Channels, Learning, Queueing and Remote Estimation Systems With A Utilization-Dependent Component

Varun Jog, Richard J. La and Nuno C. Martins

Abstract—In this article, we survey the main models, techniques, concepts, and results centered on the design and performance evaluation of engineered systems that rely on a utilization-dependent component (UDC) whose operation may depend on its usage history or assigned workload. More specifically, we report on research themes concentrating on the characterization of the capacity of channels and the design with performance guarantees of learning algorithms, queueing and remote estimation systems. Causes for the dependency of a UDC on past utilization include the use of replenishable energy sources to power the transmission of information among the sub-components of a networked system, the influence of the dynamics of optimization iterates on the convergence of learning mechanisms and the assistance of a human operator for servicing a queue. Our analysis unveils the similarity of the UDC models typically adopted in each of the research themes, and it reveals the differences in the objectives and technical approaches employed. We also identify new challenges and future research directions inspired by the cross-pollination among the central concepts, techniques and problem formulations of the research themes discussed.

Index Terms—Channel capacity, learning algorithms, task scheduling, queueing, remote estimation, energy harvesting, human factors, age of information, security.

I. INTRODUCTION

As new technologies and applications emerge, the algorithms that determine the functionality and regulate the operation of engineered systems have to contend with unexperienced constraints and nonstandard problems. This evolution has been evident for communication [1], cyber-physical [2], [3], human-assisted [4], networked estimation [5] and control [6] systems, which are now designed for maximal performance subject to restrictions that are more intricate than the conventional limits on reliability and power usage. In this article, we provide a partial account of such advances by surveying models, concepts and results on the characterization of channel capacity, and the design and performance analysis of learning algorithms, queueing and remote estimation systems, all of which have in common the unconventional attribute of relying on a component whose performance may be constrained by its usage history and possibly also be affected by the workload assigned to it. We refer succinctly to this class of components as UDC, which stands for utilization-dependent component. We do not aim at a comprehensive survey; instead, we will cite a selection of published work relevant to each key concept, problem formulation or technique on an as-needed basis for illustration.

A primary goal of this article is to foster future research that builds on the cross-pollination among the methods and problem formulations originally developed and employed on each of the research themes broached.

Paper structure: After the Introduction, in Section II we define a class of UDC that is general enough to model the performance restrictions imposed by the reliance on the energy harvested from stochastic sources, human-assisted decision-making, or human labor. In Section III we introduce widely-used models quantifying certain performance-limiting factors, such as mental workload, queueing workload and the state of charge of the battery of an energy harvesting module. Subsequently, in Sections IV-VII we employ these definitions as a unifying framework to discuss research on methods to design and analyze the performance of systems comprising a UDC in the context of communication, learning algorithms, queueing and remote estimation, respectively. This article ends with the conclusions and future directions proposed in Section VIII.

II. A GENERAL UTILIZATION-DEPENDENT COMPONENT (UDC) MODEL

We start by proposing a model that is general enough to describe all types of UDC considered throughout this article. Without loss of generality, we limit our discussion to discrete-time processes and models. Namely, time takes values in the set of non-negative integers \(\mathbb{N}\), and we use \(\mathbb{N}_+\) to indicate the subset that excludes 0.

Definition 1. (UDC Model) The following describes the two main sub-components of the UDC model (see Fig. I).

- The first sub-component is a partially-observed controlled Markov chain (POCMC), whose state is represented as \(S := \{S(k) : k \in \mathbb{N}\}\). It has two inputs denoted as \(Y := \{Y(k) : k \in \mathbb{N}\}\) and \(U := \{U(k) : k \in \mathbb{N}\}\), where the latter is an external control process. The outputs are indicated as \(O := \{O(k) : k \in \mathbb{N}\}\) and \(W := \{W(k) : k \in \mathbb{N}\}\). The former is characterized by an output kernel and is available to the policy that generates \(U\) and, in some cases, also \(X\), while the latter is a deterministic function of \(S\) that we refer to as the performance process. The processes \(U, Z, S\)
and $W$ take values in given alphabets $\mathbb{U}$, $\mathbb{Z}$, $\mathbb{S}$ and $\mathbb{W}$, respectively, which are subsets of real coordinate spaces. The POCMC is specified by maps $S : \mathbb{S}^2 \times \mathbb{U} \times \mathbb{Y} \rightarrow [0, 1]$, $O : \mathbb{O} \times \mathbb{S} \times \mathbb{U} \rightarrow [0, 1]$ and $W : \mathbb{U} \times \mathbb{S} \rightarrow \mathbb{W}$. The first two determine the state transition probability and the output kernel as follows:

$$S(s^+|s, u, y) := \mathbb{P}(S(k+1)|S(k), U(k), Y(k))(s^+|s, u, y),$$

$$O(v|s, u) := \mathbb{P}(O(k)|S(k), U(k))(v|s, u),$$

The performance process is determined as $W : (U(k), S(k)) \rightarrow W(k)$.

- The second sub-component is an action kernel that models the functionality whose performance is affected by the process $W$. More specifically, the output of the action kernel is a state and the inputs are $X = \{X(k) : k \in \mathbb{N}\}$ and an external source or command signal denoted as $Y := \{Y(k) : k \in \mathbb{N}\}$. The processes $X$ and $Y$ take values in given alphabets $\mathbb{X}$ and $\mathbb{Y}$, respectively, which are subsets of real coordinate spaces. A map $A : \mathbb{X} \times \mathbb{S} \rightarrow [0, 1]$ specifies probabilistically $Y$ in terms of $X$ and $S$ as follows:

$$A(y|x, s) := \mathbb{P}(Y(k)|X(k), S(k))(y|x, s),$$

$$y \in \mathbb{Y}, \ x \in \mathbb{X}, \ s \in \mathbb{S}$$

The definition of the UDC model is not complete until we specify the probabilistic dependence among the implied sources of randomness of the state recursion and the output kernels. These will be particularized throughout the text on an as-needed basis. Typically, $S(k+1)$ and $O(k)$ are conditionally independent given $S(k)$, $U(k)$ and $Y(k)$; and $Y(k)$ and $O(k)$ are conditionally independent given $S(k)$, $U(k)$ and $X(k)$.

Although most existing work adopts variations of the models discussed in Section III, we opted to define a UDC model that is general enough to be used as a common framework for future work.

### III. Commonly Used POCMC Models

We proceed with defining a few common POCMC models, which we will invoke later appropriately altered to suit a specific application. In Sections III-A and III-B we will specify $W$ as a function of $U$ and we will explicitly describe a state recursion in cases when it is clearer to do so. Because the output kernel is application dependent, we will defer its specification on an as-needed basis to Sections IV-VII. Typical cases include when $O$ equals $S$ or when additive measurement noise is present.

To be concise, when the POCMC is deterministic, we specify it via the functional recursion that governs the state update in terms of the inputs, and we express each output as a function of the current state and inputs. In the stochastic case, the probabilistic state recursion and output kernel can always be specified by the conditional probabilities associated with $S$ and $O$, respectively. The action kernel is specified in an analogous manner.

In order to appropriately indicate the dependence on certain parameters, or to discern which model is associated with a given internal process, such as $S$ and $W$, we often annotate them with a self-descriptive superscript.

#### A. Utilization Ratio and Workload Models

In human-assisted systems, the concept of mental workload [4] refers broadly to the burden imposed on a human operator by the difficulty of and frequency with which tasks are assigned. Hence, considering that it is known to influence the performance of a human operator [7], quantifying workload is important for the design of task assignment policies. In spite of having a rather simple structure, [8] Chapter 11 explains why the utilization ratio defined below is a pertinent mental workload metric. Here, the UDC is a human operator who has to service tasks from a queue. The types of services carried out by an operator include classification, supervision [9], [10] or assembly jobs within a production system [11].

**Definition 2. Utilization ratio:** The utilization ratio POCMC for a given positive averaging horizon $T$ is defined as:

$$W^{xT}(k) = \frac{1}{T} \min_{i=1}^{T(k)} U(k - i), \ k \in \mathbb{N}_+$$

where we adopt the convention that $\mathbb{W}^{xT} = \mathbb{R}_+$, $W^{xT}(0) = 0$ and $U$ is a scalar non-negative process that governs the level of utilization.

In its simplest and most prevalent form, the set of utilization levels $U$ would be $\{0, 1\}$, and $U(k) = 1$ and $U(k) = 0$ would indicate whether the component is being used or not, respectively, at time $k$. The research on intelligent task management reported in [12], also uses $U = \{0, 1\}$ for the following alternative mental workload metric quantifying utilization ratio with a forgetting factor.

**Definition 3. Utilization ratio with forgetting factor:** Given a forgetting factor $\alpha$ in $(0, 1)$, the associated utilization ratio POCMC is defined as:

$$S^{x, \alpha}(k+1) = \alpha S^{x, \alpha}(k) + U(k), \ k \in \mathbb{N}, \ S^{x, \alpha}(0) = 0$$

$$W^{x, \alpha}(k) = (1 - \alpha) S^{x, \alpha}(k), \ k \in \mathbb{N}$$

where we adopt the convention that $\mathbb{W}^{x, \alpha}$ and $S^{x, \alpha}$ are $\mathbb{R}_+$ and $U$ is a scalar non-negative process that governs the
level of utilization. Notice that the performance process can be computed directly for \( k \) greater than or equal to 1 as
\[
W^\tau,k(k) = (1 - \alpha) \sum_{i=0}^{k-1} \alpha^{k-1-i} U(i).
\]

The authors of [13] also adopt the utilization ratio with forgetting factor to model the dynamics of the temperature of the circuitry of the transmitter that broadcasts information across an additive noise link, subject to a transmission power process \( U \). In this context, \( U \) is \( \mathbb{R}_+ \) and the UDC is the resulting communication channel whose performance is adversely affected by the thermal noise that intensifies with increasing temperature. Related work on allocation of energy harvested from a stochastic source for wireless transmission subject to constraints on temperature is reported in [14], while thermal effects were considered in [15] in the context of distributed estimation.

In contrast to the concept of mental workload, in the queueing literature the workload affecting the performance of the server quantifies the effort needed to complete the tasks apportioned to the server, but not yet completed. The following is a discrete-time approximation of the continuous-time model governing the workload process analyzed in [16].

**Definition 4. Queueing Workload:** The following defines the queueing workload POCMC for a component acting as a server:
\[
S^w(k + 1) = S^w(k) + U(k) - \tilde{Y}(k), \quad k \in \mathbb{N}, \quad S^w(0) = 0
\]
\[
W^w(k) = S^w(k)
\]
where \( S^w, U \) and \( \tilde{Y} \) are \( \mathbb{N} \). Here, \( U(k) \) and \( \tilde{Y}(k) \) may represent the number of work quanta, or effort, associated with the incoming and completed tasks at time \( k \), respectively. Notice that, given the structure in Fig. 4, \( \tilde{Y}(k) \) must be determined as a function of \( Y(k) \). We assume that \( \tilde{Y}(k) \) must be zero when \( S^w(k) \) and \( U(k) \) are zero, which requires a properly defined action kernel.

**B. Energy harvesting models**

We also consider cases in which the state of the POCMC is governed not only by utilization but, unlike the models covered in Section 11-A, also by extrinsic stochastic processes. Prime examples of these, include models of the so-called state of charge (SOC) quantifying the energy stored in a battery that is repeatedly recharged using energy harvested from unsteady sources. We propose the following model that is both a generalization and an adaptation to our discrete-time framework of ubiquitous models, such as those used in [17, 18]. Our model is general enough to capture the effects described in [19, Part IV] for microbatteries that are often used in small devices powered by energy harvesting, including implanted medical devices [20].

**Definition 5. Energy harvesting (EH) model**

Let \( S^w := \{ S^w(k) : k \in \mathbb{N} \} \) be a given homogeneous Markovian process that quantifies not only the energy harvested over time but possibly also other stochastic phenomena that influence the operation of the battery and its recharging subsystems. This process takes values in a subset \( \mathbb{S}^w \) of a real coordinate space. A given map \( \Delta^\tau : \mathbb{S}^{soc} \times \mathbb{S}^w \times \mathbb{R}_+ \rightarrow \mathbb{R}_+ \) governs the dynamics of the energy harvesting POCMC according to the following recursion:
\[
S^{soc}(k+1) = S^{soc}(k) + \Delta^\tau(S^{soc}(k), S^w(k), W^\tau(k)), \quad k \in \mathbb{N}
\]

where we assume that \( S^{soc}(k) \), which quantifies the SOC at time \( k \), is in \( \mathbb{S}^{soc} := [0, s^{soc}] \) and \( s^{soc} \) denotes the maximum SOC. The initial SOC is quantified by \( S^{soc}(0) \). Here, \( W^\tau \) takes values in \( \mathbb{R}_+ \) and quantifies the energy effectively extracted from the battery for use by the action kernel. The map \( \Delta^\tau \) quantifies the net change in the battery charge resulting from the difference between the effect of \( W^\tau(k) \) and the energy harvested. The state of the POCMC can be chosen as \( S^w := \{ (S^{soc}(k), S^w(k)) : k \in \mathbb{N} \} \). The map \( W^\tau \) that determines \( W^\tau \) in terms of \( S^w \) and \( U \) is described below in Remark 7.

The map \( \Delta^\tau \) is characteristic of each battery and it must satisfy the following consistency conditions:
\[
\Delta^\tau(s^{soc}, s^w, w^\tau) \leq s^{soc} - 0, \quad s^{soc} \geq 0,
\]
\[
\Delta^\tau(s^{soc}, s^w, w^\tau) \geq -s^{soc}, \quad s^{soc} \geq 0,
\]
for all \( s^{soc}, s^w \) and \( w^\tau \) in \( \mathbb{S}^{soc}, \mathbb{S}^w \) and \( \mathbb{R}_+ \), respectively.

**Remark 1. (Description of \( W^\tau \))**

As is known since the early work in [21], each battery type has a discharge curve that characterizes the voltage in terms of the SOC. Invariably, even for modern batteries [22], the voltage decreases as the SOC drops, which leads to the following constraints:

- The maximum energy that can be delivered by the battery at any time \( k \) is a decreasing function of the SOC, which we represent as \( D : \mathbb{S}^{soc} \rightarrow \mathbb{R}_+ \). More specifically, the constraint is given by:
\[
W^\tau(k) \leq D(S^{soc}(k)), \quad k \in \mathbb{N}
\]

- There is a positive minimum SOC, denoted as \( s^{soc} \), below which the voltage is too low to power the component. This leads to the following constraint that must be satisfied for every \( k \) in \( \mathbb{N} \):
\[
\left( \Delta^\tau(S^{soc}(k), S^w(k), W^\tau(k)) + S^{soc}(k) - s^{soc} \right) W^\tau(k) \geq 0
\]

Consequently, \( U \), which represents the energy requested by the control policy, may differ from \( W^\tau \). More specifically, the energy extracted from the battery is determined in terms of \( U \) and the state of the EH model via the map \( W^\tau : (s^w, u) \mapsto w^\tau \) specified as follows:

\[
w^\tau = \max \left\{ \tilde{w}^\tau \geq 0 \mid \tilde{w}^\tau \leq u, \tilde{w}^\tau \leq D(s^{soc}) \right\}
\]

The following is a simplified version of the EH model that is characterized by \( \Delta^\tau \) with a linear range in which it quantifies...
the difference between the energy used and harvested, and it also implements a saturation that restricts the SOC to the interval $S_{SOC}^\circ$.

**Definition 6. (Linear-saturated EH model)**

The linear-saturated EH model is specified as follows for every $k$ in $\mathbb{N}$:

$$S_{SOC}^\circ(k+1) = \min \{S_{SOC}^\circ(k) + S^i(k) - W^i(k), S_{SOC}^\circ \}$$ (10a)

$$W^i(k) = \begin{cases} U(k) & \text{if } U(k) \leq S_{SOC}^\circ(k) + S^i(k) \\ S_{SOC}^\circ(k) + S^i(k) & \text{otherwise} \end{cases}$$ (10b)

where $S^i(k)$, in this simplified model, represents the energy harvested at time $k$.

The linear-saturated EH model does not capture the effects of the discharge curve and the changes that the mechanisms of charge and discharge go through as the SOC varies. The following is another simplified model in which the SOC takes values in a finite set and evolves as a controlled Markov chain.

**Definition 7. (Finite-state EH model)**

The SOC evolves according to a controlled Markov chain whose state $S_k^e := \{S^e(k) : k \in \mathbb{N} \}$ takes values in $S^e := \{0, 1, \ldots, \bar{s} \}$. The following map determines the probability transition map for $S^e$ in terms of $U$:

$$S^e(s^+|s, u, y) = \begin{cases} \Gamma^e_s(u) & \text{if } s^+ = s + 1, s < \bar{s} \\ \Gamma^e_s(u) & \text{if } s^+ = s - 1, s > 0 \\ 1 - \Gamma^e_s(u) - \Gamma^e_u(u) & \text{if } s^+ = s \end{cases}$$

where $\Gamma^e_s : U \rightarrow [0, 1]$ and $\Gamma^e_u : U \rightarrow [0, 1]$ are given maps satisfying $\Gamma^e_0(u) = 0$, $\Gamma^e_s(u) = 0$, and $\Gamma^e_s(u) + \Gamma^e_u(u) \leq 1$.

Here, we assume that $S^e(k+1)$ is independent of $Y(k)$ when conditioned on $S^e(k)$ and $U(k)$ taking values $s$ and $u$, respectively. In this case, a given map $W^e : S^e \times U \rightarrow \mathbb{R}_+$ determines the energy used by the action kernel as $W^e : (S^e(k), U(k)) \mapsto W^e(k)$.

### IV. Capacity of Communication Channels

In recent years, there has been a tremendous amount of research focused on energy harvesting wireless communication systems. For a comprehensive survey, we refer the reader to survey articles [23], [24], [25]. As detailed above, an energy harvesting transmitter may be modeled using the UDC model with a state that indicates the amount of charge available for usage. Therefore, we may use a UDC-based energy harvesting model to study various communication problems with energy harvesting devices. In what follows, we provide a brief survey of two key areas: determining channel capacities and optimal scheduling policies in energy harvesting systems.

From an information-theoretic perspective, a key problem is identifying the capacity of an energy harvesting communication channel. The capacity of an additive white Gaussian noise (AWGN) channel with an energy harvesting transmitter was analyzed in references [26] and [27], for the infinite battery case and the no-battery case, respectively. Various upper and lower bounds on capacities have been studied in [28], [29], [30], [31], [32]. A general formula for the capacity of a point-to-point energy harvesting channel was established in [33]. Reference [35] also established a novel connection between the channel capacity and the optimal throughput (discussed below) for an energy harvesting transmitter. Beyond point-to-point channels, the capacity of energy harvesting MAC channels has been analyzed in [34], [35], where a general capacity formula is derived, along with lower and upper bounds on capacity.

A significant amount of research has focused on the problem of scheduling for an energy harvesting transmitter. In this case, the transmitter has an energy queue as well as a data queue, and the goal is to transmit data to the recipients in the least amount of time, or equivalently transmit the maximum amount of data until a certain time. This problem has been studied in the offline setting, where the energy arrivals are non-causally known, as well as the online setting where the transmitter has causal information about energy and data arrival [24], [25]. For the offline case, a variety of channel models have been investigated including point-to-point channels, broadcast channels [36], [37], interference channels [38], and MAC channels [39]. The online case has also been studied for the point-to-point channel [40], the broadcast channel [41], and the MAC channel [42]. We refer to [42] for a thorough list of references concerning online and offline scheduling in energy harvesting channels. In all the models described so far, the transmitter utilizes energy for the sole purpose of transmission. Energy harvesting transmitters which expend energy on sensing, computing, communicating, and possessing imperfect batteries have been surveyed in [25].

### A. Channels with evolving power constraints

In addition to energy harvesting systems, we show that the UDC framework may also be used to analyze more general communication channels with time evolving power constraints. We describe these constraints below. The AWGN channel is one of the most popular channel models in information theory due to its relevance in practical applications. Evaluating the capacity of the AWGN channel under a variety of power constraints is a problem that has received much attention in the literature. The classical constraint studied by Shannon [43] involved an average power constraint of $P_{avg}$. In other words, if $(X(1), X(2), \ldots, X(n))$ is the input to a channel, then it must satisfy

$$\frac{1}{n} \sum_{k=1}^{n} ||X(k)||^2 \leq P_{avg}.$$  

Shannon showed that the capacity of this channel is achieved using a random Gaussian codebook. In addition to the average power constraint, another practically relevant power constraint is the peak power constraint. A peak power constraint of $P_{peak}$ stipulates that every input $X$ to the channel should satisfy $||X||^2 \leq P_{peak}$. Finding the capacity of this channel in the scalar case was first studied by Smith [44]. Smith showed that, although it is not possible to express the capacity in a closed-form expression, it may be calculated efficiently. The
The key observation in [44] was that the capacity is achieved by a discrete input distribution that is supported on a finite number of atoms in $[-\sqrt{P_{\text{peak}}}, \sqrt{P_{\text{peak}}}]$.

The flexible UDC framework allows us to model a variety of power-constrained channels. For example, the average power and peak power constraint may be restated as

$$\|X(k)\|^2 \leq \min \left\{ kP_{\text{avg}} - \sum_{j=1}^{k-1} \|X(j)\|^2, P_{\text{peak}} \right\}, \quad k \in \mathbb{N}_+.$$  

It is evident that the power constraint on the $k$-th channel use depends not only on $P_{\text{avg}}$ and $P_{\text{peak}}$ but also on the symbols transmitted prior to time $k$. Therefore, this power constraint is utilization dependent. We now provide a description of the UDC framework used for modeling a large class of power-constrained communication channels.

**Definition 8** (Evolving power constraints). An evolving power constraint is defined via a sequence of functions $\{P_k : k \in \mathbb{N}\}$, where $P_k : \mathbb{R}_+^k \rightarrow \mathbb{R}_+$ such that $P_k(u_1, u_2, \ldots, u_k)$ determines the power constraint on the $(k+1)$-th transmission $X(k+1)$, where $u_i$ is the power of the $i$-th transmitted symbol, i.e., $u_i = ||X(i)||^2$, $i \geq 1$.

**Definition 9** (Evolving power constraint with state). An evolving power constraint with a state is characterized by three sequences of functions: (i) $\{f_k : k \in \mathbb{N}\}$ with $f_k : \mathbb{R}^k \rightarrow \mathbb{R}$, (ii) $\{p_k : k \in \mathbb{N}\}$, where $f_k : \mathbb{R}^2 \rightarrow \mathbb{R}$, and (iii) $\{pk : k \in \mathbb{N}\}$ such that $p_k : \mathbb{R} \rightarrow \mathbb{R}$. These functions satisfy the property that $f_k(u_1, \ldots, u_k) = f_k(p_{k-1}(u_1, \ldots, u_{k-1}), u_k)$, i.e., the value of function $f_k$ at time $k$ can be computed from that of function $f_{k-1}$ at time $k-1$ and $u_k$.

An evolving power constraint is said to have a state $f_k(u_1, \ldots, u_k)$ at time $k+1$ if the sequence of functions $P_k$ in Definition 8 may be written as $P_k(u_1, \ldots, u_k) = p_k(f_k(u_1, \ldots, u_k))$. Thus, the power constraint on the $k$-th symbol depends on the history of transmitted symbols up to time $k-1$ through the state at time $k$.

Evolving power constraints may be used to describe several power constraints studied in the literature. We provide a few examples below:

**Example 1.** Consider the standard average power constrained communication channel. Here, the constraint on the $k$-th symbol $X(k)$ is given by

$$\|X(k)\|^2 \leq kP_{\text{avg}} - \sum_{j=1}^{k-1} \|X(j)\|^2,$$

for some fixed $P_{\text{avg}} > 0$. This power constraint may be characterized as an evolving power constraint with state, as follows: For $k \in \mathbb{N}$, define $f_k(x_1, \ldots, x_k) = \sum_{j=1}^{k} \|x_j\|^2$. This definition satisfies the property that $f_{k+1}(x_1, \ldots, x_{k+1})$ can be calculated using $f_k(x_1, \ldots, x_k)$ and $x_{k+1}$; in particular, $f_{k+1}(u, v) = u + v$. The power constraint functions $P_k$ as $P_k(x_1, \ldots, x_k) = (k+1)P_{\text{avg}} - f_k(x_1, \ldots, x_k)$ for every $k \in \mathbb{N}$.

**Example 2.** For an average power constraint of $P_{\text{avg}}$ coupled with a peak power constraint of $P_{\text{peak}}$, the only change from above is that $P_k(x_1, \ldots, x_k) = \min(P_{\text{peak}}, (k+1)P_{\text{avg}} - f_k(x_1, \ldots, x_k))$.

**Example 3.** For a windowed-average power constraint over a window $T$, the state is the total energy expended over the last $T-1$ transmitted symbols. By allowing states to be vector valued in $\mathbb{R}^{T-1}$, this constraint is easily accommodated in Definition 8.

**Example 4.** A $(\sigma, \rho)$-power constraint is found to be relevant in energy harvesting applications as well as neuroscience. The $(\sigma, \rho)$-power constraint is defined as follows: Let $\sigma, \rho \geq 0$. A codeword $(x_1, x_2, \ldots, x_n)$ is said to satisfy a $(\sigma, \rho)$-power constraint if

$$\sum_{j=k+1}^{n} x_j^2 \leq \sigma + (l-k)\rho \quad \text{for all} \quad 0 \leq k < l \leq n. \quad (11)$$

The $(\sigma, \rho)$-power constraint essentially imposes a restriction on how bursty the transmit power can be, by constraining the total energy consumed over every interval to be approximately linear in the length of the interval. In energy harvesting communication systems, a $(\sigma, \rho)$-power constraint may be used to model a transmitter that harvests $\rho$ units of energy per unit time, and is equipped with a battery with capacity $\sigma$ units that is used to store unused energy for future transmissions. The $(\sigma, \rho)$-power constraints can be expressed equivalently by tracking a state parameter $\sigma_k$, that keeps track of the tightest constraint among the $k+1$ inequalities for $x_{k+1}$. The state function $\sigma_k$ evolves as follows:

$$\sigma_k(x_1, \ldots, x_k) = \min(\sigma, \sigma_{k-1}(x_1, \ldots, x_{k-1}) + \rho - x_k^2).$$

The power constraint function $P_k$ is defined as $P_k(x_1, \ldots, x_k) = \sigma_k(x_1, \ldots, x_k) + \rho$. Note that $\rho$ need not be constant over time, and such dependence or variability with respect to time is useful in modeling energy harvesting with arbitrary amounts of energy $\rho_k$ harvested at time $k$.

We define a UDC model that imposes evolving power constraints on an action kernel that is a communication channel.

**Definition 10** (POCMC component). Let $\{(P_k, f_k, \hat{f}_k) : k \in \mathbb{N}\}$ be an evolving power constraint with state. The POCMC component has state $S(k)$ that tracks the state of the power constraint at time $k$, i.e. $S(k) = f_{k-1}(u_1, \ldots, u_{k-1})$. An input $U(k)$ indicates the desired power output for time $k$, i.e. the energy required to send symbol $X(k)$. The performance process $W(k)$ is equal to the power constraint imposed on $X(k)$. In other words, $W(k) = P_{k-1}(u_1, \ldots, u_{k-1}) = p_{k-1}(S(k))$ is a deterministic function of $S(k)$. The state at time $k+1$ satisfies $S(k+1) = \hat{f}_k(S(k), u_k)$.

**Definition 11** (Action kernel). The action kernel is a communication channel with input $X$ and output $Y$. At time $k$, the $k$-th symbol $X(k)$ is scheduled to be transmitted. The output of the channel $Y(k)$ depends on the input $X(k)$, the noise in the channel, as well as the performance process $W(k)$. Two natural cases to consider are:

1. If $U(k) \leq W(k)$, then $X(k)$ is transmitted unaltered. Otherwise, $X(k)$ is rescaled to have power $W(k)$, i.e.
Symbol in advance and ensure that the transmitter can calculate the power constraints on the $k$-th symbol in advance and ensure that $X(k)$ satisfies these power constraints. However, this is not possible when the power constraints are random. An example of random power constraint is the following. Consider an arbitrary i.i.d. stochastic process $E(k)$. We may now define the state as $(S(k), E(k))$ and the power constraint on the $k$-th symbol is computed via $p_k(S(k), E(k))$. The state evolution proceeds as $S(k + 1) = f_k(S(k), u_k, E(k))$. This particular formulation is relevant to energy harvesting communication systems, discussed in Section III-B. In the absence of any output $O(k)$, the transmitter has no way of modifying its $k$-th symbol to satisfy the power constraints. A variety of feedback settings are worth considering: $O(k) = S(k)$ or $O(k) = (S(k), E(k))$. Additionally, the transmitter may also receive feedback from the receiver, i.e., $O(k)$ contains $Y(k)$. The capacity of energy harvesting systems with feedback has also been investigated in recent years, and it has been found that feedback increases capacity [45]. In addition to random $E(k)$, yet another setting to consider is when the sequence of $E(k)$ is completely arbitrary, but is known to lie in some set. This is analogous to arbitrarily varying channels (AVCs) [46], [47] and may also be modeled using the UDC framework.

B. Utilization-dependent Markov channels

Point-to-point communication under memoryless channels is widely studied and well-understood. More difficult is modeling channels that change with time, such as fading channels, or channels that are simply unknown, e.g., AVCs [46], [47]. There are other channel models that lie between a memoryless channel and an AVC, such as a finite-state Markov channel (FSMC) [48], and the channel with action dependent states [49]. It is easy to model these examples under the UDC framework. The key advantage of the UDC framework is, however, that the channel can also depend on its usage, i.e., on past symbols transmitted through the channel.

Inter-symbol interference (ISI): ISI channels are a natural example of such channels. We demonstrate a UDC model that describes the discrete-time Gaussian ISI channel as found in [50]. For an input $X(k)$ at time $k$, the output $Y(k)$ of this channel is given by

$$Y(k) = \sum_{i=0}^{T-1} h(i)X(k - i) + N(k),$$

where $N(k)$ is AWGN. The controlled Markov chain has the state $S(k) := (X(k - 1), \ldots, X(k - T)) \in \mathbb{R}^{T-1}$. The performance process is identical to the state. The action kernel consists of an AWGN channel with input $X(k)$ and output $Y(k)$. Based on the performance process, the output is evaluated as per equation (12). The input to the controlled Markov process at time $k$ is $U(k) := X(k)$. The state at time $k + 1$ is given by $S(k + 1) = (X(k), \ldots, X(k - T + 1))$.

The main point to note above is that the channel depends on the past usage, and is therefore a utilization-dependent channel. Just as the FSMC, we provide an example here of a finite-state Markov channel that is also utilization-dependent.

Utilization-dependent FSMC: For finite discrete sets $\mathbb{X}$ and $\mathbb{Y}$, a channel with input $X \in \mathbb{X}$ and output $Y \in \mathbb{Y}$ is defined as the collection of probability measures $p_{Y|X=x}(\cdot)$ for each $x \in \mathbb{X}$. Consider a finite number of channels $\{C_1, \ldots, C_M\}$. For each $x \in \mathbb{X}$ and $i, j \in \{1, \ldots, M\}$, consider the function $\theta(x, i, j)$, and the random process $\{S^n(k) : k \geq 1\}$ over the set $\{1, 2, \ldots, M\}$ as follows:

$$\mathcal{P}(S^n(k + 1) = j|S^n(k) = i, X(k) = x) = \theta(x, i, j)$$

Naturally, we have $\theta(\cdot, \cdot, \cdot) \geq 0$ and $\sum_{j=1}^M \theta(i, j, x) = 1$ for all $i \in \{1, \ldots, M\}$ and all $x \in \mathbb{X}$. The random process $\{S^n(k) : k \geq 1\}$ dictates which channel is available at each time $k$ in the action kernel. The main point to note is that the index of the channel at time $k + 1$, which is $S^n(k + 1)$, depends not only on $S^n(k)$, but also on the symbol $X(k)$ transmitted at time $k$. A utilization-dependent FSMC is described in Fig. 2. The main components are described below.

Markov process and action kernel: The controlled Markov chain has input $X(k)$, and state $S^n(k)$. The performance process is identical to the state. The action kernel is the communication channel, and it also has input $X(k)$. The channel at time $k$ is $C_{S^n(k)}$, and a random output $Y(k)$ is generated by passing $X(k)$ through this channel. At time $k + 1$, the state $S^n(k + 1)$ equals $j \in \{1, 2, \ldots, M\}$ with probability $\theta(S^n(k), j, X(k))$. The simplest FSMC channel to consider is the Gilbert-Elliott channel [51], where the input and output are binary. The set of channels is also binary and consists of two binary symmetric channels. A utilization-dependent Gilbert-Elliott channel is completely specified in terms of a function $\theta : \{0, 1\}^3 \rightarrow [0, 1]$. There are a number of open problems concerning such channels: Is it possible to calculate channel capacity in closed-form? If the input $\{X(k) : k \in \mathbb{N}\}$ is fixed to be Markov, what is the maximum achievable channel capacity? Is it possible to generalize these results beyond binary channels?

V. LEARNING ALGORITHMS AND ADVERSARIAL MODELS

In recent years, information-theoretic techniques have emerged as effective tools to study optimization procedures in machine learning problems [52], [53], [54]. A common problem setting is as follows: A dataset $D$ is constructed by

![Fig. 2. A utility dependent finite-state Markov Channel (FSMC).](image-url)
drawing $N$ i.i.d. samples from a distribution $p_U$ over a set $\mathcal{U}_D$; i.e., $D = \{U_1, \ldots, U_N\}$ where $U_i \sim p_U$ and, hence, $D \sim p^{\otimes N}_U$. A loss function $\ell : \mathcal{W} \times \mathcal{U}_D \to \mathbb{R}$, where $\mathcal{W}$ is a set of parameters that govern a machine learning algorithm. The goal is to identify $w^* \in \mathcal{W}$ such that

$$w^* = \arg\min_{w \in \mathcal{W}} \mathbb{E}[\ell(w, U)],$$

where $U$ has distribution $p_U$. The above expression cannot be evaluated in general since the data distribution $p_U$ is unknown. A natural idea is to use the empirical distribution of $U$ as per dataset $D$ for estimating $w^*$. The optimization problem is formulated as

$$\min_{w \in \mathcal{W}} \frac{1}{N} \sum_{i=1}^{N} \ell(w, U_i).$$

In general, such a problem is nonconvex and intractable. However, it has been observed that a local minimum obtained via gradient descent or stochastic gradient descent (SGD) is often a good enough estimate [55]. Note that the input to an algorithm is $D$, and the output is some $W \in \mathcal{W}$. We may think of the algorithm as a communication channel that maps $D$ to $W$ via $p_W|_{\mathcal{D}(|\cdot|)}$. Recent results in learning theory have shown that the mutual information $I(D; W)$ provides upper bounds on the generalization error of an algorithm, which measures the degree to which an algorithm overfits to the data. Thus, designing an algorithm is equivalent to designing the communication channel $p_W|_{\mathcal{D}(|\cdot|)}$.

In the next subsection, we show how SGD and closely related optimization procedures may be formulated using the UDC framework. The advantage of formulating the problem in this framework is that we are able to describe several interesting problems of learning in the presence of adversaries.

### A. Iterative algorithms

Iterative optimization algorithms produce a sequence of estimates $\{W(k) : k \in \mathbb{N}\}$, where $W(k+1)$ is generated using $W(k)$, the dataset $D$, and possibly some independent noise. SGD is one such procedure which we briefly describe below. At each time $k$, a point $Z(k) \in D$ is chosen according to a predefined strategy, and the estimate $W(k)$ is computed according to

$$W(k+1) = W(k) - \eta_k \nabla \ell(W(k), Z(k)).$$  \hspace{1cm} (13)

Note that $Z(k)$ takes values in the dataset $D$, and therefore has a distribution that corresponds to the empirical distribution of the data. Instead of choosing a single point from $D$, one may also choose a subset of points and evaluate the average value of the gradient evaluated at each of those points. For ease of exposition, we focus on the case when a single point is drawn from $D$. Versions of SGD have been proposed in recent years which modify the update equation to include an additional noise term as follows:

$$W(k+1) = W(k) - \eta_k (\nabla \ell(W(k), Z(k)) + \xi(k))$$  \hspace{1cm} (14)

Adding an independent noise $\xi(k)$ achieves two goals: it improves the generalization error, and it helps the optimization procedure escape shallow local minima. The update equation of SGD may be easily described using a UDC framework as shown in Fig. [5].

**Markov process and action kernel:** It is easy to see from the update equation (14) that $\{W(k) : k \in \mathbb{N}\}$ is a controlled Markov chain. We define the state of this Markov chain as $S(k) = (W(k), Z(k))$, i.e., the current estimate and the current sample drawn from $D$. The performance process is identical to the state. The other input to the action kernel is $\xi(k)$. The action kernel computes the direction to move at time $k$ by evaluating the gradient $\nabla \ell(W(k), Z(k))$ and adding $\xi(k)$ to it. The output of the action kernel is $Y(k)$, and this is an input to the controlled Markov chain. The state of the Markov chain is then updated as per equation (14). The output process $O(k)$ may be set to equal a noisy version of $W(k)$, or a delayed version of the same.

The noise $\xi(k)$ in the above model may be Gaussian noise, in which case we obtain the popular stochastic gradient Langevin dynamics (SGLD) algorithm [56], or uniformly distributed over a shell of suitable radius, in which case we obtain the algorithm of Ge et al. [57]. The UDC framework may also be used to model the algorithms such as momentum-based methods [58] in a similar fashion.

![Fig. 3. Stochastic gradient descent (SGD) via a UDC model.](image)

### B. Adversarial models

Incorporating an action kernel with input $\xi(k)$ also makes it possible to describe adversarial actions during the training process. Adversarial attacks on machine learning algorithms have been extensively studied in the past few years [59]. Many studies have focused on robustness properties of algorithms with respect to small perturbations in the test sample. More recently, other novel adversarial strategies such as data poisoning attacks [60], [61], [62], byzantine gradient attacks [63], [64] have also been proposed. In this subsection, we demonstrate how a UDC framework may be used to model such adversarial settings.

**Markov chain and action kernel:** As in the case of SGD, the controlled Markov chain consists of the state $(W(k), X(k))$ at time $k$, an output $O(k)$, and an input $Z(k)$. The observation process is assumed to convey information concerning the state, such as a noisy version of $W(k)$ or delayed updates of the $W(k)$ process. The input $X(k)$ is sampled minibatch from a dataset $D$, or it is a newly sampled data point. The performance process is identical to the state. The adversarial action is modeled via the action kernel. The adversary has causal access to the output process $\{O(k) : k \in \mathbb{N}\}$. The adversary’s goal is to add a noise $\xi(k)$ to the gradient to impede the training process.

...
are various possible adversary models that we may consider: (a) Gradient perturbation: The noise $\xi(k)$ is variance (or amplitude) constrained, (b) Data poisoning: The adversary corrupts the minibatch $X(k)$ by adding spurious points, or by perturbing some small subset of points. The action kernel takes as input the adversary’s poisoned dataset, which we call $X'(k)$, the performance process $(W(k), X(k))$, and sets $Y(k) = \nabla f(W(k), X'(k)) + \zeta(k)$, where $\zeta(k)$ is additive noise. The output $Y(k)$ is then fed back to the Markov chain, and the state evolves according to the following update equation:

$$W(k + 1) = W(k) - \eta Y(k)$$

There are several interesting problems one can consider in this model. These include an information-theoretic analysis of the generalization error in the presence of an adversary, where the output process $\{O(k) : k \in \mathbb{N}\}$ contains a bounded amount of information about $\{W(k) : k \in \mathbb{N}\}$. Another problem is identifying the impact of the adversary on training error. In particular, how strong does the adversary need to be to ensure that training does not converge? Finally, it would also be interesting to examine UDC models where the adversary has non-causal information concerning the sequence of minibatches, i.e., the process $\{X(k) : k \in \mathbb{N}\}$ that is to be used for training.

VI. QUEUEING SYSTEMS WITH UDC SERVERS

There exists a large volume of literature on queueing systems with time-varying parameters, dating back to the studies by Conway and Maxwell [65], Jackson [66], Yadin an Naor [67], Gupta [68] and Harris [69], most of which focused on the state-dependent service rates. We refer a reader interested in a summary of earlier studies on queues with state-dependent parameters to [70] and references therein.

In recent years, in addition to energy harvesting in wireless networks discussed in Section IV, much attention is given to the issue of scheduling in wireless systems in order to improve user experience (e.g., streaming on a cell phone or a tablet) and to maximize system capacity, using limited spectrum. In wireless systems, such as cellular systems, channel conditions change over time, thereby affecting the probability of successful transmissions even with adaptive transmit power control and modulation and coding schemes [71], [72], [73], [74].

A key difference between the proposed framework and existing studies on wireless scheduling (e.g., [72], [73]) is the following. The studies on wireless scheduling which take time-varying channel conditions into consideration, assume that the channel conditions change independently of the actions taken by the scheduler. In other words, the input process to the action kernel, namely the performance process $W$, does not depend on the past actions of scheduler. Therefore, $W$ can be viewed as an independent exogenous process to the action kernel, and the feedback loop present in Fig. 1 is absent in these studies. As a result, they can be considered a special case of the proposed framework in which the input process $W$ to the action kernel does not depend on the POCMC state.

The performance and management of human operators and servers has been the subject of many studies in the past, e.g., [75], [76], [77], [78]. Recently, with rapid advances in information and sensor technologies, human supervisory control (HSC) became an active research area [10], [79]. In HSC, human supervisors play a crucial role in the systems (e.g., supervisory control and data acquisition (SCADA)) and at times are required to process a large amount of information in a short period with seconds to make a critical decision (e.g., a possible failure of a nuclear reactor due to loss of coolant), potentially causing information overload. For this reason, there is a resurgent interest in understanding and modeling the performance of humans under widely varying settings. Although this is still an active research area, it is well documented that the performance of humans depends on many factors, including arousal and perceived workload [76], [78], [79], [80], [81]. For example, the well-known Yerkes-Dodson law suggests that moderate levels of arousal are beneficial, leading to the inverted-U model [75].

In the remainder of this section, we first illustrate how the proposed UDC framework can be used for two example scenarios in which the service rates depend on either queue lengths or utilization levels. The latter is recently gaining interest for modeling and studying human supervisors [12]. Also, we note that a similar model is applicable to studying and designing scheduling policies for multi-core processors with adaptive control of clock speed or voltage of each core subject to power and thermal constraints [83]. Since the cases in which service rates depend on workloads (instead of queue lengths) can be handled in an analogous manner, we do not discuss them here. Furthermore, we limit our discussion to the cases with fixed arrival rate(s) for the sake of simplicity of our discussion.

A. Queue length-dependent service rates

Many studies consider a server whose service rate depends on the queue length or backlog [65], [69], [70]. Oftentimes, the service rate of a server is allowed to change only when a task is completed or the server starts servicing a new task, e.g., [69]. For example, it is well documented that the performance of human servers (e.g., bank tellers, doctors, nurses, toll collectors) is affected by various factors including the queue length (e.g., [80], [84]). In a more recent study, Chatterjee et al. [85] investigated the capacity of systems in which the reliability or quality of service provided by a server, which is modeled as the channel quality, depends on the queue length from an information-theoretic perspective.

Task arrival processes and task workloads: Suppose that there are $T$ different types of tasks ($T \geq 1$), and denote the set of task types by $T := \{1, \ldots, T\}$. We also define another set $T_+ := T \cup \{0\}$ that includes the null type (type 0). The arrival rate vector $\lambda := (\lambda_1, \ldots, \lambda_T)$ is a $T$-dimensional vector, and the $t$-th element $\lambda_t$ indicates the probability with which a new task of type $t$ arrives at each time $k$ in $\mathbb{N}$, independently of the past and arrivals of other types. In other words, the arrival process $A := \{(A_t(k), \ldots, A_T(k)) : k \in \mathbb{N}\}$ is a collection of $T$ mutually independent Bernoulli processes with rates $\lambda_t$, $t \in T$. Although we assume Bernoulli arrivals to simplify our discussion, more general arrival distributions (e.g., Poisson
distributions) can be handled only with minor changes as it will be clear.

Each new task brings a (random) workload. The workloads of tasks of different types, especially those of tasks that arrive together, may be correlated in some cases. Although such cases can be dealt with in the proposed UDC framework, we make a simplifying assumption that the workloads of tasks belonging to different types are mutually independent.

**Servers with queue length-dependent service rates:** Consider a queueing system with vacation. There are \( L \) servers \((L \geq 1)\), and define the set of servers to be \( L := \{1, \ldots, L\}\). The servers share \( T \) unbounded queues, where the \( t \)-th queue holds incomplete tasks of type \( t \) tasks. The service rates of the \( L \) servers are queue length-dependent in the manner to be made precise. We only study non-preemptive servers here: once a server starts working on tasks, it has to complete the service of the tasks before it can either vacation or work on new tasks.

In addition, when the \( \ell \)-th server goes on a vacation, it remains idle for \( m_\ell \) units of time \((m_\ell \geq 1)\). The case with a random vacation time can be handled with minor changes in the model.

For our discussion, we assume that a server works on at most one type of tasks at any given time. The scenario in which a server can service a batch of tasks of different types can be handled with appropriate changes. In addition, we assume that the scheduling decision at time \( k \) is made prior to the arrivals of new tasks at time \( k \). Hence, new arrivals at time \( k \) are not eligible for scheduling till time \( k + 1 \).

A scheduling policy under consideration is a mapping \( \Theta : S \to \mathcal{D}(T_+^L \times \mathbb{N}^L) \), where \( \mathcal{D}(T_+^L \times \mathbb{N}^L) \) is the set of distributions over \( T_+^L \times N^L \). A scheduling vector is given by a pair \((t, n)\), where \( t := (t_1, \ldots, t_L) \in \mathbb{T}_+^L \) and \( n := (n_1, \ldots, n_L) \in \mathbb{N}^L \). The \( \ell \)-th elements \( t_\ell \) and \( n_\ell \) specify the type and the number of the tasks, respectively, assigned to the \( \ell \)-th server. When \( t_\ell = 0 \), it indicates that the \( \ell \)-th server is asked to rest. In a special case where servers can work on at most one task, i.e., they are not allowed to work on a batch of tasks, the scheduling vector is simply given by \( t \). Similarly, if there is only a single type of tasks, the scheduling vector only stipulates \( n \).

Define \( \mathcal{R}^Q : \mathbb{N}^T \times T_+^L \times \mathbb{N}^L \to \mathbb{R}_+ \) to be the service rate function: suppose (i) the queue length vector is \( q = (q_1, \ldots, q_T) \in \mathbb{N}^T \), where the \( t \)-th element \( q_t \) represents the number of uncompleted type \( t \) tasks, and (ii) \((t, n)\) is a scheduling vector. For a given triple \((q, t, n)\), the service rates of the \( L \) servers are given by \( \mathcal{R}^Q(q, t, n) := (\mathcal{R}^Q_l(q, t, n), \ldots, \mathcal{R}^Q_L(q, t, n)) \) and specify the amount of service that can be performed by each server on the tasks assigned by the scheduling vector \((t, n)\).

Note that this framework allows for heterogeneous servers with varying capabilities and the dependence of service rates across the \( L \) servers. For example, when a processor with multiple cores is subject to a power budget or a thermal constraint, their service rates (i.e., processing speeds) may be dependent. Furthermore, the service rate of a server can depend on the number of tasks that it serves simultaneously.

The aggregate workload of \( n \) type \( t \) tasks \((n \in \mathbb{N} \text{ and } t \in \mathbb{T})\) is modeled using a random variable with distribution \( \mathcal{F}_{t,n} \). For the sake of simplicity, we assume that the workloads of different batches of tasks are mutually independent.

**Task completion probabilities:** When the \( \ell \)-th server works on \( n_{\ell t} \) type \( t_\ell \) tasks, the probability that it will complete the tasks within one unit of time depends on several factors, including the queue lengths and the scheduling vector (via the service rates) as well as the total amount of service that the tasks have already received in the past.

Let \( q \) and \((t, n)\) be the queue lengths and the scheduling vector, respectively. Suppose \( r := (r_1, \ldots, r_L) \) is the cumulative service vector, where the \( \ell \)-th element \( r_\ell \) is the total amount of service that uncompleted tasks of type \( t_\ell \) currently being serviced by the \( \ell \)-th server have received before. If either new tasks are assigned to the server or \( t_\ell = 0 \), we have \( r_\ell = 0 \).

We define a task completion probability function \( \mathcal{C}^Q : \mathbb{N}^T \times T_+^L \times \mathbb{N}^L \times \mathbb{R}_+ \to [0, 1]^L \): for a given quadruple \((q, t, n, r)\), the value of the function \( \mathcal{C}^Q(q, t, n, r) := (\mathcal{C}^Q_1(q, t, n, r), \ldots, \mathcal{C}^Q_L(q, t, n, r)) \) determines the probabilities that the tasks serviced by each server as specified by the scheduling vector \((t, n)\) will be completed during one unit of time. In other words, if \( t_\ell > 0 \), \( \mathcal{C}^Q(q, t, n, r) \) is the probability that the \( n_{\ell t} \) type \( t_\ell \) tasks on which the \( \ell \)-th server works will complete their service during one unit of time, provided that the queue lengths are \( q \) and the tasks have already received \( r_\ell \) amount of service from the server before.

The values of the task completion probability function can be computed from the given workload distributions \( \mathcal{F}_{t,n}, t \in \mathbb{T} \) and \( n \in \mathbb{N} \), and the service rate function \( \mathcal{R}^Q \) as follows: for each \( t \in \mathbb{T} \) and \( n \in \mathbb{N} \), denote a generic random variable with distribution \( \mathcal{F}_{t,n} \) by \( Z_{t,n} \) (which we denote as \( Z_{t,n} \sim \mathcal{F}_{t,n} \) and define \( B_{t,n} : \mathbb{R}^L_+ \to [0, 1] \), where

\[
B_{t,n}(r, \mu) = \mathcal{P}(Z_{t,n} \leq r + \mu | Z_{t,n} > r).
\]

Then, for all \( n \in \mathbb{N}^L \), \( q \in \mathbb{N}^T \), \( r \in \mathbb{R}^L_+ \), and \( t \in \mathbb{T}_+^L \),

\[
\mathcal{C}^Q(q, t, n|r) = \begin{cases} B_{t,n}(r_\ell, \mathcal{R}^Q_l(q, t, n)) & \text{if } t_\ell > 0, \\ 0 & \text{otherwise.} \end{cases}
\]

In a special case of exponentially distributed workloads, the task completion probability function does not depend on the cumulative services \( r \) and \( \mathcal{C}^Q(q, t, n|r) = C^Q(q, t, n|0) \) for all \( r \in \mathbb{R}^L_+ \), where \( 0 = (0, \ldots, 0) \).

**POCMC state:** The POCMC state is modeled using the process \( S^Q = \{(S^Q_t(k), S^Q_t(k), S^Q_t(k), S^Q_t(k), S^Q_t(k)) : k \in \mathbb{N}\} \), where the state at time \( k \), \( S^Q_t(k) \), comprises the following:

(i) \( S^Q_t(k) := (S^Q_1(k), \ldots, S^Q_L(k)) \) is the queue length vector.

(ii) \( S^Q_t(k) := (S^Q_1(k), \ldots, S^Q_L(k)) \) indicates remaining vacation time of each server before it becomes available to service tasks.

(iii) \( S^Q_t(k) := (S^Q_1(k), \ldots, S^Q_L(k)) \) indicates the amounts of service already received by the tasks assigned by the scheduling vector. Recall that \( S^Q_1(k) = 0 \) if either the \( \ell \)-th server rested or completed the service of tasks at time \( k - 1 \). Otherwise, \( S^Q_1(k) \) is the total amount of service.
performed by the server on the uncompleted tasks prior to time $k$.

(iv) $S^u(k) := (S^u_1(k), \ldots, S^u_L(k))$ keeps track of the service rates of the servers at the previous time $k-1$, i.e., $S^u_k(k)$ is equal to the service rate of the $\ell$-th server at time $k-1$.

(v) $S^v(k) := (S^v_1(k), \ldots, S^v_L(k))$ retains the scheduling vector at time $k-1$, i.e., $S^v_k(k) = U(k-1)$ as explained below.

Although there are different queueing models we can consider, here we focus on the following simple model for discussion. The queue length process $S^q := \{S^q(k) : k \in \mathbb{N}\}$ evolves according to

$$S^q_{\ell}(k+1) = S^q_{\ell}(k) + \lambda_{\ell} - \bar{Y}_{\ell}(k), \quad t \in \mathbb{T} \text{ and } k \in \mathbb{N},$$

where $A_{\ell}(k) = 1$ (resp. $A_{\ell}(k) = 0$) with probability $\lambda_{\ell}$ (resp. $1 - \lambda_{\ell}$), and $\bar{Y}_{\ell}(k)$ is the number of type $\ell$ tasks completed at time $k$.

**Performance process:** At each time $k$, the external control input $U(k) := (U_1(k), \ldots, U_L(k))$, which is the scheduling vector for time $k$, is chosen according to a scheduling policy in place. The $\ell$-th element of $U(k)$ is a pair $U_{\ell}(k) = (U_{\ell,1}(k), U_{\ell,2}(k))$ consisting of the type ($U_{\ell,1}(k)$) and the number ($U_{\ell,2}(k)$) of tasks assigned to the $\ell$-th server and takes values in $\mathbb{T}_+ \times \mathbb{N}$.

Suppose that $(S^q(k), S^v(k), S^r(k), S^\kappa(k)) = (\mathbf{q}, \mathbf{r}, \mathbf{\mu})$ and $\bar{U}(k) = (t, n)$. Then, the value of the performance process at time $k$, $W^Q(k) := (W^Q_1(k), \ldots, W^Q_L(k))$, is given by

$$W^Q_{\ell}(k) = \begin{cases} 
B_{\ell, \sigma}(r, \mu t) & \text{if } vt_1 = 0 \text{ and } rt_1 > 0, \\
C^Q_{\ell}(q, t, n|r) & \text{if } vt_1 = 0 \text{ and } rt_1 = 0, \\
0 & \text{otherwise},
\end{cases}$$

for every $\ell \in \mathbb{L}$. Note that $S^q_{\ell}(k) = U_{\ell}(k) = U_{\ell}(k-1)$ in the first case because the servers are assumed non-preemptive.

The performance process given in (16) assumes that the service rate remains constant while a server works on a batch of tasks, which is reflected in the first case of (16). If the service rate changes during the service time of a batch, $S^r(k)$ can be dropped from the POCMC state and the value of the performance process at time $k$ simplifies to

$$W^Q_{\ell}(k) = \begin{cases} 
C^Q_{\ell}(q, t, n|r) & \text{if } vt_1 = 0, \\
0 & \text{otherwise}.
\end{cases}$$

**Output process:** The output of the action kernel at time $k$, namely $Y(k) := (Y_1(k), \ldots, Y_L(k))$, is a vector of mutually independent Bernoulli random variables with parameters $W^Q_{\ell}(k)$, $\ell \in \mathbb{L}$, and indicates the completion of tasks serviced by the servers before time $k+1$: $Y_{\ell}(k) = 1$ if the $\ell$-th server completes the service of the tasks it worked on at time $k$, and $Y_{\ell}(k) = 0$ otherwise. Therefore, the output process $Y := \{Y(k) : k \in \mathbb{N}\}$ takes values in $\mathbb{Y} := \{0, 1\}^L$.

The number of type $\ell$ tasks completed at time $k$, $\bar{Y}_{\ell}(k)$, is determined by the control input $U(k)$ and the output $Y(k)$ of the action kernel according to

$$\bar{Y}_{\ell}(k) = \sum_{\ell=1}^L \left(1 \{U_{\ell,1}(k) = t\} U_{\ell,2}(k) Y_{\ell}(k)\right), \quad t \in \mathbb{T}.$$  

Finally, the transition probability of POCMC can be obtained from the above description and the scheduling policy in place. We describe the model in more detail for the example of quorum policy.

- **Example:** $(l, K)$-quorum system with queue length-dependent service rate

Suppose that there is a single server ($L = 1$) and all tasks are of the same type ($T = 1$). Since $T = 1$, we drop the dependence on the type of task when appropriate. For instance, we write $B_n$ and $Z_n$ in place of $B_{1,n}$ and $Z_{1,n}$, respectively.

Under the $(l, K)$-quorum policy ($1 \leq l \leq K$), the server services tasks in accordance with the following rule. When the server is available to work on new tasks and finds $q$ backlogged tasks, there are two possibilities to consider:

i. If $q$ is smaller than the threshold $l$, the server rests.

ii. If $q$ is at least $l$, it works on a batch of $\min(q, K)$ tasks until their service is completed.

A special case is when $l = 1$, which corresponds to the scenario where the server rests only if the queue is empty. For example, this may describe a shuttle bus or a ferry boat transporting passengers. The shuttle bus driver may wait until at least $l$ passengers are onboard, and the shuttle bus has a capacity of $K$ passengers.

The control input $U(k)$ is the scheduling vector chosen by the $(l, K)$-quorum policy and determines the number of tasks that the server services at time $k$. There are three possibilities to consider based on the above description:

$$U(k) = \begin{cases} 
S^q(k) & \text{if } S^r(k) > 0 \\
\min(S^q(k), K) & \text{if } S^r(k) = 0 \text{ and } S^q(k) \geq l \\
0 & \text{otherwise}.
\end{cases}$$

When the server is working on tasks, its service rate may depend on the number of tasks being served as well as the number of backlogged tasks. This is a discrete-time generalization of the model studied by Neuts [86], for which the service rate is allowed to depend on the queue length. In the earlier example of a shuttle bus or a ferry, for instance, the driver may feel the pressure to shorten the trip times when there are many passengers waiting in line. Also, we assume that the service rate remains constant during the service time of a batch of tasks.

Since the server rests only when there are fewer than $l$ uncompleted tasks, it suffices to model the POCMC state at time $k$, $S^Q(k)$, using the quadruple $(S^q(k), S^r(k), S^\kappa(k), S^\kappa(k))$. For notational simplicity, for every $q \in \mathbb{N}$, we define $q^K := \min(q, K)$. Suppose $S^q(k) = (q, r, \mu, n)$ for some $k \in \mathbb{N}$. The value of performance process at time $k$, $W^Q(k)$ in (16), is given by

$$W^Q(k) = \begin{cases} 
B_n(r, \mu) & \text{if } r > 0, \\
B_{q^K}(0, \mathcal{Q}(q, q^K)) & \text{if } r = 0 \text{ and } q \geq l,
0 & \text{otherwise},
\end{cases}$$

where $B_n(r, \mu) = \mathcal{P}(Z_n \leq r + \mu Z_n > r)$ and $Z_n \sim \mathcal{F}_n$. The output of action kernel, $Y(k)$, is a Bernoulli random variable with parameter $W^Q_{\ell}(k)$.

**Transition probability of POCMC:** The following map $S^Q$ describes the transition probabilities of POCMC. For ease
of exposition, we break it into three cases. Let \( s = (q, r, \mu, n) \) and \( s^+ = (q^+, r^+, \mu^+, n^+) \):

**Case 1.** \( q < l \) (the server rests):

\[
S^Q(s^+|s, 0, 0) = \begin{cases} 
\lambda & \text{if } q^+ = q + 1 \text{ and } r^+ = \mu^+ = n^+ = 0 \\
1 - \lambda & \text{if } q^+ = q \text{ and } r^+ = \mu^+ = n^+ = 0 \\
0 & \text{otherwise}
\end{cases}
\]

**Case 2.** \( r > 0 \) (the server worked on tasks at the previous time but did not complete their service):

\[
S^Q(s^+|s, n, y) = \begin{cases} 
\lambda & \text{if } q^+ = q + 1 - ny, r^+ = (r + \mu)(1 - y), \\
\mu^+ = \mu \text{ and } n^+ = n & \text{otherwise}
\end{cases}
\]

**Case 3.** \( q \geq l \) and \( r = 0 \) (the server becomes available for new tasks, and finds at least \( l \) backlogged tasks): let \( \mu = \mathcal{R}^Q(q, k^Q) \).

\[
S^Q(s^+|s, q^K, y) = \begin{cases} 
\lambda & \text{if } q^+ = q - q^K + 1, r^+ = (r + \mu)(1 - y), \\
\mu^+ = \mu \text{ and } n^+ = q^K & \text{otherwise}
\end{cases}
\]

\( \mathcal{R}^Q(q, k^Q) \) gives the rate function and a task completion probability function given in Theorem 1.

**B. Utilization-dependent service rates**

In many cases of interest, the service rates of servers depend on their (recent) utilization levels. For example, the efficiency of human servers is not constant and varies with several factors, such as arousal and fatigue \([12], [81]\). Hence, in many applications with human servers making critical decisions (e.g., traffic control and nuclear plant monitoring), it is important to take into account the efficiency and alertness of human servers in order to improve the performance of overall systems \([12]\).

The case in which the service rate of a server varies as a function of its utilization level can be handled in a similar manner. For our discussion, we assume the same task arrival processes and setup with \( T \) types of tasks served by \( L \) servers, which are described in Section VI-A.

Let \( S^U := \{S^U(k) : k \in \mathbb{N}\} \), \( S^U := (S^U_1(k), \ldots, S^U_k(k)) \), be the process that tracks the utilization levels of the \( L \) servers. For example, \( S^U_\ell(k), \ell \in \mathbb{L} \), could represent the utilization ratio or the utilization ratio with forgetting factor \( \alpha \) of the \( \ell \)-th server (provided in Definitions 2 and 3 of Section III-A). In the remainder of this section, without loss of generality, we assume that the utilization levels are non-negative. Furthermore, for the simplicity of discussion, we only consider exponentially distributed workloads. More general workload distributions can be handled as discussed in Section VI-A.

The POCMC is given by \( S^U = \{S^U_1(k), S^U_2(k), \ldots, S^U_L(k) \} : k \in \mathbb{N}\} \), where \( S^U(k) := (S^U_1(k), \ldots, S^U_L(k)) \) indicates the availability of each server to take on new tasks. In other words, \( S^U_\ell(k) = 1 \) if the \( \ell \)-th server is available to service new tasks at time \( k \) (either after completing the tasks or resting at time \( k - 1 \)), and \( S^U_\ell(k) = 0 \) otherwise. Note that, because the workloads are assumed exponentially distributed, \( S^U(k) \) can be dropped from the POCMC state, thanks to the memoryless property of exponential distributions.

The control input at time \( k \), \( U(k) = (U_1(k), \ldots, U_L(k)) \), is the scheduling vector and is determined by the employed scheduling policy. Recall that \( U_\ell(k) \) is given by a pair \( (U_{\ell,1}(k), U_{\ell,2}(k)) \) and, if \( U_{\ell,1}(k) = 0 \), the \( \ell \)-th server rests at time \( k \).

The performance process at time \( k \), \( W^U(k) = (W^U_1(k), \ldots, W^U_L(k)) \), depends on \( S^U(k) \) and \( U(k) \), and reflects the efficiency of the \( L \) servers as a function of their utilization levels. This is determined with the help of a service rate function and a task completion probability function given by \( \mathcal{U}: \mathbb{R}_{\geq 0}^L \times \mathbb{T}^L \times \mathbb{N}^L \to \mathbb{R}_+^L \) and \( C^U: \mathbb{R}_{\geq 0}^L \times \mathbb{T}^L \times \mathbb{N}^L \to [0,1]^L \), respectively: \( \mathcal{U}_\ell(\eta, t, n) \) denotes the service rate of the \( \ell \)-th server as a function of the server utilization levels \( \eta := (\eta_1, \ldots, \eta_L) \) and the scheduling vector \( (t, n) \).

Similarly, \( C^U_\ell(\eta, t, n), \ell \in \mathbb{L} \), is the probability with which the \( \ell \)-th server completes the \( n_t \) tasks of type \( t \) in one unit of time. Because the workloads are exponentially distributed, we have

\[
C^U_\ell(\eta, t, n) = \begin{cases} 
\mathbb{P}(Z_{t,n} \leq \mathcal{U}_\ell(\eta, t, n)) & \text{if } t \neq 0, \\
0 & \text{otherwise}.
\end{cases}
\]

Suppose that \( S^U_\ell(k) = \eta \) and \( U(k) = (t, n) \). Then, for all \( \ell \in \mathbb{L} \), \( W^U_\ell(k) = C^U_\ell(\eta, t, n) \).

Analogously to the previous case of queue length-dependent service rates, the output of the action kernel is a vector of mutually independent Bernoulli random variables with parameters in \( W^U(k) \) and indicates which servers completed the tasks of the type chosen by the control input. The queue lengths evolve according to \( \mathbb{E}^U_\ell(k) \) with the number of type \( t \) tasks completed at time \( k \) given by \( \mathbb{E}^U_\ell(k) \) for all \( t \in \mathbb{T} \).

In order to describe the POCMC state transition probabilities, we still need to know how the utilization levels evolve as a function of the current POCMC state, in particular, the current utilization levels, and the control input. We capture the transition probabilities of utilization levels using a map \( \Lambda^U: \mathbb{R}_{\geq 0}^L \times \mathbb{T}^L \times \mathbb{N}^L \to [0,1] \), where \( \Lambda^U(\eta, t, n) \) is the probability that the utilization levels will transition from \( \eta \) to \( \eta^+ \) when the control input is \( (t, n) \). In a simple setting, the transition probability of the utilization level of the \( \ell \)-th server may depend only on its current utilization level and whether or not the control input requires it to work on tasks.
Example: Throughput-optimal scheduling policy for a single server with a utilization-dependent service rate

In a recent study, Lin et al. [32] studied the influence of server utilization in a system with one server servicing a single type of tasks whose workloads are exponentially distributed. We shall use this study as an example to illustrate how the UDC framework can be used to investigate the problem of designing a simple yet efficient scheduling policy for systems in which service rates depend on utilization. This study also illustrates how the UDC framework allows us to leverage a simpler system to facilitate the analysis. Since there is only one type of tasks, we drop the dependence on the type of task as explained earlier.

The task arrival rate is denoted by \( \lambda > 0 \). The server is allowed to work on at most one task. Therefore, the control input \( U(k) \) at time \( k \) can be specified using a binary value: \( U(k) = 1 \) if the server works on a task at time \( k \), and \( U(k) = 0 \) otherwise. Here, as mentioned earlier, we take the view that \( U(k) \) represents the number of tasks that the server works on at time \( k \). Similarly, \( S^a(k) = 1 \) if the server is available to work on a new task at time \( k \), and \( S^a(k) = 0 \) otherwise (indicating that the server is still working on a task that it did not complete at time \( k - 1 \)).

The utilization level of the server is modeled using a controlled Markov chain \( S^u = \{S^u(k) : k \in \mathbb{N} \} \) that takes values in a finite set \( S^u := \{1, \ldots, s_{\text{max}} \} \). The transition probabilities of the utilization level \( S^u(k) = \eta \) at time \( k \) depend on (i) the current value of utilization, \( \eta \), and (ii) the control input \( U(k) \), and are governed by the following mapping:

\[
\Lambda^U(\eta^+, \eta, n) = \begin{cases} 
\xi^+_\eta & \text{if } n = 1, \eta^+ = \min(s_{\text{max}}, \eta + 1) \\
1 - \xi^+\eta & \text{if } n = 1, \eta^+ = \eta \\
\xi^-\eta & \text{if } n = 0, \eta^+ = \max(1, \eta - 1) \\
1 - \xi^-\eta & \text{if } n = 0, \eta^+ = \eta 
\end{cases}
\]

It is clear from the given transition probabilities that if the server works on a task (resp. rests) at time \( k \), the utilization level either remains at \( \eta \) with probability \( 1 - \xi^+\eta \) (resp. \( 1 - \xi^-\eta \)) or goes up by one with probability \( \xi^+\eta \) if \( \eta < s_{\text{max}} \) (resp. goes down by one with probability \( \xi^-\eta \) if \( \eta > 1 \)) with the convention \( \xi^-\eta = 0 \).

When the server works on a task, its service rate depends on its current utilization level, \( \eta \). For notational simplicity, we define \( \tilde{C}^U : S^u \to [0, 1] \) such that \( \tilde{C}^U(\eta) = C^U(\eta, 1) \) for all \( \eta \in S^u \). In other words, \( \tilde{C}(\eta) \) is equal to the probability that the server will complete a task within one unit of time when its utilization level is \( \eta \).

Transition probability of POCMC: For this model, it is not necessary to retain the control input at the previous time. Hence, the state of POCMC at time \( k \) reduces to a triple \( S^U(k) = (S^q(k), S^a(k), S^u(k)) \).

Based on the mapping \( \Lambda^U \), the transition probabilities of the POCMC are as follows: for all \( (q, \eta, a), (q^+, \eta^+, a^+) \in S^U = \mathbb{N} \times S^u \times \{0, 1\} \),

\[
\begin{align*}
\Phi^U((q^+, \eta^+, a^+)| (q, \eta, a), 0, 0) = \\
&= \begin{cases} 
\Lambda^U(\eta^+| \eta, 0) & \text{if } q^+ = q + 1 \text{ and } a^+ = 1 \\
(1 - \lambda) \Lambda^U(\eta^+| \eta, 0) & \text{if } q^+ = q \text{ and } a^+ = 1 \\
0 & \text{otherwise}
\end{cases}
\end{align*}
\]

Case 1. \( n = 0 \) (the server rests):

\[
\begin{align*}
\Phi^U((q^+, \eta^+, a^+)| (q, \eta, a), 0, 0) &= \begin{cases} 
\Lambda^U(\eta^+| \eta, 0) & \text{if } q^+ = q + 1 \text{ and } a^+ = 1 \\
(1 - \lambda) \Lambda^U(\eta^+| \eta, 0) & \text{if } q^+ = q \text{ and } a^+ = 1 \\
0 & \text{otherwise}
\end{cases}
\end{align*}
\]

Case 2. \( n = 1 \) (the server works on a task):

\[
\begin{align*}
\Phi^U((q^+, \eta^+, a^+)| (q, \eta, a), 1, y) &= \begin{cases} 
\Lambda^U(\eta^+| \eta, 1) & \text{if } q^+ = q + 1 - y \text{ and } a^+ = y \\
(1 - \lambda) \Lambda^U(\eta^+| \eta, 1) & \text{if } q^+ = q - y \text{ and } a^+ = y \\
0 & \text{otherwise}
\end{cases}
\end{align*}
\]

Stationary randomized scheduling policies and stability: The authors of [32] considered the following class of stationary policies that map the current state of POCMC to the probability of scheduling a task when the queue is not empty.

**Definition 12.** An admissible stationary randomized scheduling policy (SRSP) is a mapping \( \Theta : S^U \to [0, 1] \) such that, for all \( (q, \eta, a) \in S^U \), \( \Theta(q, \eta, a) \) is the probability that the server is asked to work on a task when the controlled Markov chain state is \( (q, \eta, a) \).

Recall that, because the server is assumed non-preemptive, once it starts working on a task, it is required to continue to service the task until completion and, hence, \( \Theta(q, \eta, 0) = 1 \) for all \( q > 0 \) and \( \eta \in S^u \). Also, for a fixed SRSP \( \Theta \), the controlled Markov chain \( S^U \) is a discrete-time Markov chain with a countable state space.

**Definition 13.** For a fixed task arrival rate \( \lambda > 0 \), the controlled Markov chain \( S^U \) under a chosen SRSP \( \Theta \), denoted by \( S^U_\Theta \), is said to be **stable** if

1. there exists at least one recurrent communicating class of \( S^U_\Theta \);
2. all recurrent communicating classes are positive recurrent; and
3. the number of transient states is finite.

In addition, \( \Theta \) is said to stabilize \( S^U \) for \( \lambda \).

Using this notion of stability, the authors of [32] investigated the problem of designing a throughput-optimal scheduling policy that stabilizes \( S^U \) for any arrival rate \( \lambda \) for which there exists a stabilizing SRSP. In order to facilitate their analysis, they made use of a virtual queue that always has a task waiting for service when the server becomes available. Removing the queue size, the state of the virtual queue is given by the process \( S^U : = \{(S^u(k), S^a(k)) : k \in \mathbb{N} \} \).

**Threshold scheduling policies:** In order to identify a throughput-optimal scheduling policy with simple structure, the authors of [32] focused on a family of threshold policies: fix \( \tau \in S^{\eta^+} := \{1, \ldots, s_{\text{max}} + 1\} \). A threshold (scheduling) policy for the virtual queue with threshold \( \tau \) is a deterministic scheduling policy given by a mapping \( \Phi_\tau : S^u \times \{0, 1\} \to \{0, 1\} \), where

\[
\Phi_\tau(q, a) := \begin{cases} 
0 & \text{if } \eta \geq \tau \text{ and } a = 1, \\
1 & \text{otherwise.}
\end{cases}
\]
Clearly, when the server is available to service a new task, the threshold policy $\Phi_\tau$ assigns a new task if and only if the utilization threshold is less than the threshold $\tau$. The virtual queue $\mathcal{S}^U$ under a threshold policy $\Phi_\tau$ for $\tau > 1$ can be modeled using a finite-state Markov chain with a unique stationary distribution $\bar{\pi}_\tau$ concentrated on the set
\[
\mathcal{S}_\tau := \{(\eta, a) | \eta \in \{\tau - 1, \ldots, s_{\max}\}, a \in \{0, 1\}\}.
\]
Define
\[
\lambda^* := \max_{\tau \in \mathcal{S}_\tau} \left( \sum_{(\eta, a) \in \mathcal{S}_\tau} \bar{\pi}_\tau(\eta, a) \Phi_\tau(\eta, a) \bar{C}^U(\eta) \right) \tag{19}
\]
to be the maximum average task completion rate for the virtual queue among all threshold policies of the form in \[18\]. Let $\tau^*$ be a maximizer of the right-hand side of $19$. Let $\tau^*$ be the maximum average task completion rate for the virtual queue. Thus, the scheduling policy $\theta^*$ is defined as $\theta^* := \max_{\tau \in \mathcal{S}_\tau} \Phi_\tau$. The authors of \[82\] showed that if there is a stabilizing $\Phi^*$, the workloads of tasks are modeled using i.i.d. random variables.

Remark 2. Throughput-optimal scheduling policy $\theta^*_\tau$.
i. The value of an optimal threshold $\tau^*$ can be easily identified by solving the optimization problem in (19) by searching through the finite set $\mathcal{S}^U$ with $s_{\max} + 1$ elements. Thus, this greatly simplifies identifying the throughput-optimal policy provided in (20).

In a closely related study, Savla and Frazzoli \[12\] investigated a similar problem of designing a task release control policy. There are two key differences between these two studies. First, the model employed by Savla and Frazzoli assumes that the service time function is convex, which is analogous to the service rate function employed in \[82\] being unimodal. Lin et al., however, do not impose any assumption on the service rate function. Second, a threshold policy is proved to be maximally stabilizing only for the case with identical task workload by Savla and Frazzoli. In the study of Lin et al., the workloads of tasks are modeled using i.i.d. random variables.

VII. REMOTE ESTIMATION ACROSS A PACKET-DROP LINK POWERED BY ENERGY HARVESTING

We begin this section by describing a UDC consisting of a packet-drop link powered by energy harvested and stored according to the models delineated in Section III-B. The apportionment of energy for transmission of information across the link over time is governed by a control process. We then proceed to discussing a few research themes in which the link is used in a remote estimation context.

A. Packet-drop links powered by energy harvesting

At each time $k$, the link can either convey unerringly a symbol in $\mathcal{X}$ or a packet drop occurs. Implementation of the packet-drop link using wireless communication requires, for each $k$, that a codeword appropriately encoding $X(k)$ is placed for transmission across one or more physical channels. The transmission of a codeword will, in general, require multiple uses of each channel. A decoder at the receiver attempts to recover $X(k)$ and a packet drop occurs when it fails due to an outage caused by fading, interference or other detrimental effects. If $\mathcal{X}$ is infinite, such as when it is a real coordinate space, we assume that the codeword length is large enough to encode $X(k)$ with negligible quantization error.

Definition 14. (EH packet-drop link) A packet-drop link comprises an action kernel whose output alphabet is $\mathcal{Y} := \mathcal{X} \cup \{\mathcal{E}\}$, where $\mathcal{E}$ indicates a packet drop. The input-output relationship is specified as follows:

\[
Y(k) = \begin{cases} X(k) & \text{if } L(k) = 1 \\ \mathcal{E} & \text{if } L(k) = 0 \end{cases} \tag{21}
\]

where the link process $L$ indicates that there is a successful transmission when $L(k) = 1$ and the packet is dropped otherwise. We assume that a map $L : \mathcal{Y}^k \rightarrow [0, 1]$ characterizes $L$ probabilistically as follows:

\[
P_{L(k)}(s^k, u^k) = C(w^k), \quad k \in \mathbb{N}, \ s^k \in \mathcal{S}^k, \ u^k \in \mathcal{U} \tag{22}
\]

which quantifies the probability of packet drop. Here, $W^k$ and $S^k$ are obtained from the EH model described in Definition 5 or a simplified version, such as the one specified in Definition 4. In addition, we assume that $L(k)$, $S^k(k+1)$ and $O(k)$ are conditionally independent given $S^k(k)$ and $U(k)$.

In a wireless communication setting, an outage causing a packet drop occurs when fading, which is stochastic in general, attenuates the transmitted signal to a point that the received power is below a threshold needed for decoding \[74, 87\]. The threshold depends on the codeword length, noise, interference characteristics \[88\] and it may also be stochastic. Here, we assume that fading and the transmission power are constant during the transmission of the codeword encoding $X(k)$. Moreover, $W^k(k)$ represents the total energy used attempting to transmit $X(k)$. Hence, $L$, which quantifies the probability of outage given the transmission power as in \[22\], is a non-increasing function that can be determined on a case-by-case basis, such as in \[89\].
B. Design of remote estimation systems: problem definitions

Henceforth, we prioritize the discussion of research on the design of remote estimation systems. Our choice is motivated not only by applications, such as monitoring of physical processes, but also by relevance for the design of control systems.

We consider the configuration depicted in Figure 4 in which an estimator $\mathcal{E}$ is a causal map that is possibly time-varying, and seeks to reconstruct a process $V$ based on information sent to it via a packet-drop link according to $\mathcal{E} : y(1:k) \mapsto \nu^\mathcal{E}(k)$, for $k \in \mathbb{N}$. A transmitter is a causal map $\mathcal{T}$ that is possibly time-varying, and uses $V^\mathcal{T}$ and $O$ to produce $X$ and $U$ according to $\mathcal{T} : (V^\mathcal{T}(1:k), O(1:k)) \mapsto (X(k), U(k))$, for $k \in \mathbb{N}$. In most cases of interest $V^\mathcal{T}$ is either $V$ itself, or a causal function of $V$ possibly disrupted by additive or multiplicative noise. We refer to the pair $(\mathcal{T}, \mathcal{E})$ in conjunction with the UDC that specifies the EH packet-drop link as a remote estimation system.

Remark 3. Synchronization between $\mathcal{T}$ and $\mathcal{E}$
Notice that one-step delayed feedback from the output of the link can be made available to $\mathcal{T}$ through $O$ by augmenting the state of the POCMC so as to include $Y(k-1)$. When such a feedback is present, a copy of the estimate $V^\mathcal{E}(k-1)$ can be replicated by $\mathcal{T}$ at time $k$. This synchronization often simplifies the joint design of $\mathcal{T}$ and $\mathcal{E}$ to meet stability or optimality conditions.

We proceed with discussing the chronology of research on the design of remote estimation systems and control, with emphasis on the former.

Problem 1. (Optimal remote estimation system design)
Let an EH packet-drop link, the joint probabilistic description of $V^\mathcal{T}(1:k)$ and $V(1:k)$ for all $k \in \mathbb{N}$ be given. For predetermined sets $\mathcal{T}$ and $\mathcal{E}$ of allowable transmitters and remote estimators, respectively, determine whether a pair $(\mathcal{T}, \mathcal{E})$ exists that is optimal with respect to a given figure of merit $\mathcal{J} : \mathcal{T} \times \mathcal{E} \to \mathbb{R}_+$ that should assess the quality of $V^\mathcal{E}$ relative to $V$ and can include additional costs. If such a pair exists, determine one.

Unless stated otherwise, we assume the following widely-used covariance-based cost structure:

$$
\mathcal{J}^{(2q, K)}(\mathcal{T}, \mathcal{E}) := \frac{1}{K} \sum_{k=1}^{K} \mathcal{E} \left[ \left( (V^\mathcal{E}(k) - V(k))^T (V^\mathcal{T}(k) - V(k)) \right)^q \right]^{\frac{1}{q}} \tag{23}
$$

where $q$ is a positive integer and $K$ indicates the length of the optimization horizon.

Stabilizability in the $m$-th moment sense, as defined below, is another relevant design objective.

Problem 2. ($m$-th moment stabilizability)
Let an EH packet-drop link, the joint probabilistic description of $V^\mathcal{T}(1:k)$ and $V(1:k)$ for all $k \in \mathbb{N}$ be given. Consider that the $m$-th moment of $V$ is unbounded. Determine whether a pair $(\mathcal{T}, \mathcal{E})$ exists for which the $m$-th moment of $V(k) - V^\mathcal{E}(k)$ is bounded for all $k \in \mathbb{N}$. If such a pair exists, determine one.

Notice that the existence of a solution that is optimal for $\mathcal{J}^{(2q, K)}(\mathcal{T}, \mathcal{E})$ in the limit when $K$ tends to infinity may imply, under certain conditions, $2q$-th moment stabilizability.

When either $\mathcal{T}$ or $\mathcal{E}$ is a singleton in Problems 1 or 2 we say that the associated design problem is of the single-block type, and we qualify it as two-block otherwise.

Remark 4. (Relevance of remote estimation for control systems)
There are at least two scenarios for which Problems 1 or 2 are relevant in the context of control systems. The first is when a packet-drop link connects the sensors that access the output of the plant to the controller. In this case, the transmitter is collocated with the sensors and the remote estimator is typically a component of the controller. The second setting is when the controller includes a transmitter to send its command signals to a remote estimator that is collocated with the actuator. A combination of both cases is also possible.

C. Uncontrolled transmission: optimal policies
As is surveyed in [6], the design of stabilizing and, whenever possible, optimal estimation and control systems whose components communicate via packet-drop links has been an active research topic for at least fifteen years. Early work assumed that the link process $L$ was an uncontrolled time-homogeneous Markov chain. This assumption is realistic when the fading process, as indexed by $k$, is a real-valued time-homogeneous Harris chain and $\mathcal{T}$ does not have the authority to select the transmission power, which may be kept constant thanks to a dependable energy supply.

Henceforth, we limit our discussion to remote estimation systems in which $V$ and $V^\mathcal{T}$ are obtained as follows:

$$
V(k+1) = AV(k) + N(k), \quad k \in \mathbb{N} \tag{24a}
$$

$$
V^\mathcal{T}(k) = C V(k) + N^\mathcal{T}(k), \quad k \in \mathbb{N} \tag{24b}
$$

where $A$ and $C$ are real matrices of appropriate dimensions and the noise processes $N$ and $N^\mathcal{T}$ are independent and white with nonsingular covariance. In the context of control systems, an additional input term may be present in the right hand side of (24a) and (24b).

At first, the effect of uncontrolled packet drops was modeled as multiplicative noise [93], [94], which makes the analysis of stability and second moment optimal design amenable to techniques inspired on Markovian jump linear system theory [95]. Typically the noise process would be Bernoulli, which would
take value 0 when a drop occurs. In a control systems setting, these multiplicative noises could affect the links carrying sensor measurements to the controller and controls signals to the actuator. Most approaches focused on single-block design, which, depending on which links suffer packet drops, would be either a component at the sensors that processes measurements prior to transmission, a controller [93] or a remote estimator. As a consequence of the simplicity of the single-block framework, optimal policies and tight stabilizability conditions for state estimation and control can be obtained even when there is no link output feedback [94], [95], which can be viewed as a form of user datagram protocol (UDP).

The two-block remote estimation system formulated in [96] was the first to consider the simultaneous design of and . When is detectable [97], [98] the approach in [96], which is specified in continuous-time, can be immediately adapted to our discrete-time framework. In such a case, when is a Bernoulli process, the remote estimation system is moment stabilizable if and only if the following condition holds:

\[ p^{\text{drop}} \rho(A)^m < 1 \]  

(25)

where \( p^{\text{drop}} := P_L(0) \) is the probability of drop and \( \rho(A) \) is the spectral radius of \( A \). As is shown in [96], a stabilizing solution is obtained by selecting as a Kalman filter and as its state followed by a properly designed estimator . Subsequent work in [99] showed that the scheme in [96] is optimal with respect to a quadratic cost when and are independent white Gaussian processes. Stabilizability in a control systems context was characterized in [100] using similar techniques for the case in which measurements are conveyed to the controller using two packet-drop links, with each having a distinct transmitter block. The setting in which a packet-drop link conveys command signals from the controller to the actuator was investigated in [101].

Interestingly, (25) can be obtained as the limiting case [102] when tends to infinity of the condition in [103], [104] that characterizes stabilizability when a -ary erasure channel connects the transmitter to the remote estimator.

D. Controlled transmissions without packet drops

We now consider the case in which transmissions may be controlled through , while and are modeled by [24]. When restricted to the remote estimation framework adopted here, in which must be designed to appropriately generate both and , controlled transmissions were first studied in a stabilizability context in [96].

In [96], incorporates a Kalman filter that uses to generate a local estimate of . In addition, it implements policies that use the magnitude of to determine the likelihood that a transmission is requested, which requires synchronization between and so that can be reconstructed at the transmitter. Notice that, in the absence of packet drops, and can be synchronized without the need for feedback through since can be causally computed at the transmitter based on and . In this context, when is scalar and the noises and are Gaussian, a policy that requests a transmission when the magnitude of exceeds a threshold was later shown [3] in [105] to be optimal jointly with a Kalman-like estimator, with respect to a cost that linearly combines the expected squared estimation error and the time-averaged probability of transmission. As reported in [107], threshold-type policies remain optimal when has dimension two or higher, provided that is a scaled orthogonal matrix. Although [103] shows that a jointly optimal transmitter and estimator pair exists for the aforementioned setting even when is any real-valued matrix, the question of whether there is a jointly optimal pair admitting threshold-type policies for transmission remains an open problem. Certainty equivalence properties for these estimators, which are relevant for the design of optimal controllers, are investigated in [109]. Optimal strategies subject to restrictions on the total number of transmissions were determined in [93]. Results reported in [110] show that threshold-based schemes can be adapted to guarantee stabilizability of a system formed by a network of plants and controllers connected by multiple packet-drop links.

The framework in [107] was the first, in the context of remote estimation considered here, to allow for transmission policies that account for energy harvesting. Notably, it considers that is generated based not only on but also on , as determined by the linear-saturated EH model in [103] for which the arrival process is assumed i.i.d. and is normalized so that each transmission at time requires . When the noises in [24] are zero-mean Gaussian and there are no packet drops, it follows from [107] Theorems 3 and 4] that there are transmission and estimation policies with the structure in (26) and (27), respectively, that are jointly optimal for the scalar case.

\[ V^e(k) = \begin{cases} AV^e(k-1) & \text{if } Y(k) = \mathcal{E} \\ X(k) & \text{if } Y(k) \neq \mathcal{E} \end{cases} \quad k \in \mathbb{N} \]  

(26)

\[ U(k+1) = \begin{cases} 1 & \text{if } |\hat{V}(k+1) - AV^e(k)| > \mathcal{G}(k, S^{\text{soc}}(k+1)) \\ 0 & \text{otherwise} \end{cases} \]  

(27a)

\[ X(k) = \hat{V}(k), \quad k \in \mathbb{N} \]  

(27b)

Here, \( \mathcal{G} \) is a threshold that depends on time and the state of charge \( S^{\text{soc}}(k) \). The threshold determines when \( U(k) = 1 \), in which case a transmission setting \( Y(k) \) to \( X(k) \) is requested at time \( k \). Methods to determine \( \mathcal{G} \) are described in [107]. It is remarkable that a policy pair with the simple structure in (26) and (27) is jointly optimal, which also guarantees that it accomplishes the best trade-off between transmitting at time \( k \) or saving energy to transmit later.

It is important to note that, barring the dependence of the thresholds for transmission on \( S^{\text{soc}}(k) \), (26) and (27) are akin to the optimal policies in [103], [106]. As is explained in [107], one way to obtain these results is to establish that there is a jointly optimal policy pair whose estimator has the structure (26), after which the problem of finding a

---

4The techniques and results in [105] are to a significant extent equivalent to the research reported in [106] for paging and registration policies.
corresponding optimal transmission policy can be cast as an MDP [111] whose state is finite dimensional because \( T \) and \( \mathcal{E} \) are synchronized. Subsequently, well-known results can be invoked to prove that restricting transmission policies to be memoryless functions of the state of the MDP incurs no loss of optimality. Properties of the probability distributions of the noises, such as symmetry and unimodality, are used to show that there is no optimality loss when these policies are further restricted to be of the form [27].

E. Controlled transmissions with packet drops and perfect feedback

In this subsection, we discuss recent work for the framework that extends of that of Section VII-D by allowing packet drops in the link that connects the transmitter to the remote estimator.

Assumption 1. Unless noted otherwise, here we assume that there is a causal map with which \( Y(k) \) can be recovered unerringly from \( O(1:k+1) \), for all \( k \in \mathbb{N} \), which also implies that \( T \) and \( \mathcal{E} \) can be synchronized.

Assumption 2. We also assume that \( S^{\text{occ}}(k) \) and \( S^\psi(k) \) can be recovered from \( O(1:k) \), for all \( k \in \mathbb{N} \).

We proceed to defining and subsequently discussing advantages and properties of a class of covariance-based transmission policies, which has been adopted in [112], [113], [114], [115], to list a few.

A transmission policy is classified as covariance-based when the dependence of \( U \) on \( V^\psi \) and \( O \) can be recast in terms of a matrix-valued process \( P^\psi \) that is determined from \( Y \) as follows, for \( k \in \mathbb{N} \):

\[
P^\psi(k) := \mathcal{L}\{ (V(k) - V^\psi(k))^T (V(k) - V^\psi(k)) \mid Y(1:k) \}
\]

where \( P^\psi(0) \) is predetermined.

There is a recursive time-update mechanism [112] for \( P^\psi \) that guarantees that it is an information state [116], which, as we discuss below, may be used to recast the underlying optimization as an MDP, subject to the following set of policies.

Definition 15. ( \( T^c \) - Memoryless covariance-based transmission policy set) We use \( T^c \) to denote the set of transmitters for which there is a map \( \mathcal{I}^c \) determining \( U \) according to \( \mathcal{J}^c \) : \( P^\psi(k-1), S^{\text{occ}}(k), S^\psi(k) \rightarrow U(k) \).

Now, consider the formulation in [112], in which for each \( k \) \( T \) selects \( X(k) \) equal to \( V(\hat{k}) \), and \( U(k) \) is either zero (no transmission) or a pre-selected energy quantum, as opposed to allowing two or more energy levels. A transmitter that seeks to convey \( X(k) = \hat{V}(k) \) to the estimator is often labeled smart sensor to distinguish it from the scheme in [114], which attempts to forward the unprocessed measurements by setting \( X(k) \) equal to \( V^\psi(k) \). If \( N \) and \( N^\psi \) are Gaussian then there is a tractable method to design remote estimation systems that are optimal subject to the restriction that \( T \) is in \( T^c \). Namely, in exchange for the possible loss of optimality that results from this restriction, as pointed out in [112], there is no further loss of optimality by also assuming that the estimator has the structure in [26]. This property allowed the authors of [112] to show that there are coordinate-wise threshold transmission policies that are optimal among those in \( T^c \). In spite of these advantages, there is no known bound on the performance loss incurred by this method.

The authors of [113] investigated methods to determine optimal power selection policies when the probability of outage depends exponentially on the transmission power, which in their framework is allowed to vary among two or more levels. Notably, short of allowing for varying transmission power levels, the formulation of [113] is analogous to the one in [112]. Notwithstanding their similarities, the analysis in the former demonstrates why allowing the transmitter to select among multiple power levels complicates significantly the search and characterization of optimal policies. In order to contend with the complexity of the problem, work in [113] includes useful approximations and tractable methods. The analysis and framework in [117], which also examines a control problem, provides suboptimal policies and numerical methods to address the case in which Assumptions 1 and 2 are not satisfied.

Tight necessary and sufficient conditions for the existence of a transmission policy that stabilizes the estimation error in the second-moment sense have been recently determined in [118]. The formulation in [118] considers memoryless policies that use the state of charge \( S^{\text{occ}}(k) \) to decide, at each time \( k \), whether a transmission should be attempted and if so at which power level \( U(k) \). In order to state the stabilizability conditions, we refer to a map \( L^c : S^{\text{occ}} \rightarrow [0, 1] \) representing the probability of outage in terms of the state of charge when a transmission is requested. More specifically, the map \( L^c \), which is represented with \( d \) in [118], must quantify the combined effect of the power selection policy, the battery model that yields the effective power \( W^c(k) \) and \( L(W^c(k)) \), which quantifies the outage probability according to [22]. The probability that a transmission is requested at time \( k \) is a function of \( S^{\text{occ}}(k) \), specified by a randomized transmission-request policy map, which is represented as \( L^c : S^{\text{occ}} \rightarrow [0, 1] \) and is denoted as \( \theta \) in [118]. Theorem 3.1 in [118] states that given \( L^c \), there is a stabilizing transmission-request policy if only if the following inequality holds:

\[
\lambda^c \rho(A)^2 \leq 1
\]

where the nonnegative real constant \( \lambda^c \) is a function of \( L^c \) and \( S^\psi \), which specify the probabilities of transition of \( S^{\text{occ}} \) in terms of \( U \). In addition, it is stated in [118] Theorem 3.1] that it suffices to consider deterministic transmission-request policies and according to [118] Theorem 3.2] the search can be further narrowed to threshold policies when \( L^c \) is non-increasing.

Notice that [23] is a generalization of [25] and the two conditions coincide when \( L^c \) is constant and equal to \( p^{\text{occ}} \).

VIII. Conclusions and future directions

Our overview of the concepts, formulations, and methods utilized on the research themes expounded in Sections IV - VII evinces not only the similarities elicited by the presence of a UDC, but it also unveils a clear distinction among the
objectives, techniques and assumptions adopted in each theme. This disconnection creates new research opportunities and challenges that would benefit from the fusion of the techniques and approaches that hitherto have been routinely employed by the information theory, wireless communication, operations research, networking and control theory communities. More specifically, we concluded that the research challenges described in Sections VIII-A - VIII-C are currently not fully addressed, and constitute significant opportunities for future work that would also lead to methods for tackling problems specified by more realistic models and assumptions. Subsequently, in Sections VIII-D - VIII-G we proceed with suggesting additional future research directions that broach aspects of security and secrecy, effective methods to cope with systems comprising multiple UDCs, UDC in learning and more realistic battery models, respectively.

A. Noisy channels for remote estimation

Most work discussed in Section VII presumes that, in the absence of an outage, an EH packet-drop link can convey a real vector unerringly from the sensor to the estimator when a transmission is requested. Considering the unidealized case in which a noisy channel links the sensor to the estimator would require the investigation of causal encoding and decoding schemes possibly inspired on modifications of those discussed in Section IV introducing channel encoding and decoding, and possibly lossy source compression, as was done in [119] for an independent Gaussian source would expand the set of policies to include high and low fidelity solutions whose implemention may consume more or less energy [120], respectively, in addition to that required for transmission. Obtaining methods for the design of such policies with stability and performance guarantees is, therefore, an important open challenge.

The case in which the UDC would depend not only on the energy available but also on the state of a physical system, such as the position and velocity vectors of a mobile agent, would be an interesting extension of this framework. In this setting, the UDC could be a communication channel between the agent and a base station whose outage likelihood would increase with distance for each transmission power level. The scenario in which the UDC would be a global positioning module (GPS) whose accuracy would depend on the location and power level, with higher fidelity consuming more power, would be an example relevant to autonomous navigation [121] of unmanned assets. In these cases, one needs to consider policies that not only allocate power for the UDC but also govern the control action that steers the agent. As is discussed in [122], many active sensing problems could be formulated similarly once energy harvesting constraints are included.

B. Queueing, remote estimation and age of information

According to the optimality principle used in [99], for the framework adopted in Section VII-C if a sensor has access to the state $\hat{V}$ or is able to compute the optimal state estimate $\hat{V}$ - cases we refer to as full-information sensor or smart sensor, respectively - then it should always attempt to transmit the latest one to the remote estimator. Hence, given a choice, it is optimal to discard state estimates corresponding to failed transmission attempts in favor of the most recent one - a principle we denote as most-recent-only optimality. In fact, this most-recent-only optimality principle for a full-information/smart sensor remains valid even in the controlled transmission setting described in Section VII-B. Hence, these observations suggest that introducing a packet management layer, such as establishing a queue, prior to transmission is not necessary and may even be counterproductive when the sensor is full-state/smart.

However, when using an existing transmission system one may be left with no option but to deal with a pre-existing first-in-first-out queue-based non-preemptive management system in which a packet leaves the queue only when it is successfully conveyed to the remote estimator. Notably, as is proved in [123], [124] for the aforementioned scenario, for the case in which the source is a Wiener or Ornstein-Uhlenbeck process and the sensor is full-state, it is never optimal to submit a measurement for transmission when the queue is non-empty, and when a new measurement is inserted in the empty queue for transmission it must be the current state of the process, which can be viewed as a version of the most-recent-only optimality principle for the case when pre-emption is not allowed. Interestingly, the optimal rule proposed in [124] to determine whether to submit the latest measurement for transmission, subject to the queue being empty, follows an event-based threshold policy that is analogous to the one found to be optimal for the closely-related case analyzed in [125]. The fact that the most-recent-only optimality principle may no longer hold when the sensor is neither full-state nor smart [114] raises the question of whether, if the sensor in the framework of [124] could transmit only noisy output measurements $V^\beta$, there would be optimal policies for which a transmission would be scheduled even when the queue is non-empty. Furthermore, if the queue is served by a channel powered by energy harvested from stochastic sources then we are left with the currently unsolved problem of designing policies that determine not only when and which estimates or measurements should be placed for transmission it must be the current state of the process, which can be viewed as a version of the most-recent-only optimality principle for the case when pre-emption is not allowed. A typical approach would be to characterize stabilizing policies first, perhaps within an appropriately parametrized class, followed by the characterization of structural properties that could facilitate the computation of optimal policies using tractable methods. The stability problem may require the integration of techniques such as the ones used in [52] and [118], which were discussed in Sections VI and VII in the context of queueing and remote estimation, respectively. Devising methods to design optimal policies may involve fusing the techniques adopted in Section VII-B and [123], [124], and possibly leveraging the fact that our UDC model is amenable to existing methodologies [116], [111] for partially observable controlled Markov chains. Since the fidelity of the estimate constructed at the remote estimator depends on the recency of the information received by the remote estimator, both the stability and the optimization problems are related to

The techniques used in [123] are analogous to the ones adopted for the case without packet drops in [105].
recent work seeking to analyze and design data-transmission systems that effectively regulate the age of information \[126, 127\]. In fact, it has been suggested in \[128, 129\] that the remote estimation and age of information problems are inextricably tied.

C. Feasible region and trade-off among performance metrics

Most existing studies in which queue length, utilization or workload affects servers, including those mentioned in Section VII, examine the effects on a single aspect of server performance, oftentimes their service rates being the choice. In another example, the study by Chatterjee et al. \[85\] takes into account the service quality (which is modeled as channel condition in their study) as a function of queue length and examines the information-theoretic capacity of such systems.

In many cases of interest, however, including human supervisors \[79\], several performance aspects, including service rate and service quality (e.g., reliability or frequency of mistakes or poor decisions), can be affected at the same time by work history via server state. Moreover, the requirements (e.g., service rate vs. reliability) in different applications are likely to vary considerably based on the types of tasks that need to be processed.

From this viewpoint, it is important to develop a comprehensive theory for these systems, including their fundamental limits. Regrettably, to the best of our knowledge, little is known about the feasible region of multiple performance metrics which can be achieved simultaneously and how to design suitable policies for carrying out a desired trade-off among various performance metrics in the feasible region, in particular on the Pareto frontier.

D. Secure remote estimation powered by energy-harvesting

Preventing, or at the very least mitigating the effect of, attacks on the channels connecting the sensors to every component relying on remotely constructed state estimates is critical to ensure the safe operation \[130\] of networked cyber-physical systems. While clever encoding and decoding schemes \[131\], some of which may be implemented efficiently using event-based algorithms, may thwart or curb the effect \[132\] of certain types of attacks, a relentless surreptitious Man-in-the-Middle (MitM) attack injecting false data \[133\] may significantly degrade the performance of any remote estimation system. The case-study in \[134\] illustrates that by employing message authentication codes (MACs), even if infrequently, may afford performance guarantees against MitM attacks. It further demonstrates that although MAC are known to substantially increase communication overhead, which is particularly critical when using bandwidth-limited networks such as the ones found in automobiles, its parsimonious use may suffice for practical purposes. A promising new research avenue is to investigate estimation-oriented encoding and decoding schemes and MAC scheduling policies that would jointly provide stability and performance guarantees, or would even be jointly optimal with respect to a given estimation error metric, in the presence of MitM attacks. Realistic problem formulations, in which information transmission is powered by an energy harvesting module, would have to account for the additional energy required for the transmission of MAC. A new type of EH link \[3\] that would account not only for packet-drop events, but also MitM attacks whose likelihood and severity would depend on the power employed in each transmission for the inclusion of MAC could be a useful abstraction to design and evaluate the performance of such systems. The open problems discussed here would also be relevant for distributed function calculation \[135\] in the cases in which information would be wirelessly disseminated among the agents via such security-threatened EH links.

Finally, it would be important to investigate all of these problems in light of other security threats \[136\], including denial-of-service attacks \[137\].

E. Coping with multiple UDCs

In many situations of practical interest, there are a set of servers working on tasks (e.g., emergency rooms at hospitals). Furthermore, the availability of servers may be affected by some exogenous processes (e.g., schedules of doctors and nurses at hospitals). For example, data centers comprise a large number of server racks that are connected by high-speed networks and are sometimes subject to power constraints. Also, because the reliability of hardware components, such as CPUs, GPUs and memory modules, degrades when the temperature exceeds some threshold, they need to be cooled for stable operation, for instance, via direct-to-chip liquid cooling. Moreover, because new server racks are added over time to meet increasing demands and old or failed racks are replaced at different times, the computational capabilities offered by various computational resources, which are designed for different types of tasks (e.g., CPUs vs. GPUs), can vary significantly.

Another class of problems well suited for the UDC framework with multiple UDCs, which is also related to those in Sections VIII-B and VIII-D is information collection from multiple sources over time. These sources may be distributed sensors in wireless sensor networks (WSNs), which are powered by renewable energy, or “friends” in social networks who prefer not to be bothered constantly for the latest information. One can view the “usefulness” of the information collected from each sensor or friend as the reward. Such usefulness of information from a sensor or a friend will likely be stochastic. However, there are certain factors that would affect the usefulness of the information. These include (i) the accuracy or quality of the sensors or the importance of the friends in social networks (which are often measured using their “centrality” in social networks \[138\]) and (ii) the age-of-information from each sensor or friend introduced in Section VIII-B as well as the frequency of information requests.

Unfortunately, the quality of sensors and the importance of friends may not be known in advance. In addition, in many practical scenarios, we may be able to poll or collect information from only a limited number of sensors or friends at any given time and only so often. In WSNs, for instance, the number of available channels or timeslots in a frame may
constrain the number of measurements we can collect at each time and, when the sensors are powered by renewable energy, they may not be able to report measurements even when they are polled, as their availability for reporting measurements will be governed by a stochastic process.

Despite their prevalence, not much is known about their fundamental performance limits and efficient resource management in such systems. Only recently research has demonstrated the benefits of task-aware scheduling at data centers (e.g., [139]). Consequently, there is a rich set of open problems in related domains. For instance, consider a system in which human servers are employed for processing tasks of different types, which arrive at fixed rates (e.g., assembly lines at manufacturing facilities). One interesting open question is how one should schedule tasks so that the long-term fraction of time the human servers are required to work is minimized, in order to reduce the fatigue or operational costs (e.g., wages for employees and costs of keeping the facilities running). Similarly, when servers are heterogeneous and designed/optimized for different types of tasks, how should we schedule arriving tasks so that the system remains stable and the average service times of the tasks are minimized subject to utilization-based constraints similar to those discussed in the previous sections? These are some of questions, the answer to which can have significant impact on many areas, including crucial applications involving HSC (e.g., air traffic control and nuclear power plant monitoring). A useful approach for studying these problems, especially when some of the parameters are unknown, is the restless multi-armed bandit model, which has been previously applied to stochastic scheduling [140], [141].

**F. UDC in learning**

Recent years have highlighted several exciting problems at the intersection of machine learning and control theory. The dual component structure of UDC models is particularly relevant in adversarial machine learning, as touched upon in Section V-B. We mention here a few of other areas where UDC-based models are likely to have impact.

Generative adversarial networks (GANs) have been successfully applied to text, image, or video generation, drug discovery, and image-to-image synthesis [142], [143], [144]. A GAN has two components: a discriminator and a generator. The generator component produces finer and finer approximations of a data distribution of interest, whereas the discriminator component distinguishes samples from the true data distribution and the generator’s output. Training a GAN may be thought of as a two-person zero-sum game between the generator and the discriminator. Since SGD is used for updating parameters of the generator and discriminator, the training process for both components may be thought of as Markov processes. However, the generator’s update relies on the discriminator’s current state, and the discriminator’s update relies on the generator’s current state. Thus, the performance processes for both these components consist of their corresponding states at a certain time \( k \), and these two components drive each other’s state. Although such a model is not described using the UDC framework from this paper, a variation where the action kernel is allowed to have its own internal state can model the training process for a GAN.

There has been a lot of recent work at the intersection of reinforcement learning and robust control [145], [146]. Learning optimal policies for applications such as self-driving vehicles requires a large amount of data, and it is often difficult or expensive to obtain such data in large quantities. For this reason, reinforcement learning algorithms are often trained on simulated data. Since the algorithm is tested in conditions that may differ significantly from what it was trained on, an important new tool has been to incorporate robust control techniques in the training process. There are several interacting components in this system: the learning algorithm that responds to the environment, a model of the environment with uncertainty quantification, and also adversaries that may change the environment in adversarially optimal ways to derail the learning algorithm [147], [148]. As shown in Section V-B, a UDC-based model may easily describe the algorithm and the adversary. An interesting addition is the component that models the environment and quantifies the corresponding uncertainty. This component, which is also a Markov process, also acts as the controller for the learning algorithm.

**G. More realistic battery models for energy harvesting: leakage and nonlinearities**

Although, as we discussed in Section III-B, the batteries used in energy harvesting modules have a rather complex behavior, the existing work discussed throughout this article adopts either the linear-saturated or the finite-state approximations. These simplified models do not capture a host of issues that could possibly require new methods and abstractions. This is illustrated by the following two features that could be captured by our general model of Definition 5.

a) **Leakage:** The chemistry of every battery and the operation of its auxiliary circuitry will cause charge to leak, even when it is not supplying power. Hence, the charge that is stored in a battery may be partially lost unless it is used quickly or the leakage is offset by harvesting, which raises the issue of age of energy. This is a relevant problem for low-power remote sensing devices that operate over long periods of time.

b) **Nonlinearities:** In Section III-B we mentioned the fact that, due to the discharge curve, in general there is a state of charge threshold below which the voltage of the battery does not suffice to power the other components. Consequently, if the voltage is near the required minimum then leakage effects may drain the state of charge below the aforesaid threshold, after which enough energy must be harvested before the battery can function again. The fact that the state of charge also governs the portion of the energy harvested that is effectively stored constitutes another important nonlinearity. Notably, as the state of charge nears its maximum and minimum the ability of the battery to store energy varies considerably.

**Acknowledgement**

The authors would like to thank Sennur Ulukus (UMD), Yasser Shoukry (UMD) and Vijay Gupta (UND) for helpful
discussions and suggestions.

REFERENCES

[1] S. Ulukus, A. Yener, E. Erkip, O. Simeone, M. Zorzi, P. Grover, and K. Huang, “Energy harvesting wireless communications: a review of recent advances,” IEEE Journal on Selected Areas in Communications, vol. 33, pp. 360–381, March 2015.

[2] K.-D. Kim and P. R. Kumar, “Cyber-physical systems: a perspective at the centennial,” Proceedings of the IEEE, pp. 1287–1308, May 2012.

[3] R. Baheti and H. Gill, The impact of control technology, ch. Cyber-physical systems, pp. 161–166. IEEE Control Systems Society, 2011.

[4] M. A. Staal, “Stress, cognition and human performance: A literature review and conceptual framework,” Tech. Rep. NASA/TM-2004-212824, NASA, August 2004.

[5] A. S. Leong, D. E. Quevedo, and S. Dey, Optimal control of energy resources for state estimation over wireless channels. Briefs in electrical and computer engineering, Springer, 2018.

[6] J. P. Hespansha, P. Naghshtabrizi, and Y. Xu, “A survey of recent results in networked control systems,” Proceedings of the IEEE, vol. 95, no. 1, pp. 138–162, 2007.

[7] K. H. Teigen, “Yerkes-Dodson: a law for all seasons,” Theory & Psychology, vol. 4, no. 4, pp. 525–547, 1994.

[8] C. D. Wickens and J. G. Hollands, Engineering psychology and human performance. Prentice Hall, third edition ed., 2000.

[9] M. L. Cummings and C. E. Nehme, “Modeling the impact of workload in network control supervisory control settings,” in Proceedings of the 2nd Annual Sustaining Performance Under Stress Symposium, February 2009.

[10] J. R. Peters, V. Srivastava, G. S. Taylor, A. Surana, M. P. Eckstein, and F. Bullo, “Human supervisory control of robotic teams: Integrating cognitive modeling with engineering design,” IEEE Control Systems Magazine, vol. 35, pp. 57–80, December 2015.

[11] H. P. G. van Ooijen and J. W. M. Bertrand, “The effects of a simple arrival rate control policy on throughput and work-in-process in production systems with workload dependent processing rates,” International Journal of Production Economics, vol. 85, pp. 61–68, 2003.

[12] K. Savla and E. Frazzoli, “A dynamical queue approach to intelligent task management for human operators,” Proceedings of the IEEE, vol. 100, pp. 672–686, March 2012.

[13] T. Koch, A. Lapidoth, and P. P. Sotiriadis, “Channels that heat up,” IEEE Transactions on Information Theory, vol. 55, pp. 3594–3612, August 2009.

[14] A. Baknina, O. Ozel, and S. Ulukus, “Energy harvesting communications under explicit and implicit temperature constraints,” IEEE Transactions on Wireless Communications, vol. 17, pp. 6680–6692, October 2018.

[15] D. Forte and A. Srivastava, “Thermal-aware sensor scheduling for distributed estimation,” ACM Transactions on Sensor Networks, vol. 9, pp. 53:1–53:31, July 2013.

[16] F. Baccelli and P. Brémaud, Elements of queueing theory. Springer, second edition ed., 2003.

[17] S. Sudevalayam and P. Kulkarni, “Energy harvesting sensor nodes: Survey and implications,” IEEE Communications Surveys & Tutorials, vol. 13, no. 3, pp. 443–461, 2011.

[18] D. Gunduz, K. Stamatou, N. Michelusi, and M. Zorzi, “Designing intelligent energy harvesting communication systems,” IEEE communications magazine, vol. 52, no. 1, pp. 210–216, 2014.

[19] S. Ulukus, A. Yener, E. Erkip, O. Simeone, M. Zorzi, P. Grover, and K. Huang, “Energy harvesting wireless communications: A review of recent advances,” IEEE Journal on Selected Areas in Communications, vol. 33, no. 3, pp. 360–381, 2015.

[20] O. Ozel and S. Ulukus, “Achieving AWGN capacity under stochastic energy harvesting,” IEEE Transactions on Information Theory, vol. 58, no. 10, pp. 6471–6483, 2012.

[21] O. Ozel and S. Ulukus, “AWGN channel under time-varying amplitude constraints with causal information at the transmitter,” in 2011 Conference Record of the Forty Fifth Asilomar Conference on Signals, Systems and Computers (ASILOMAR), pp. 373–377, IEEE, 2011.

[22] Y. Dong and A. Ozgür, “Approximate capacity of energy harvesting communication with finite battery,” in 2014 IEEE International Symposium on Information Theory, pp. 801–805, IEEE, 2014.

[23] V. Jog and V. Anantharam, “An energy harvesting AWGN channel with a finite battery,” in 2014 IEEE International Symposium on Information Theory, pp. 806–810, IEEE, 2014.

[24] K. Tutuncuoglu, O. Ozel, A. Yener, and S. Ulukus, “Improved capacity bounds for the binary energy harvesting channel,” in 2014 IEEE International Symposium on Information Theory, pp. 976–980, IEEE, 2014.

[25] Y. Dong, F. Farnia, and A. Ozgür, “Near optimal energy control and approximate capacity of energy harvesting communication,” IEEE Journal on Selected Areas in Communications, vol. 33, no. 3, pp. 540–557, 2015.

[26] O. Ozel, K. Tutuncuoglu, S. Ulukus, and A. Yener, “Fundamental limits of energy harvesting communications,” IEEE Communications Magazine, vol. 53, no. 4, pp. 126–132, 2015.

[27] D. Shaviv, P. Nguyen, and A. Ozgür, “Capacity of the energy-harvesting channel with a finite battery,” IEEE Transactions on Information Theory, vol. 62, no. 11, pp. 6436–6458, 2016.

[28] O. Ozel and S. Ulukus, “On the capacity region of the Gaussian MAC with batteryless energy harvesting transmitters,” in 2012 IEEE Global Communications Conference (GLOBECOM), pp. 2385–2390, IEEE, 2012.

[29] H. Inan, D. Shaviv, and A. Ozgür, “Capacity of the energy harvesting Gaussian MAC,” IEEE Transactions on Information Theory, vol. 64, no. 4, pp. 2347–2360, 2018.

[30] O. Ozel, J. Yang, and S. Ulukus, “Optimal scheduling over fading broadcast channels with an energy harvesting transmitter,” in 2014 44th IEEE International Conference on Communications (ICC), June 2014.

[31] O. Ozel, J. Yang, and S. Ulukus, “Optimal broadcast scheduling for an energy harvesting rechargeable transmitter with a finite capacity battery,” IEEE Transactions on Wireless Communications, vol. 11, no. 6, pp. 2193–2203, 2012.

[32] K. Tutuncuoglu and A. Yener, “Sum-rate optimal power policies for energy harvesting transmitters in an interference channel,” Journal of Communications and Networks, vol. 14, no. 2, pp. 151–161, 2012.

[33] J. Yang and S. Ulukus, “Optimal packet scheduling in a multiple access channel with energy harvesting transmitters,” Journal of Communications and Networks, vol. 14, no. 2, pp. 140–150, 2012.

[34] D. Shaviv and A. Ozgür, “Universally near optimal online power control for energy harvesting nodes,” IEEE Journal on Selected Areas in Communications, vol. 34, no. 12, pp. 3620–3631, 2016.

[35] A. Baknina and S. Ulukus, “Optimal and near-optimal online strategies for energy harvesting broadcast channels,” IEEE Journal on Selected Areas in Communications, vol. 34, no. 12, pp. 3696–3708, 2016.

[36] A. Baknina and S. Ulukus, “Energy harvesting multiple access channels: Optimal and near-optimal online policies,” IEEE Transactions on Communications, vol. 66, no. 7, pp. 2904–2917, 2018.

[37] C. E. Shannon, “A mathematical theory of communication,” Bell Sys. Tech. J., vol. 27, pp. 379–423, 1948.

[38] J. G. Smith, “The information capacity of amplitude-and variance-constrained scalar Gaussian channels,” Information and Control, vol. 18, no. 3, pp. 203–219, 1971.

[39] D. Shaviv, A. Ozgür, and H. Permuter, “Can feedback increase the capacity of the energy harvesting channel?” in IEEE Information Theory Workshop (ITW), 2015, pp. 1–5, IEEE, 2015.

[40] D. Blackwell, L. Breiman, and A. J. Thomasian, “The capacities of certain channel classes under random coding,” in 1948.

[41] J. Csiszar and J. Körner, Information theory: coding theorems for discrete memoryless systems. Cambridge University Press, 2011.
[147] L. Pinto, J. Davidson, R. Sukthankar, and A. Gupta, “Robust adversarial reinforcement learning,” in Proceedings of the 34th International Conference on Machine Learning-Volume 70, pp. 2817–2826, JMLR.org, 2017.

[148] J. Fu, K. Luo, and S. Levine, “Learning robust rewards with adversarial inverse reinforcement learning,” arXiv preprint arXiv:1710.11248, 2017.