{t;B} \tag{E.16}$$

$$\begin{aligned} \mathcal{E}_c^*(L,T) &= \frac{T_G}{T_G+T_B}\mathcal{E}_t(L;G) \\ &+ \frac{T_B}{T_G+T_B}\mathcal{E}_t(L;B). \end{aligned} \tag{E.17}$$

Therefore,

$$\mathcal{E}_c^*(L,T) \sim T^{-(n-1)} \tag{E.18}$$

References

[1] M. Satyanarayanan, Fundamental Challenges in Mobile Computing, In Proceedings of ACM Symposium on Principles of Distributed Computing, ACM Press, 1996, pp. 1-7.

[2] A. P. Miettinen and J. K. Nurminen, Energy Efficiency of Mobile Clients in Cloud Computing, In Proceedings of the 2nd USENIX conference on hot topics in cloud computing, Berkeley, CA, USA, 2010.

[3] K. Kumar and Y. H. Lu, Cloud Computing for Mobile Users: Can Offloading Computation Save Energy?, IEEE Computer, Vol. 43, No.4, April 2010, pp. 51-56.

[4] Armbrust, M, Fox, A., Griffith, R., Joseph, A., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Zaharia, A view of cloud computing, Communication of the ACM, 53 (4), 2010, pp. 5058.

[5] M. Satyanarayanan, P. Bahl, R. Caceres and N. Davies, The Case for VM-based Cloudlets in Mobile Computing, IEEE Pervasive Computing, Oct-Dec 2009, Vol. 8, No. 4, pp. 14-23.

[6] B. G. Chun and P. Maniatis, Augmented Smartphone Applications Through Clone Cloud Execution, In Proceedings of the 12th conference on Hot topics in operating systems, Berkeley, CA, USA, 2009.

[7] R. K. Balan, et al, The Case for Cyber Foraging, In Proceedings of 10th ACM Special Interest Group on Operating Systems European Workshop(SIGOPS), ACM Process, 2002, pp. 87-92.

[8] B.G. Chun, S.h. Ihm, P. Maniatis, M. Naik and A. Patti, CloneCloud: Elastic Execution between Mobile Device and Cloud, In Proceedings of the 6th European Conference on Computer Systems (EuroSys 2011), Apr 2011.

[9] X. W. Zhang, A. Kunjithapatham, S. Jeong and S. Gibbs, Towards an Elastic Application Model for Augmenting the Computing Capabilities of Mobile Devices with Cloud Computing, Mobile Network Applications, 2011(16), pp. 270-284.

[10] J. M. Rabaey, Digital Integrated Circuits, Prentice Hall, 1996.

[11] T. Burd and R. Broderson, Processor Design for Portable Systems, Journal of VLSI Singapore Process, Aug 1996, Vol. 13, No. 2, pp. 203-222.

[12] J. Lee and N. Jindal, Delay Constrained Scheduling over Fading Channels: Optimal Policies for Monomial Energy-Cost Functions, IEEE International Conference on Communications (ICC), Dresden, Germany, June 2009.

[13] J. Lee, and N. Jindal, Energy-efficient Scheduling of Delay Constrained Traffic over Fading Channels, IEEE Trans. Wireless Communications, Apr 2009, vol. 8, no. 4, pp. 1866-1875.

[14] M. Zafer and E. Modiano, Delay Constrained Energy Efficient Data Transmission over a Wireless Fading Channel, Workshop on Information Theory and Application, University of California, San Diego, Feb 2007.

[15] M. Zafer and E. Modiano, Minimum Energy Transmission over a Wireless Fading Channel with Packet Deadlines, Proceedings of IEEE Conference on Decision and Control (CDC), New Orleans, LA, Dec 2007.

[16] J. R. Lorch and A. J. Smith, Improving Dynamic Voltage Scaling Algorithms with PACE, Proceedings of ACM SIGMETRICS 2001, Cambridge, MA, USA, June 2001.

[17] W. H. Yuan and K. Nahrstedt, Energy-Efficient Soft Real-Time CPU Scheduling for Mobile Multimedia Systems, Proceedings of ACM SOSP'03, New York, USA, October 19-22, 2003.

[18] W. H. Yuan and K. Nahrstedt, Energy-Efficient CPU Scheduling for Multimedia Applications, ACM Transactions on Computer Systems, June 2005, Vol. V, NO. N, pp. 1-36.

[19] C. Xian, Y. Lu, and Z. Li, Energy-Aware Scheduling for Real-Time Multiprocessor Systems with Uncertain Task Execution Time, Design Automation Conference, 44th ACM/IEEE, 2007, pp. 664-669.

[20] C. X, Y. H. Lu and Z. Y Li, Adaptive Computation Offloading for Energy Conservation on Battery-Powered Systems, In Proceedings of 2007 International Conference on Parallel and Distributed Systems, Dec 2007, Vol. 2, pp. 1-8.

[21] P. Rong and M. Pedram, Extending the Lifetime of a Network of Battery-Powered Mobile Devices by Remote Processing: a Markovian Decision-based Approach, In Proceedings of the 40th Annual Design Automation Conference, New York, USA, 2003, pp. 906-911.

[22] A. Rudenko, P. Reiher, G. Popek and G. Kuenning, Saving Portable Computer Battery Power through Remote Process Execution, Mobile Computing and Communication Reviews, 1998, Vol. 2, No.1, pp. 19-26.

[23] A. Rudenko, P. Reiher, G. Popek and G. Kuenning, The Remote Processing Framework for Portable Computer Power Saving, in Proceedings of the ACM Symposium on Applied Computing, San Antonio, TX, 1999.

[24] M. Othman and S. Hailes, Power Conservation Strategy for Mobile Computers Using Load Sharing, Mobile Computing and Communication Review, 1998, Vol. 2, No.1, pp.44-51.

[25] E. Uysal-Biyikoglu, B. Prabhakar, and A. El Gamal, Energy Efficient Packet Transmission over a Wireless Link, IEEE/ACM Transactions on Networking, Aug 2002, Vol. 10, No. 4, pp.487-499.