Optimal Computation Offloading in Edge-Assisted UAV Systems

Davide Callegaro and Marco Levorato
The Donald Bren School of Information and Computer Sciences, UC Irvine, CA, US
e-mail: {dcallega, levorato}@uci.edu

***Abstract*—The ability of Unmanned Aerial Vehicles (UAV) to autonomously operate is constrained by the severe limitations of on-board resources. The limited processing speed and energy storage of these devices inevitably makes the real-time analysis of complex signals – the key to autonomy – challenging. In urban environments, the UAV can leverage the communication and computation resources of the surrounding city-wide Internet of Things infrastructure to enhance their capabilities. For instance, the UAVs can interconnect with edge computing resources and offload computation task to improve response time to sensor input and reduce energy consumption. However, the complexity of the urban topology and the large number of devices and data streams competing for the same network and computation resources create an extremely dynamic environment, where poor channel conditions and edge server congestion may penalize the performance of task offloading. This paper develops a framework enabling optimal offloading decisions as a function of network and computation load parameters and current state. The optimization is formulated as an optimal stopping time problem over a Markov process.**

***Index Terms*—Edge Computing, Urban Internet of Things, Unmanned Aerial Vehicles, Autonomous Systems.**

I. Introduction

The use of Unmanned Aerial Vehicles (UAV) is being increasingly proposed for a wide spectrum of applications, including surveillance and monitoring, disaster management, agriculture, and network coverage extension [1]–[5]. Their autonomous operations require the acquisition and real-time analysis of information from the surrounding environment. However, processing information-rich signals, such as video and audio input, to inform navigation and, in general, autonomous decision making, is an extremely demanding task for these constrained platforms. In fact, due to the limited on-board computation resources, the analysis process may require a significant amount of time, thus decreasing the UAV responsiveness to stimuli. Additionally, continuously running heavy-duty analysis algorithms imposes a considerable energy expense burden to these battery-powered devices. In summary, the degree of autonomy of UAVs may be limited, and may come at the price of a reduced operational lifetime.

In urban environments, the UAV can leverage the resources of the surrounding urban Internet of Things (IoT) infrastructure to overcome some of its limitations and enhance its capabilities. For instance, the UAV can use the communication infrastructure to connect to edge servers – that is, compute-capable machines positioned at the network edge – to offload data processing tasks [6]. By offloading the computation task to a more powerful device, the UAV can possibly reduce the capture-to-decision time, thus improving its reaction time to sensor input. Additionally, offloading data processing to an edge server can reduce the amount of energy needed to complete the mission.

However, the urban IoT is a highly dynamic system, where a myriad of devices, and data streams, compete for the available communication and computation resources. As a result, the network connecting the UAV and edge server may be congested, and the transportation of the data to the edge server may require a large time. Additionally, the topology of urban areas may degrade the capacity of the channel due to path loss and shadowing. Finally, the edge server may have a queue of computation tasks from other devices and services that need to be completed before processing the data from the UAV. Therefore, in some conditions and locations, the time needed to transport the data to the edge server and receive the outcome of analysis may exceed that of local processing at the UAV.

In this paper, we present an optimization process through which the UAV decides whether to process locally or offload the computation task to the edge server. The decision is based on a series of interactions between the UAV and the IoT system, where the UAV receives feedback on the state of the network and edge server, which allows the estimation of the residual time to task completion. Based on this information, the UAV solves an optimization problem aiming at the minimization of a weighted sum of delay and energy expense.

Numerical results, which are based on parameters extracted from a real-world implementation of the system, demonstrate that the proposed intelligent and sequential probing technique effectively adapts the processing strategy to the instantaneous state of the network-edge server system. The outcome is a reduced processing delay and energy expense, two extremely important metrics in the considered application.

The rest of the paper is organized as follows. Section II provides an overview of the system considered in this paper. Section III describes in detail the parameters and operations of the UAV-edge server system. Section IV introduces a Markovian description of the system's dynamics and formulates and solves the problem for the optimization of the offloading decisions. Section V presents numerical results illustrating the performance of the proposed adaptive offloading strategy. Section VI concludes the paper.

II. System and Problem Overview

We consider a scenario where a UAV autonomously navigate an urban environment. The UAV is assigned the task

This work was partially supported by the NSF under grant IIS-1724331.

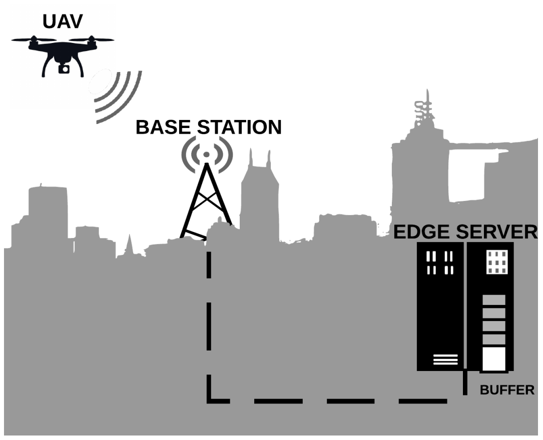


Figure 1: Illustration of the considered scenario and system: a UAV interconnects with an edge server through a low latency wireless link to offload computation tasks. Poor channel conditions and high processing load at the edge server may result in a larger delay and energy expense compared to local on-board processing.

to acquire and process complex data in predefined locations within the city, where the outcome of processing may influence sensing and navigation actions. A relevant case-study application is city-monitoring, in which the UAV captures a panoramic sequence of pictures at each location and process them using a classification algorithm to detect objects or situations of interest. In case of positive detection, the UAV may stay at the location to capture more detailed or higher-resolution pictures of a specific portion of its view.

Intuitively, processing information-rich signals using a computation-intense algorithm is a challenging task for inherently constrained platforms such as UAVs. In fact, the limited processing power of the on-board computation resources results in long capture-to-output time of the algorithm, which decreases responsiveness to stimuli and increases mission time. Additionally, on-board processing consumes a significant amount of energy, even when compared to motion and navigation, thus shortening the lifetime of these battery-powered systems. Note that in the scenario described above, the UAV is hovering while waiting for the classification algorithm to complete, as the outcome will determine its subsequent action. Thus, a large processing time incurs at additional energy expense penalty associated with longer flight time.

The UAV can leverage the resources of the surrounding urban IoT infrastructure to improve its performance. In the scenario at hand, edge servers placed at the network edge can take over the task of processing the data acquired by the UAV. Intuitively, the larger processing power of edge servers compared to that of UAVs grants a much faster completion of the processing task, thus allowing a faster decision making and a smaller capture-to-decision time. Additionally, the UAV would be relieved from the energy expense burden of processing, at the price of energy expense associated with data transmission. We remark that a shorter time to receive the output of the classification algorithm also corresponds to a smaller energy expense associated with hovering.

However, as noted in the introduction, the urban IoT is a highly dynamic environment, where a myriad of data streams and services coexist and compete for the same communication resources. In the considered scenario, the wireless channel connecting the UAV to a wireless access point may have a low capacity due to the physical properties of signal propagation, but also due to the existence of interfering communications which use part of the time/frequency channel resource. Additionally, the edge servers may be serving other devices offloading their computation tasks, and the UAV task may suffer queuing delay, or a reduced processing speed. As a result, in certain conditions, offloading the computation task to an overloaded edge server connected to the UAV through a poor communication channel may lead to a longer capture-to-decision time. Again, this corresponds to less efficient mission operations, but also a possibly large energy expense due to hovering while waiting for a response.

In order to fully harness the possible performance gain granted by the available resources provided by the urban IoT infrastructure, the UAV needs to make informed decisions whether or not to offload computation. To this aim, we equip the UAV with the ability to interact with the surrounding network and edge devices and acquire information regarding the status of the communication and processing pipeline. The information is used to evaluate the progress of the task and predict the future cost of the binary decision between local and edge-assisted computing.

III. System Model

In this section, we formalize and discuss an abstraction of the system composed of the UAV, a network access point and an edge server. We divide the description into modules focusing on the communication, computation and energy expense aspects of the system.

A. Communications

The UAV is connected to the network access point through a wireless channel of finite capacity. The data to be transferred for offloading have size L-bits. The UAV transmits with fixed power P and rate R in the finite set of $K{+}1$ transmission rates $\{R_0, R_1, R_2, \ldots, R_K\}$, where $R_0{=}0$ corresponds to disconnection from the network, and thus no data transmission. The link between the UAV and the AP is a wireless link affected by path loss, fading and noise. The SNR at the receiver is

$$\text{SNR} = \frac{gP}{\sigma^2}, \tag{1}$$

where σ^2 is the noise power and g is the channel attenuation coefficient including path loss and fading. We assume exponential path loss and Rayleigh flat fading. Thus, the distribution of g is

$$\Theta_g(x){=}\Pr(g \leq x){=}1{-}e^{-\frac{x}{\gamma}}, \tag{2}$$

where γ is the path loss.

Assuming channel knowledge and a capacity achieving scheme, the selected transmission rate of the UAV is equal to R_i bits/s if $g \in (g_i, g_{i+1}]$, where

$$g_i = g : R_i = \mathcal{C}(g\text{SNR}), \quad i{=}1, \dots, K, \tag{3}$$

and

$$\mathcal{C}(x){=}\log(1+x). \tag{4}$$

The resulting transmission time is L/R_i s.

B. Computation

The time to complete the computation task locally at the UAV and at the edge server are captured using the random variables X' and X, respectively. We assume that X' and X follow an exponential distribution of rate μ' and μ tasks/s, respectively. The edge server accumulates incoming computation tasks in a finite buffer of size B tasks. Excluding the task generated by the UAV, tasks arrive according to a Poisson process of rate λ tasks/s, with $\lambda{<}\mu$.

C. Energy

As described in the previous section, at each predefined location the UAV captures the data, and then completes the computation task – either locally or at the edge server – while hovering maintaining the position. We define a rate of energy expense for the two fundamental operational blocks that are influenced by the offloading decision: processing and hovering. Specifically, we define P_{P} and P_{H} as the Watts used to respectively process the data and hover. As mentioned earlier, the transmission power is equal to P Watts.

IV. Optimal Offloading Decisions

In the considered scenario, the two most relevant performance metrics are energy expense E and time T per location. Herein, we assume the state of the system at each location to be independent. Importantly, the costs E and T are a function of the offloading decision, that is, whether the computation task is completed at the UAV or at the edge server.

Given the knowledge of the system parameters, the UAV can compute the average cost E and time T corresponding to each of the two options, where the average is over realizations of the stochastic process associated with the system dynamics. However, within that average there are realizations in which offloading is advantageous (high channel capacity and low processing congestion) or disadvantageous (low channel capacity and high processing congestion). In order to fully harness the performance gain edge computing can provide while facing the dynamics of the IoT system, we develop a sequential probing and decision making framework. At each stage, the UAV observes the current realization, estimates the residual cost to complete the operations, and makes a decision about whether to initiate local processing or not. This formulation corresponds to an optimal stopping time problem on a semi-Markov process.

Under the assumptions listed in the previous section, the temporal evolution of the system can be represented as a semi-Markov process. Let's define as t_j^+, $j{=}0,1,2,\dots$ the time instants right after the occurrence of an event, defined as the establishment of the connection with the network, the delivery of the data to the edge server, or the completion of a computation task at the UAV or edge server. We denote the state of the system at time t_j^+ as the random variable $S(t_j^+)$. The state space $\mathcal{S}$ of $S(t_j^+)$ consists of an initial state s_0, two termination states s_{UAV} and s_{ES}, and a number of states describing data transmission and task queueing process. The termination states correspond to the computation task being completed locally at the UAV (s_{UAV}) and offloaded to the edge server (s_{ES}). Specifically, we include *(i)* a set of $K{+}1$ states $R_0, R_1, \dots, R_K$ associated with a transmission rate, that is, a channel state in the ranges defined in the previous section; and *(ii)* a set of $C{+}1$ states $B_1, \dots, B_{C+1}$ associated with the position of the UAV task in the task buffer at the edge server. Note that B_{C+1} corresponds to a full buffer at arrival, that is, the UAV task is rejected. It can be shown that the process $\mathbf{S}{=}(S(t_j^+))_{j=0,1,\dots}$ is a Markov process.

At each time instant t_j^+, the UAV is notified of the state $S(t_j^+)$ from the network access point or the edge server, and makes a binary decision $u{\in}\{0,1\}$, where 0 and 1 correspond to local computing and continuing on the edge-assisted pipeline – that is, further deferring local computing, respectively.

A. Transition Probabilities

We now describe the transition probabilities governing the dynamics of the stochastic process $\mathbf{S}$. For the sake of notation clearness, we denote the time t_j^+ with its index j. We define, then

$$P(s'|s,u){=}\Pr(S(j{+}1){=}s'|S(j){=}s, U(j){=}u). \tag{5}$$

If the decision is equal to 0, the transition probabilities from any state s are

$$P(s'|s,0){=}\begin{cases}1 & \text{if} \quad s'{=}s_{\text{UAV}};\\ 0 & \text{otherwise.}\end{cases} \tag{6}$$

That is, if the decision is to compute locally, the process moves to state s_{UAV} deterministically from any state.

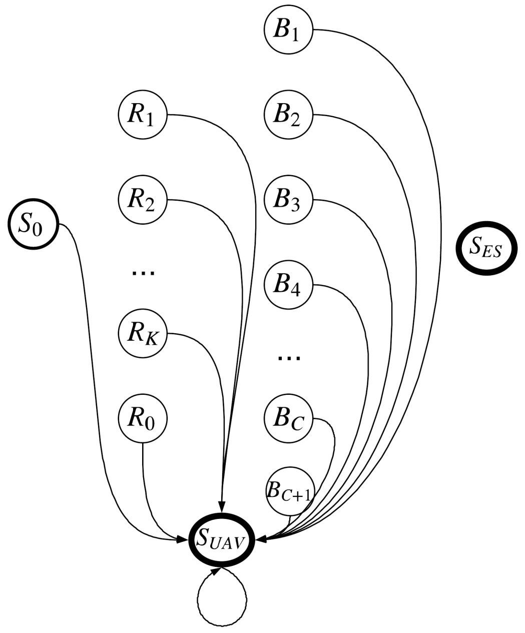
We, then, analyze the transition probabilities if the decision is 1, that is, the UAV further defers the initiation of local computation. In such case, from the initial state s_0, the channel distribution is sampled, and the state moves to one of the pre-transmission states R_i with probability equal to that of the associated interval. Thus,

$$P(s'|s_0,1){=}\begin{cases}\pi_i & \text{if} \quad s'{=}R_i, \ i{=}0,1,\dots,K;\\ 0 & \text{otherwise,}\end{cases} \tag{7}$$

where $\pi(i){=}\Theta_g(g_{i+1}) - \Theta_g(g_i)$.

In any state R_i, the UAV is reported the transmission rate, that is, the index i, from the wireless access point. If the decision is to defer local processing, the transition probabilities from R_i, i=1,…,K, are

$$P(s'|R_i,1){=}\begin{cases}\sigma_{c-1} & \text{if } s'{=}B_c\\ 0 & \text{otherwise.}\end{cases} \tag{8}$$



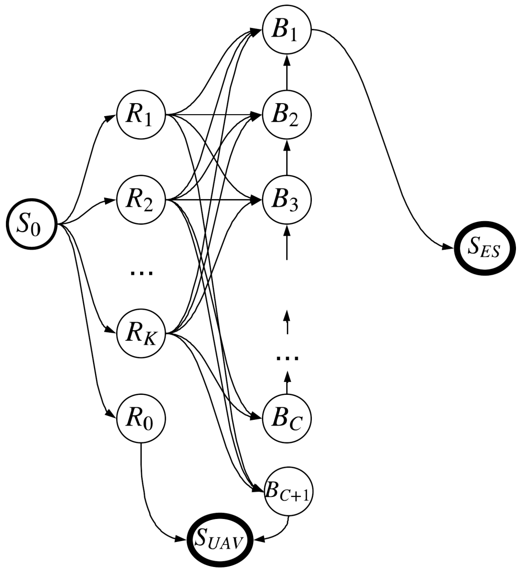


Figure 2: Representation of state transitions with non-zero probability in the Markov Chains associated with decision $u = 0$ (left) and $u = 1$ (right).

σ_c is the probability that the UAV task will find c tasks stored in the edge server buffer at arrival. It is known that

$$\sigma_c{=}\frac{(1-\lambda/\mu)(\lambda/\mu)^c}{1-(\lambda/\mu)^{C+1}}. \tag{9}$$

The state R_0, corresponding to disconnection from the network, deterministically leads to s_{UAV}.

At the beginning of any state B_c, the UAV is notified of the index c. For states B_c, $c{=}2,\dots,C$, the transition probabilities are

$$P(s'|B_c,1){=}\begin{cases}1 & \text{if } s'{=}B_{c-1}\\ 0 & \text{otherwise.}\end{cases} \tag{10}$$

State B_{C+1} corresponds to a full task queue and, thus, rejection of the UAV task. Therefore, from B_{C+1} the system deterministically moves to s_{UAV}. State B_1 corresponds to the UAV task being in the first position, and deterministically leads to s_{ES}.

B. Cost Functions and Optimal Policy

With the transition probabilities conditioned on the state and action, we can now build the optimization process. We consider a formulation where the objective of the UAV is to minimize $E(V)$, with

$$V = \omega E + (1{-}\omega)T, \tag{11}$$

where ω is a positive weight in $[0, 1]$.

To this aim, define the time and energy spent in state $s{\in}\mathcal{S}$ as $\Phi(s, u)$ and $\Psi(s, u)$ conditioned on the action u, respectively. Note that both the latter and the former are random variables. We denote their average as $\phi(s, u){=}E(\Phi(s, u))$ and $\psi(s, u){=}E(\Psi(s, u))$. We further define $C(s, u){=}\omega\Phi(s, u) + (1{-}\omega)\Psi(s, u)$, with average $c(s, u)$.

The average time and energy cost associated with the initial state are equal to 0. In the termination states s_{UAV} and s_{ES}, we have

$$\phi(s_{\mathrm{UAV}}) = 1/\mu', \tag{12}$$
$$\psi(s_{\mathrm{UAV}}) = (P_{\mathrm{P}} + P_{\mathrm{H}})/\mu', \tag{13}$$

and

$$\phi(s_{\mathrm{ES}}) = 1/\mu, \tag{14}$$
$$\psi(s_{\mathrm{ES}}) = P_{\mathrm{H}}/\mu. \tag{15}$$

Note that in the termination states the action is pre-determined and does not need to be formally included in the cost. From any transmission and queueing state $R_0, \dots, R_K$ and $B_1, \dots, B_{C+1}$, if the decision is to initiate local processing at the UAV ($u{=}0$), the process immediately moves to s_{UAV} and the energy and time cost are both equal to 0. Note that such decision is forced in states R_0 and B_{C+1}.

If the decision is to defer local processing ($u{=}1$), the costs are

$$\phi(R_i, 1) = L/R_i, \tag{16}$$
$$\psi(R_i, 1) = (P_{\mathrm{H}} + P)L/R_i. \tag{17}$$

with $i{=}, 1, \dots, K$, and

$$\phi(B_i, 1) = 1/\mu, \tag{18}$$
$$\psi(B_i, 1) = P_{\mathrm{H}}/\mu. \tag{19}$$

The problem of minimizing the expected total cost can be rephrased as a Markov Decision Process over a finite temporal horizon. We aim at finding, then, the (deterministic) optimal policy $u^*(s)$, where

$$u^*(s) = \arg\min_{u=\{0,1\}} E\big(V_{\mathrm{res}}(s, u)\big), \tag{20}$$

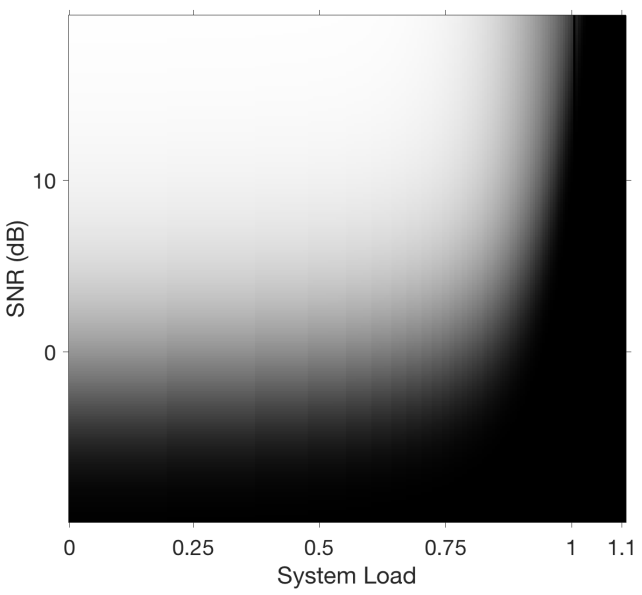


Figure 3: Probability of offloading to the edge server (lighter shades corresponds to higher probability) with $\omega = 0$

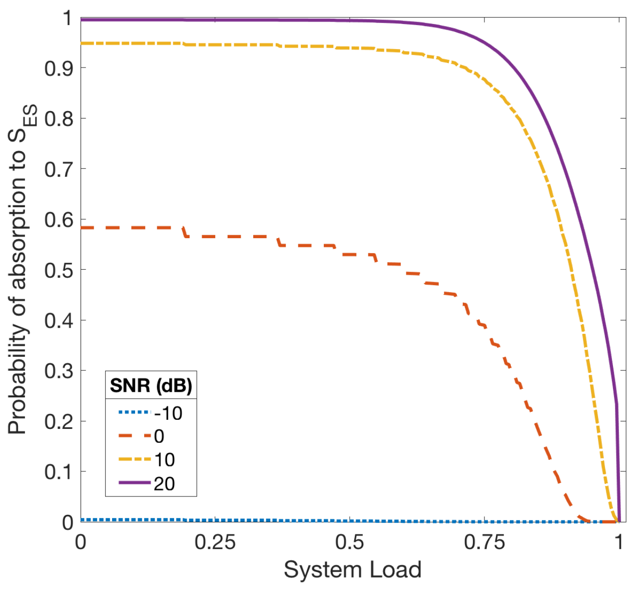


Figure 4: Probability of offloading the computation to the edge server as a function of the system load ρ.

where $E(V_{\text{res}}(s,u))$ is the expected minimum cumulative residual cost to a termination state $s^\dagger$ from state s given that decision u is selected, that is,

$$\min_{\mathbf{U_1}^{j^\dagger}} E\left(\sum_{j=0}^{j^\dagger} c(S(j),U(j)|U(0){=}u,S(0){=}s)\right), \tag{21}$$

where

$$j^\dagger = \min(j : S(j){\in}\{s_{\text{UAV}}, s_{\text{ES}}\}), \tag{22}$$

and $\mathbf{U}_1^{j^\dagger} = (U(0),\ldots,U(j^\dagger))$. The optimal policy can be recursively found using well-known techniques.

V. Numerical Results

To make our observations meaningful, we derive the optimal policies under different channel and load conditions using parameters obtained from real-word experimentations. Specifically, we used a 3DR Solo Drone mounting a Pixhawk flight controller running ArduCopter connected to a Raspberry Pi model 3B as companion computer. We use as edge server a Laptop with 16GB RAM and Intel Core i7-6700HQ processor with Nvidia GM204M GPU. We set the number of pictures collected in each location to 20, where each picture has resolution equal to 720×480. The average size of each picture after encoding is 80 KB. The pictures are processed implementing a face recognition algorithm using a multi scale Haar Cascade, which takes on average $1/\mu'$=0.56s at the UAV and $1/\mu$=0.046s at the edge server. We consider SNR values in the range $[-10, 20]$ dB and transmission rates in the range from 1 Mbps to 11 Mbps (matching a system using WiFi IEEE 802.11). Power consumption rates are based on battery level readings in the same set up: in particular, we set $P_h = 0.1$ levels/s, $P_p = 10\% \cdot P_h$ levels/s. The optimal deterministic policy $\mathbf{U}_1^{j^\dagger}$ given the parameters is computed using Equation (21).

In Figure 3, we show the probability of offloading to the Edge Server as a function of the SNR, and the system load $\rho = \lambda/\mu$. This probability corresponds to the probability of the process being absorbed in S_{ES} from S_0 conditioned on the control policy, defined as

$$P^\infty_{S_0}(Y) = \lim_{t\to\infty} P(S(t) = S_Y|S(0) = S_0, U = \mathbf{U}_1^{j^\dagger}) \tag{23}$$

where $Y \in \{S_{\text{UAV}}, S_{\text{ES}}\}$, and $P^\infty_{S_0}(S_{\text{UAV}}) + P^\infty_{S_0}(S_{\text{ES}}){=}1$. In Figure 3, we plot $P^\infty_{S_0}(S_{ES})$, using lighter pixel color for higher probabilities. As expected, for low values of ρ and high values of the SNR, the offloading probability is almost equal to 1, that is, the UAV offloads computation when system conditions are favorable. When the SNR is sufficiently low, the UAV will likely be disconnected, or the cost of offloading might exceed that of local computation due to the large time needed to transport the data to the edge server. Similarly, if the load parameter ρ is large, that is, the ES buffer has frequent arrivals or computation tasks take a large time to be completed, the UAV choses to compute locally.

In Figure 4 we fix the SNR, and show $P^\infty_{S_0}(S_{ES})$ as a function of ρ. As the SNR decreases, the probability of offloading to the edge server decreases as well. Intuitively, the SNR influences the shape of the probability curve. Interestingly, high SNR values show a sharp transition from offloading to local computing, whereas low SNR values have a more progressive transition, most likely due to the distribution of the communication time.

Figure 5 compares the total delay incurred by fixed strategies – local computing or edge offloading – with the proposed adaptive approach as a function of the system load ρ. Clearly, local computing at the UAV is not influenced by ρ. For small system loads, offloading grants a reduced delay. As ρ increases, the total delay associated with offloading increases to exceed that of local computation. The adaptive strategy

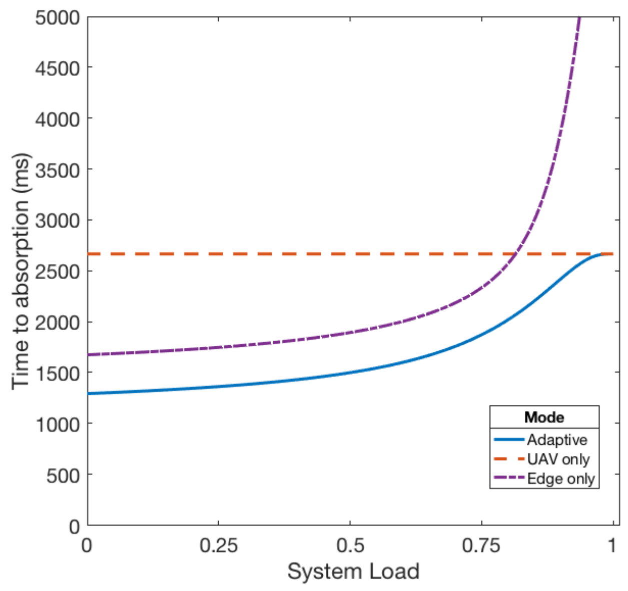


Figure 5: Total delay as a function of the system load ρ for the fixed strategies UAV-only and edge server-only, and the proposed adaptive strategy.

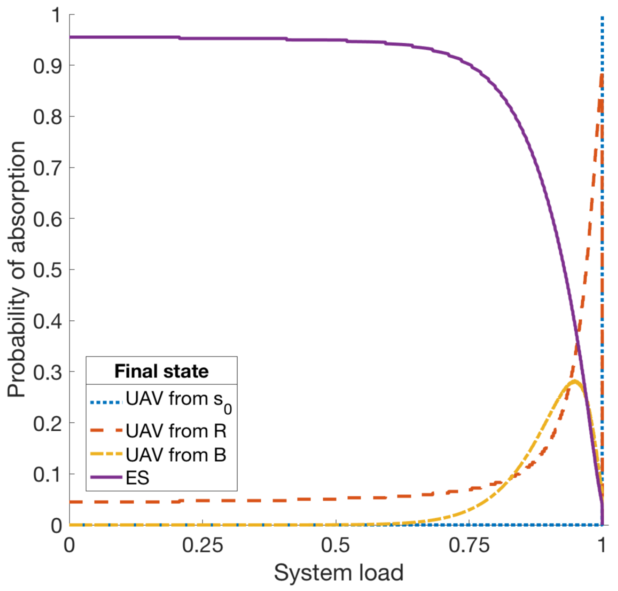


Figure 6: Probability of selecting local computation in the three main decision stages or offloading to the edge server.

opportunistically allocates the computation task to the UAV or the edge server, avoiding specific realizations of the system states associated with poor channel quality or congestion at the edge server. As a consequence, the adaptive strategy achieves a reduce delay except that in parameter regions where the choice is deterministic, where the adaptive strategy falls back to a fixed decision (corresponding to ρ approaching 1).

Finally we illustrate the value of probing compared to a simple decision informed by the average delay pre-computed based on a priori knowledge of the parameters. Figure 6 shows the probability that the decision of processing locally is taken at the different stages or that offloading is selected. Specifically, the decision stages are:

- **Stage 0:** the initial stage S_0, where the UAV knows the parameters, but not the channel or queue state;
- **Stage 1:** R_i, where the UAV has connected with the network and is aware of the maximum transmission rate;
- **Stage 2:** B_j, where the UAV reached the edge server, that is, upon transmission after transmission, and is reported the position in the processing queue.

For small values of ρ, offloading is predominant, with a small probability of local computing decision forced by extremely poor channel conditions. As ρ increases, the set of rates corresponding to local computing decisions increases. In fact, the delay requirement for the data transportation becomes more stringent as the average time spent in the edge server buffer increases. In the transition phase between offloading and local computation, we can observe a spike in the probability that the UAV will select local computing after the edge server is reached, as the probability finding a number of tasks in the buffer sufficiently large to make offloading disadvantageous increases before a region in which probing is not even attempted.

Results not shown here due to the lack of space show similar trends for cost functions accounting for the energy consumption of the UAV.

VI. Conclusions

In this paper, we presented a framework to control processing task offloading to edge servers in UAV systems. The framework sequentially probes the state of the network and of the edge server buffer to make optimal decisions between local and edge-assisted computing. Numerical results show the delay reduction granted by the proposed technique, which is based on a Markov Decision Process formulation solving an optimal stopping time problem, compared to non-adaptive strategies.

References

[1] Kapseong Ro, Jun-Seok Oh, and Liang Dong. Lessons learned: Application of small uav for urban highway traffic monitoring. In *45th AIAA aerospace sciences meeting and exhibit*, pages 2007–596, 2007.

[2] Janosch Nikolic, Michael Burri, Joern Rehder, Stefan Leutenegger, Christoph Huerzeler, and Roland Siegwart. A UAV system for inspection of industrial facilities. In *IEEE Aerospace Conference*, pages 1–8, 2013.

[3] Chunhua Zhang and John M Kovacs. The application of small unmanned aerial systems for precision agriculture: a review. *Precision agriculture*, 13(6):693–712, 2012.

[4] Milan Erdelj, Enrico Natalizio, Kaushik R Chowdhury, and Ian F Akyildiz. Help from the sky: Leveraging uavs for disaster management. *IEEE Pervasive Computing*, 16(1):24–32, 2017.

[5] Shams ur Rahman, Geon-Hwan Kim, You-Ze Cho, and Ajmal Khan. Deployment of an SDN-based UAV network: Controller placement and tradeoff between control overhead and delay. In *International Conference on Information and Communication Technology Convergence (ICTC)*, pages 1290–1292. IEEE, 2017.

[6] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. Fog computing and its role in the internet of things. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pages 13–16. ACM, 2012.