Energy-Optimal Resource Scheduling and Computation Offloading in Small Cell Networks

Wael Labidi, Mireille Sarkiss and Mohamed Kamoun
CEA, LIST, Communicating Systems Laboratory
BC 173, 91191 Gif-sur-Yvette, France
{wael.labidi, mireille.sarkiss, mohamed.kamoun}@cea.fr

***Abstract*—This paper provides a joint optimization framework of radio resource scheduling and computation offloading in small cell LTE based networks. We consider that mobile users are served by nearby small cell base stations which can be endowed with some computational capabilities. The objective is to minimize the average energy consumption at the user terminal to run its mobile applications, either locally or remotely, while satisfying average delay constraints tolerated by these applications. For this problem, we investigate offline dynamic programming approaches and we devise two solutions: deterministic and randomized, to find the optimal radio scheduling-offloading policy. We show that the dynamic offline strategies are able of achieving optimal energy efficiency at the mobile terminals. Indeed, they can adapt the processing decisions between: local processing, offloading, and staying idle, by exploiting their knowledge on the channel conditions and the application properties.**

I. Introduction

Small Cells technology has gained recently tremendous attention in wireless communities and has become an essential component to improve cell coverage and boost network capacity for future Long Term Evolution (LTE) standards. Deployed closer to mobile users, small cell base stations, known as Small Cell enhanced Node B (SCeNBs), are expected to provide new services at high data rates, reduced energy and low cost, thus improving cellular applications for homes, enterprises, as well as for metropolitan and rural areas [1].

Parallel to the fast small cell deployments, resource-hungry applications have been continuously gaining popularity, driven by the development of smart devices, such as smartphones and tablets. Besides increasing users' demands for high-volume multimedia data, such mobile applications are typically demanding intensive computation and high energy consumption. Mobile devices, however, are still facing resource-constraints in terms of computation, storage and, in particular, energy due to their limited battery capacity. Recently, mobile cloud computing has emerged as a promising solution to extend these mobile terminals capabilities [2], [3]. By providing users on-demand wireless access to cloud infrastructures and exploiting cloud virtualization, computation offloading is enabled from mobile devices to resourcefull remote servers. The mobile users can execute remotely parts of their sophisticated applications and hence, prolong their battery lifetime by reducing locally the energy consumption.

In order to take advantage of improvements of both small cells and cloud computing technologies, the European research project TROPIC has proposed lately the new concept of femto-cloud computing [4]. In this framework, the online cloud services are available at proximity to the mobile users by equipping the SCeNBs in LTE based networks with some computational and storage resources. Therefore, the users can benefit from computation offloading and distributed storage services to nearby SCeNBs in the small cell cloud with reduced energy and latency. At the same time, they can experience better quality of service on the LTE radio links and higher capacity offered by the cooperating small cells.

Computation offloading has been significantly studied during the past decade to improve performance or save energy of mobile devices by analyzing radio and computation resources including bandwidths, available memory, server speeds and loads, and the amount of data exchanged between the mobile users and the cloud, *c.f.* [3], [5] and references therein. In [5], an efficient offloading policy is proposed by jointly configuring the clock frequency in the mobile device and scheduling the data transmission across stochastic wireless channel to minimize the energy consumption. In [6], mobile computation offloading and mobile data backup communication costs are evaluated in terms of bandwidth and energy consumption by realistic measurements on real devices. Considering a small cell cloud context, [7] addressed the tradeoff between latency and energy consumption in application offloading and derived an optimal communication strategy assuming multiple antennas and the optimal distribution of computational load between a terminal and the serving cells. Moreover, [8] proposed a centralized dynamic application partitioning for computation offloading to jointly optimize communication and computation resource allocations under latency constraints.

Similarly to the previous works, we consider in this paper, a small cell dynamic environment experiencing time-varying fading channels and dynamic traffic loads at the mobile terminals through variable rate applications. In this scenario, we investigate joint dynamic optimization of radio resource scheduling and computation offloading to minimize the energy consumed at the user to process its applications while satisfying the application delay requirements. Unlike the approach of [8] which resolves the optimization problem for instantaneous channel realization and under instantaneous delay constraint, we consider here a time-averaged optimization. Therefore, the scheduling and processing decisions can adapt to the time-varying channel states and user buffer states to reduce the average energy consumption under average delay constraints. We formulate this problem as a Constrained Markov Decision Problem (CMDP). Then, using dynamic programming tools, we investigate offline solutions to find the optimal scheduling-offloading strategies based on prior knowledge of the channel distribution and the application arrival rates.

978-1-4799-8078-9/15/$31.00 ©2015 IEEE

The remainder of the paper is organized as follows. In section II, we describe the system model from communication and computation perspectives and we derive the energy consumed to process the application data according to different possible decisions. In section III, we formulate the optimization problem and in section IV, we present the proposed offline dynamic programming strategies. Numerical results are analyzed in section V and finally conclusions are given in section VI.

II. System Model

A. Communication Model

We consider a small-cell cloud scenario where mobile users can be served by one or multiple nearby SCeNBs that are endowed with some cloud resources. We focus in the sequel on a point-to-point communication between a given User Equipement (UE) and its serving SCeNB cloud enabled (SCeNBce). Whenever the UE has intensive and delay sensitive applications to run, the main concern is to minimize the energy consumed by the execution of such applications in order to save UE battery. Therefore, it can be envisioned that the UE sends an offloading request to an entity playing the role of Small Cell Cloud Manager (SCM), aware of all radio and computation resources at UEs and SCeNBsce. Then, this SCM makes a decision and sends it back to the UE whether to process the application locally at the mobile terminal or to exploit the available SCeNBce resources by offloading totally or partially the computation. These processing decisions depend on the transmission channel states, the computation resources at both sides and the latency imposed by the applications.

We assume that the processing decisions are performed at successive time slots of duration T. These periods can be aligned with the frame duration in LTE networks and with the instants where radio scheduling messages are executed. The communication channel between the UE and the SCeNBce is considered as Rayleigh fading channel with complex fading gain h. The channel remains constant during T and changes in an independent and identically distributed (i.i.d.) manner across time slots. Let $z = |h|^2$ denote the channel state, it is a continuous exponentially distributed random variable with probability density function $p(z) = \frac{1}{\tau} e^{-\frac{z}{\tau}}$ with mean τ. For practical considerations, only quantized channel state x is considered and thus x takes discrete values from a finite channel state space $\mathcal{X}$, corresponding to a sequence of fading power quantization thresholds [9]. Without loss of generality, we assume in our model the same communication channel for uplink (UL) and downlink (DL) transmissions between the UE and the SCeNB, and we consider that both transmissions undergo the same white Gaussian noise with zero mean and variance N_0.

B. Computation model

At the user side, the mobile application profile can be defined with two parameters: the input data size, *i.e.,* the number of data bits to be processed in the application and the completion deadline *i.e.,* the delay before which the application should be completed [5]. In our model, the input data is defined in terms of data packets of equal size, l bits. The packets' arrival follows a Poisson distribution with an average arrival rate of γ packets and these arrivals are i.i.d. The data packets are queued in the UE buffer until they are scheduled, either for local execution or for remote processing on the SCeNBce. The user buffer has a finite size of B packets. Let q denote the queue length in the buffer at a given time slot. Then, the buffer state q can take values from a set $\mathcal{Q} = [0, 1, \ldots, B]$ with cardinality $|\mathcal{Q}| = (1 + B)$. In addition, we consider that buffer overflow can occur when the queued packets exceed the buffer size, thus leading to packets loss. This event is defined with a probability of buffer overflow that should be kept under a predefined threshold δ_O.

The described Buffer State Information (QSI) and Channel State Information (CSI) define our system state as $s = (q, x)$. Then, the discrete finite space $\mathcal{S}$ of states s has cardinality $|\mathcal{S}| = |\mathcal{Q}| \times |\mathcal{X}|$. We assume that the state information is sent by the UE to the SCeNBce through channel quality indicators and buffer status reports as in LTE standards. Based on these status reports and on the available computation resources, the small cell manager SCM performs decisions upon user offloading requests to minimize the energy consumption at the UE terminal. Thus, at each time slot duration T, three scheduling decisions are possible: Processing of u queued packets ($u \leq q$) either by *(1)* local processing at the UE or by *(2)* computation offloading at the SCeNBce, and *(3)* idle mode *i.e.,* waiting until the next time slot to process. These decisions can generate different costs in terms of energy consumption and latency. They are discussed in the sequel.

1- *Local processing*: The mobile user executes u packets of its application data locally on its terminal. Let M_l denote the maximal number of packets that can be processed locally within a time period T. Then, $u \in [1, \ldots, M_l]$. Assuming that the UE terminal consumes a power P_u per packet processed locally, the energy consumption to process u packets during T is given by

$$E = u \times P_u \times T. \tag{1}$$

2- *Offloading*: For computation offloading, the mobile user transmits u packets for execution at the SCeNbce within a time period T. Let M_o denote the maximal number of offloaded packets that can be processed remotely within T. Then, $u \in [1, \ldots, M_o]$. In this case, the UE would incur an extra overhead in terms of energy for transmitting the data packets to be processed at the SCeNBce. Once the remote processing is completed, the SCeNBce will send back the resulting data to the UE. Accordingly, the energy consumed at the mobile terminal will include the energy spent for: data transmission, data reception and waiting the remote processing completion. It is derived as

$$\begin{aligned} E &= \frac{uL_{\mathrm{UL}}P_t}{W_{\mathrm{UL}}\log_2\left(1+\frac{P_t x}{W_{\mathrm{UL}}N_0}\right)} + \frac{uL_{\mathrm{DL}}P_r}{W_{\mathrm{DL}}\log_2\left(1+\frac{P_{t'} x}{W_{\mathrm{DL}}N_0}\right)} \\ &+ uT_wP_w, \end{aligned} \tag{2}$$

where P_t is the power used at the UE to transmit UL data, $P_t \in]0, \ldots, P_{\max}]$, $P_{\max}$ being the maximum transmission power of the UE. P_r is the power consumed by the UE to receive DL data. $P_{t'}$ is the transmission power used at the cloud enabled SCeNB to send the processed result to the UE. The transmitted data packets have equal size of $L_{\mathrm{UL}} = l$ bits and the received processed packets have equal size of L_{DL} bits. W_{UL} and W_{DL} are respectively the bandwidths allocated for these UL and DL transmissions. P_w is the power consumed

by the UE while waiting for the processing at the SCeNBce and T_w is the corresponding time spent by the SCeNBce to execute 1 offloaded packet. Note that for simplicity, only radiated power is considered in the power consumption in (2).

As aforementioned, processing decisions occur at fixed time instants with slot duration T. Thus, the offloading decision can be possible only if there exists $P_t \in]0, \dots, P_{\max}]$ to transmit the offloaded packets $u \leq q$ such that the processing at the SCeNBce is achieved within a time T. This condition can be expressed by

$$\frac{uL_{\mathrm{UL}}}{W_{\mathrm{UL}}\log_2\left(1+\frac{P_t x}{W_{\mathrm{UL}}N_0}\right)}+\frac{uL_{\mathrm{DL}}}{W_{\mathrm{DL}}\log_2\left(1+\frac{P_{t'} x}{W_{\mathrm{DL}}N_0}\right)}+uT_w \leq T. \tag{3}$$

Satisfying the equality allows to compute the transmit power P_t at the UE and define the maximal number of offloaded packets M_o with respect to the channel conditions.

3- *Idle*: The mobile user stays in an idle state without processing neither locally, nor remotely, and waits the next processing decision time, $u = 0$. Though no data is processed, the UE transmission circuitry is switched on and hence it spends energy equal to

$$E = P_{\mathrm{idle}} \times T. \tag{4}$$

III. Problem Formulation

According to the described system model and defined system states $s = (q, x)$, the main objective of the SCM is to make decisions by finding the optimal scheduling-offloading strategy that minimizes the energy consumption at the user terminal subject to delay constraints tolerated by the application. Let μ denote this optimal policy. It is a sequence of control decisions (actions) that specify at each time slot the number of packets u to be processed based on the past history of the system states and the past decisions. Given the statistical knowledge of this history, the system state in the next time slot depends only on the current state and the current control decision, making thus the random state process $\{s\}$ a discrete time Markov chain. Therefore, the scheduling-offloading optimization problem can be formulated as a constrained Markov decison process CMDP characterized by the following parameters [10], [11]:

- *State space*: The state space $\mathcal{S}$ is the set of system states $s = (q, x)$ composed of buffer and channel states. It is a discrete finite space with $|\mathcal{S}| = |\mathcal{Q}| \times |\mathcal{X}|$.

- *Action space*: The action space $\mathcal{U}$ corresponds to the space defining the control actions u given by the three different decisions of local processing, offloading or staying idle. The action space is also a discrete finite space and has cardinality $|\mathcal{U}| = M_l + M_o + 1$. M_l and M_o are respectively the maximum number of packets that can be processed locally or remotely within time T as described in the previous section.

- *Transition Probability*: The state transition probability of the MDP is defined by $p\,(s'|s, u)$, the probability to go from state $s = (q, x)$ to the next state $s' = (q', x')$ when action $u \in \mathcal{U}$ is processed. Assuming that the buffer state (queue length) and the channel state (channel fading) are independent of each other and that the channel states are not correlated, the probability of state transition is defined by

$$p\,(s'|s, u) = p(q'|q, u)p(x') = p(a)p(x'), \tag{5}$$

where the next buffer state q' is defined according to the current buffer state q, the number of data packets' arrivals a and the number of processed packets u at state s as $q' = q - u + a$. $p(x')$ is the probability distribution of the quantized channel states and $p(a)$ is the probability distribution of the packet arrivals. Given that the packets' arrival follows a Poisson distribution with average arrival rate γ, the probability of generating a packets is $p(a) = e^{-\gamma}\frac{\gamma^a}{a!}$. Then, the transition probability is expressed as

$$p\,(s'|s, u) = p(x')e^{-\gamma}\frac{\gamma^{(q'-q+u)}}{(q'-q+u)!}. \tag{6}$$

For this CMDP, the optimal scheduling-offloading policy μ is then a mapping function from the state space $\mathcal{S}$ to the action space $\mathcal{U}$. It is a stationary policy that does not depend on the time at which the decision is made. In an infinite horizon, the control policy μ aims at minimizing the average consumed energy at the UE side while satisfying quality of service (QoS) requirements in terms of average delay experienced by the UE packets before being processed and average buffer overflow constraint. The *cost function* and the *constraint functions* of this infinite horizon CMDP problem are defined in the following.

The average energy cost under the strategy μ is given by

$$\bar{E}^{\mu} = \lim_{N\to\infty}\frac{1}{N}\mathbb{E}^{\mu}\left[\sum_{n=1}^{N} C_E(s_n, u_n)\right], \tag{7}$$

where $\mathbb{E}^{\mu}$ is the expectation with respect to μ. At a given time slot $n, n = [1, \dots, N]$, the system state is denoted s_n and $\mu(s_n) = u_n$ is the action deciding the number u_n of packets to be processed. Then, $C_E(s_n, u_n)$ is the instantaneous energy cost when the action u_n is performed at state s_n. This instantaneous average cost is given by $C_E(s_n, u_n) = E$ as derived in (1), (2) or (4) according to the three possible actions.

The average delay constraint under strategy μ can be formulated by Little's Law as an average queue length of the user buffer [9]. Let $C_Q(s_n, u_n)$ be the instantaneous delay cost experienced by the UE packets at state s_n under action u_n. It is defined by $C_Q(s_n, u_n) = C_Q(s_n) = q_n$, and hence the time averaged delay constraint is given by

$$\bar{Q}^{\mu} = \lim_{N\to\infty}\frac{1}{N}\mathbb{E}^{\mu}\left[\sum_{n=1}^{N} C_Q(s_n, u_n)\right] = \lim_{N\to\infty}\frac{1}{N}\mathbb{E}^{\mu}\left[\sum_{n=1}^{N} q_n\right] \tag{8}$$

The average buffer overflow constraint is defined according to an overflow probability of the buffer overflow events as mentioned in the computation model. Let $C_O(s_n, u_n)$ be the instantaneous buffer overflow cost at state s_n under action u_n. The overflow probability is given by

$$\bar{O}^{\mu} = \lim_{N\to\infty}\frac{1}{N}\mathbb{E}^{\mu}\left[\sum_{n=1}^{N} C_O(s_n, u_n)\right] = \lim_{N\to\infty}\frac{1}{N}\mathbb{E}^{\mu}\left[\sum_{n=1}^{N} \mathbf{1}_{(q_n > B)}\right] \tag{9}$$

Then, the scheduler objective is to find the optimal strategy which resolves the constrained problem by minimizing the average energy consumed at the UE while satisfying an average delay on the processed packets, *i.e.,* a queue length below a

certain value δ_Q, and a buffer overflow probability under a fixed threshold δ_O. The optimization problem can be stated as

$$\begin{aligned} \mu^* &= \min_{\mu} \bar{E}^{\mu} \\ \text{s.t.} \quad & \bar{Q}^{\mu} \leq \delta_Q \quad \text{and} \quad \bar{O}^{\mu} \leq \delta_O. \end{aligned} \tag{10}$$

IV. Offline Dynamic Programming Strategies

Two main approaches have been proposed to resolve an infinite horizon average cost problem [9]–[11]: offline and online dynamic programming strategies. The online approach is based on learning from past to find a policy that can cope with unknown environment [9]. In [12], we have studied the online learning approach for our problem and have shown that it is suboptimal in terms of energy consumption due to its imperfect knowledge on the data arrival and the channel state distributions. In this paper, we investigate the offline approach that requires *a priori* knowledge of the channel statistics and the application properties. This knowledge can be acquired in advance or estimated over sufficiently long period before running the application. Then, the pre-calculated offline strategy can provide an accurate modelling of the state transition probabilities of the constrained problem, leading thus to optimality. Two offline solutions are proposed in the sequel, namely: deterministic offline strategy and randomized offline strategy.

A. Deterministic Offline Strategy

The deterministic offline policy consists in a deterministic mapping from the state space $\mathcal{S}$ to the action space $\mathcal{U}$, associating for each state s a unique action u that is performed when this state is visited. This scheduling-offloading policy, denoted μ, determines the number u_n of packets to be processed as $u_n = \mu(s_n)\ \forall n \in \mathbb{N}$.

In order to find the optimal deterministic policy μ^*, the CMDP problem can be converted into an unconstrained problem using Lagrangian approach as

$$\begin{aligned} &\mathcal{L}(\mu, \lambda_1, \lambda_2) = \bar{E}^{\mu} + \lambda_1(\bar{Q}^{\mu} - \delta_Q) + \lambda_2(\bar{O}^{\mu} - \delta_O) \\ &= \lim_{N\to\infty} \tfrac{1}{N}\mathbb{E}^{\mu}[\textstyle\sum_{n=1}^{N} C_E(s_n, u_n) + \lambda_1(q_n - \delta_Q) \\ &\qquad +\lambda_2(\mathbf{1}_{(q_n > B)} - \delta_O)], \end{aligned} \tag{11}$$

where λ_1 and λ_2 are the Lagrange multipliers corresponding to the delay and overflow constraints, and an instantaneous cost per stage is defined as

$$\begin{aligned} C_{\mathcal{L}}(s_n, u_n, \lambda_1, \lambda_2) &= C_E(s_n, u_n) + \lambda_1(q_n - \delta_Q) \\ &+ \lambda_2(\mathbf{1}_{(q_n > B)} - \delta_O). \end{aligned} \tag{12}$$

Satisfying in addition the saddle point optimality condition [9], the problem of (10) can be reformulated as

$$(\mu^*, \lambda_1^*, \lambda_2^*) = \operatorname*{argmax}_{\lambda_1 \geq 0, \lambda_2 \geq 0} \operatorname*{argmin}_{\mu} \mathcal{L}(\mu, \lambda_1, \lambda_2), \tag{13}$$

where the policy μ^* is optimal for the constrained problem for given values λ_1 and λ_2. λ_1^* and λ_2^* are the optimal Lagrange multipliers and $(\mu^*, \lambda_1^*, \lambda_2^*)$ is the saddle point that minimizes the Lagrangian.

The minimization problem is handled by solving dynamic programming equations, known as Bellman optimality equations using Relative Value Iteration Algorithm (RVIA) [11]. For fixed values of Lagrange multipliers λ_1 and λ_2, the RVIA computes the optimal value function $V^*_{\lambda_1,\lambda_2}(s)$ for a state $s \in \mathcal{S}$ iteratively by satisfying the Bellman equation as

$$V_{\lambda_1,\lambda_2}(s) = \min_{u} \left[C_{\mathcal{L}}(s, u, \lambda_1, \lambda_2) + \sum_{s' \in \mathcal{S}} p\left(s'|s,u\right) V_{\lambda_1,\lambda_2}(s') - \beta \right] \tag{14}$$

where β is the optimal cost per stage [11].

Then, for fixed values of λ_1, λ_2, the optimal policy $\mu^*_{\lambda_1,\lambda_2}$ can be obtained using policy iteration approach based on two steps executed simultaneously:

1- Policy evaluation: Given the policy $\mu^i_{\lambda_1,\lambda_2}$, and starting from an initial state s_0, compute the corresponding cost to go (value function) from all the states s on the infinite horizon. This can be achieved by using iteratively the RVIA algorithm to find:
- the average cost per stage $\beta^i_{\lambda_1,\lambda_2}$ for which $V^i_{\lambda_1,\lambda_2}(s_0) = 0$
- the average cost to go for each state $s \neq s_0$ as

$$\begin{aligned} V^i_{\lambda_1,\lambda_2}(s) &= C_{\mathcal{L}}(s, \mu^i_{\lambda_1,\lambda_2}(s), \lambda_1, \lambda_2) \\ &+ \sum_{s' \in \mathcal{S}} p\left(s'|s, \mu^i_{\lambda_1,\lambda_2}(s)\right) V^i_{\lambda_1,\lambda_2}(s') - \beta^i_{\lambda_1,\lambda_2} \end{aligned}$$

2- Policy improvement: Obtain a new policy $\mu^{i+1}_{\lambda_1,\lambda_2}$ allowing a better average cost

$$\mu^{i+1}_{\lambda_1,\lambda_2}(s) = \operatorname*{argmin}_{\mu} \left[C_{\mathcal{L}}(s, \mu, \lambda_1, \lambda_2) + \sum_{s' \in \mathcal{S}} p\left(s'|s,\mu\right) V^i_{\lambda_1,\lambda_2}(s') \right]$$

The policy iteration algorithm iterates until satisfying: $\mu^i_{\lambda_1,\lambda_2} = \mu^{i+1}_{\lambda_1,\lambda_2} = \mu^*_{\lambda_1,\lambda_2}$ and thus the optimal policy for a fixed pair (λ_1, λ_2) is given by

$$\mu^*_{\lambda_1,\lambda_2} = \operatorname*{argmin}_{\mu} \mathcal{L}(\mu, \lambda_1, \lambda_2). \tag{15}$$

In order to find now the optimal Lagrange multipliers λ_1^*, λ_2^*, an iterative gradient descent technique is used in an outer loop to update these parameters as

$$\lambda_1^{n+1} = \lambda_1^n + \zeta_1 \left(\bar{Q}_n^{\mu^*_{\lambda_1^n,\lambda_2^n}} - \delta_Q \right) \tag{16}$$

$$\lambda_2^{n+1} = \lambda_2^n + \zeta_2 \left(\bar{O}_n^{\mu^*_{\lambda_1^n,\lambda_2^n}} - \delta_O \right) \tag{17}$$

where ζ_1, ζ_2 are decreasing increment coefficients and $\bar{Q}^{\mu^*_{\lambda_1,\lambda_2}}$ and $\bar{O}^{\mu^*_{\lambda_1,\lambda_2}}$ are the average delay cost and the average buffer overflow cost, respectively, according to the optimal policy $\mu^*_{\lambda_1,\lambda_2}$. Under this policy, these average costs are evaluated with respect to a state distribution $\rho^{\mu^*_{\lambda_1,\lambda_2}}(s)$ as

$$\bar{Q}^{\mu^*_{\lambda_1,\lambda_2}} = \sum_{s \in \mathcal{S}} \rho^{\mu^*_{\lambda_1,\lambda_2}}(s) C_Q(s, \mu^*_{\lambda_1,\lambda_2}(s)) \tag{18}$$

$$\bar{O}^{\mu^*_{\lambda_1,\lambda_2}} = \sum_{s \in \mathcal{S}} \rho^{\mu^*_{\lambda_1,\lambda_2}}(s) C_O(s, \mu^*_{\lambda_1,\lambda_2}(s)) \tag{19}$$

with the state distribution defined as function of the state transition probability under policy $\mu^*_{\lambda_1,\lambda_2}$ and is expressed $\forall s \in \mathcal{S}$ as

$$\rho^{\mu^*_{\lambda_1,\lambda_2}}(s) = \sum_{s' \in \mathcal{S}} \rho^{\mu^*_{\lambda_1,\lambda_2}}(s') p\left(s|s', \mu^*_{\lambda_1,\lambda_2}(s)\right). \tag{20}$$

B. Randomized Offline Strategy

The randomized offline policy consists in a probabilistic mapping from the state space $\mathcal{S}$ to the action space $\mathcal{U}$, associating for each state s an action $u = \mu(s)$ that is a random variable with a probability distribution uniquely associated to the visited state. Therefore, the offline randomized strategy μ relies on weighting each (state s, action u) pair with a given probability called the occupation measure. This measure represents the probability to visit each (s, u) pair according to the devised strategy [10].

Let p_0 be the probability distribution of the initial state s_0. For an infinite length trajectory, the occupation measure $\rho^{\mu}(s, u)$ of a state-action pair (s, u) when the policy μ is employed corresponds to the fraction of time slots where the state s is visited and the action $u = \mu(s)$ is performed under the condition that the initial state s_0 of the trajectory is drawn according to p_0. It is defined by

$$\rho^{\mu}(s,u) = \lim_{N\to\infty} \frac{1}{N} \mathbb{E}^{\mu} \left[\sum_{n=1}^{N} \mathbf{1}_{(s_n=s, \mu(s_n)=u_n)} \right] \quad (21)$$

The optimal occupation measure can be obtained by formulating the CMDP optimization problem (10) as a linear programming problem [10] as

$$\rho^* = \underset{\mu}{\operatorname{argmin}} \sum_{s\in\mathcal{S}, u\in\mathcal{U}} \rho(s,u) C_E(s,u)$$

$$\text{s.t.} \quad \sum_{s\in\mathcal{S}, u\in\mathcal{U}} \rho(s,u) q \le \delta_Q \ , \quad \sum_{s\in\mathcal{S}, u\in\mathcal{U}|q>B} \rho(s,u) \le \delta_O$$

$$\sum_{s'\in\mathcal{S}, u\in\mathcal{U}} \rho(s',u) \left(\mathbf{1}_{(s'=s)} - p(s|s',u) \right) = 0 \quad \forall s \in \mathcal{S} \quad (22)$$

where the last line results from the Markov property of the process (s_n, u_n). An optimal randomized policy can be then obtained from ρ^* the solution of the above problem by choosing a random action u at a given state s according to the following probability

$$p_s(u) = \frac{\rho^*(s,u)}{\sum_{u'\in\mathcal{U}} \rho^*(s,u')}. \quad (23)$$

V. Numerical Results

In this section, we evaluate by numerical results the deterministic and randomized offline strategies devised for the scheduling-computation offloading problem. We consider the single cell single user scenario described in section II with the following characteristics. The allocated bandwidths for UL and DL transmissions are equal to $W_{\mathrm{UL}} = 500$KHz and $W_{\mathrm{DL}} = 5$MHz and the time slot duration is $T = 2$ ms. For the communication channel model, the channel state z follows an exponential distribution with mean $\tau = 0.47$ and the quantized channel state x can take 8 possible values from the finite space $\mathcal{X} = [-13, -8.47, -5.41, -3.28, -1.59, -0.08, 1.42, 3.18]$ dB. All the powers are normalized by the signal to noise ratio experienced by the SCeNB when unitary power signal is sent at the UE side. This is equivalent to normalizing by WN_0/α where α is the path-loss of the channel between the UE and the SCeNB without fading. The numerical values of these powers at the user are given by: $P_u = 0.7, P_{\max} = 6, P_r = 0.5, P_{\mathrm{idle}} = 0, P_w = 0.2$. At the SCeNBce, the transmission power is $P_{t'} = 20$ and the time spent to process one packet is $T_w = 0.1$ms.

For the application model, the data arrival follows a Poisson distribution with an average rate γ bits/sec and packets' size equal to $l = 500$ bits while the user's buffer is of size $B = 50$ packets. Then, the packets to be processed have equal size $L_{\mathrm{UL}} = l = 500$ bits and the resulting packets of remote processing of the offloaded ones have size $L_{\mathrm{DL}} = 5000$ bits. We assume that, within a time slot T, the maximum number of packets that can be processed locally is $M_l = 1$ and the maximum number of offloaded packets that can be processed remotely is $M_o = 5$. For the system constraints, we consider an application latency that corresponds to an average queue constraint of $\delta_Q = 10$ packets and we tolerate an overflow probability of $\delta_O = 10^{-5}$.

In Figure 1, we illustrate the average consumed energy as function of the data arrival rates for different offline strategies. We compare the deterministic dynamic programming policy denoted "Dynamic programming policy" and the randomized dynamic programming policy denoted "Linear programming policy" to three other policies that maintain the same average queue length. The "Only offloading dynamic policy" is based on the deterministic dynamic programming policy but takes only offloading decisions to process the packets remotely at the SCeNBce. The "Only local processing" and "Immediate scheduling" policies are however static policies. The former one consist in processing locally at the UE all the packets that exceed the average queue constraint δ_Q. The latter one decides to process u packets when the queue length exceeds δ_Q according to: $u = \max\left(\min(q + a - \delta_Q, M_o), 0\right)$. If $u = M_l = 1$, it can choose either to process locally or to offload depending on the action that minimizes the instantaneous consumed energy. On the other side, if $u > M_l$, it performs only offloading.

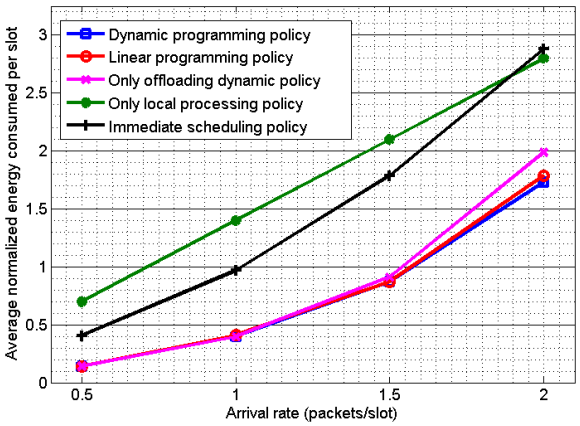

Fig. 1. Average Energy Consumption for different policies

We can observe that the deterministic dynamic programming strategy achieves the optimal performance of the randomized linear programming strategy and gives the lowest energy consumption. Indeed, for these strategies, the system has perfect knowledge of the states transitions. It can adapt its processing decisions according to the arrival rate and the channel conditions while satisfying the imposed delay constraints. Thus, more packets can be offloaded in good channel states whereas transmission is limited and local processing or

idle mode are decided in case of bad channel states. The only offloading dynamic policy has close performance to the latter strategies for low and moderate arrival rates but consumes more energy for high arrival rates. This is due to the fact that the packets are offloaded for remote processing even when the channel is in bad conditions since local processing decisions are not possible. We can notice also that both static policies consume much more energy than dynamic ones since they do not exploit the radio link quality. Idle mode is not allowed in case of high arrival rates and bad channel states.

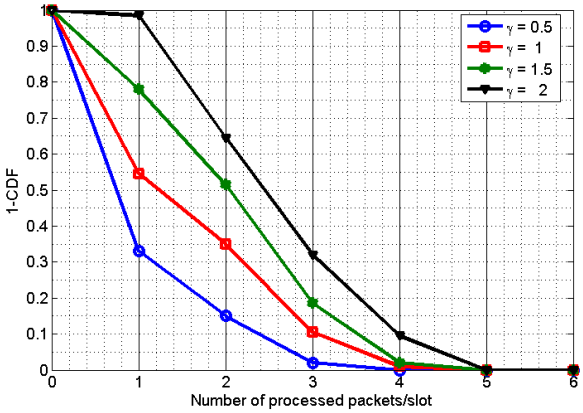

Fig. 2. CDF of the processed packets per slot

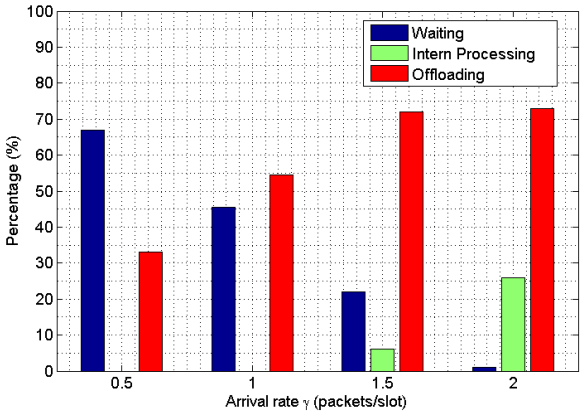

Fig. 3. Percentage of processing decisions for different arrival rates

Figure 2 shows the cumulative distribution function CDF of the number of processed packets per slot for different average arrival rates under the offline dynamic programming policy. It can be seen that more packets are processed for higher arrival rates. This result is also observed in Figure 3 that presents the processing decisions distribution for different arrival rates. For high arrival rates (2 packets/slot), we can notice that offloading decisions are favoured for most of the time. Indeed, the arrival rate approaches, in this case, the maximal rate of offloading (2.5 packets/slot) according to the considered channel distribution. Therefore, under delay and overflow constraints, the offline strategy makes decisions to process the queued packets while taking benefit of the channel conditions. The packets are then offloaded when channel states are good and processed locally under bad channel conditions, limiting hence the idle mode. On the other hand, for low and moderate arrival rates (0.5, 1 and 1.5 packets/slot), Idle mode is decided whenever the channel is in bad states and offloading decisions are made when the channel conditions allow for better energy saving, limiting thus the energy-consuming local processing.

VI. Conclusion

In this paper, we have addressed computation offloading problem from a mobile user to its serving resourcefull cloud enabled SCeNB. We have proposed to jointly optimize the radio resource scheduling and offloading in order to minimize the average energy consumed by the mobile terminal to process its application under average delay constraints. Both deterministic and randomized offline policies based dynamic programming have been studied. The offline dynamic strategies can benefit from their prior knowledge on the application rates and the channel statistics to perform offloading decisions in good channel states and local processing or staying idle under bad channel conditions. Hence, they have shown through numerical results optimal energy saving gains compared to only offloading and static processing strategies.

Acknowledgment

This work was supported by the European Commission 7^{th} Framework Program Project TROPIC, under grant Nr. 318784.

References

[1] V. Chandrasekhar and J. Andrews, "Femtocell Networks: A Survey," *IEEE Communication Magazine*, vol. 46, no. 9, pp. 59–67, Sept. 2008.

[2] K. Kumar and Y.-H. Lu, "Cloud Computing for Mobile Users: Can Offloading Computation Save Energy," *IEEE Computer*, vol. 43, no. 4, pp. 51–56, Apr. 2010.

[3] K. Kumar, J. Liu, Y.-H. Lu, and B. Bhargava, "A Survey of Computation Offloading for Mobile Systems," *Mobile Networks and Applications, Springer Science*, vol. 18, no. 1, p. 129140, Feb. 2013.

[4] "TROPIC: Distributed computing, storage and radio resource allocation over cooperative femtocells," http://www.ict-tropic.eu.

[5] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. Wu, "Energy-Optimal Mobile Cloud Computing under Stochastic Wireless Channel," *IEEE Trans. Wireless Communications*, vol. 12, no. 9, p. 45694581, Sept. 2013.

[6] M. V. Barbera, S. Kosta, A. Mei, V. C. Perta, and J. Stefa, "To offload or not to offload? the bandwidth and energy costs of mobile cloud computing," *Proc. IEEE International Conference on Computer Communications (INFOCOM13)*, Apr. 2013.

[7] O. Munoz, A. P.-Iserte, and J. Vidal, "Optimization of Radio and Computational Resources for Energy Efficiency in Latency-Constarined Application Offloading," *IEEE Trans. on Vehicular Technology*, no. 99, Nov 2014.

[8] S. Barbarossa, S. Sardellitti, and P. Di Lorenzo, "Communicating While Computing: Distributed mobile cloud computing over 5G heterogeneous networks," *IEEE Signal Processing Magazine*, vol. 31, no. 6, pp. 45–55, Nov. 2014.

[9] N. Salodkar, A. Bhorkar, A. Karandikar, and V.S. Borkar., "An on-line learning algorithm for energy efficient delay constrained scheduling over a fading channel," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 4, p. 732742, May 2008.

[10] E. Altman, *Dynamic Programming and Optimal Control*. Chapman and Hall/CRC.

[11] D. P. Bertsekas, *Dynamic Programming and Optimal Control*. Athena Scientific, 4th Edition, 2005.

[12] M. Kamoun, W. Labidi, and M. Sarkiss, "Joint Resource Allocation and Offloading Strategies in Cloud Enabled Cellular Networks," *IEEE International Conference on Communications (ICC) 2015*.