A Dynamical Queue Approach to Intelligent Task Management for Human Operators

Ketan Savla Emilio Frazzoli

Abstract

Formal methods for task management for human operators are gathering increasing attention to improve efficiency of human-in-the-loop systems. In this paper, we consider a novel dynamical queue approach to intelligent task management for human operators. We consider a model of a dynamical queue, where the service time depends on the server utilization history. The proposed queueing model is motivated by, but not restricted to, widely accepted empirical laws describing human performance as a function of mental arousal. The focus of the paper is to characterize the throughput of the dynamical queue and design corresponding maximally stabilizing task release control policies, assuming deterministic arrivals. We focus extensively on threshold policies that release a task to the server only when the server state is less than a certain threshold. When every task brings in the same deterministic amount of work, we give an exact characterization of the throughput and show that an appropriate threshold policy is maximally stabilizing. The technical approach exploits the optimality of the one-task equilibria class associated with the server dynamics. When the amount of work associated with the tasks is an i.i.d. random variable with finite support, we show that the maximum throughput increases in comparison to the case where the tasks have deterministic amount of work. Finally, we provide preliminary empirical evidence in support of the applicability of the proposed approach to systems with human operators.

I. Introduction

Recent years have witnessed great technological advancements in automation, which in turn have marginalized the role of humans in many engineering applications. Nevertheless, the role of humans

This is revised version of the original draft which was submitted to the *Proceedings of the IEEE* in June 2011.

K. Savla is with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139 (ksavla@mit.edu).

E. Frazzoli is with the Laboratory for Information and Decision Systems, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA 02139 (frazzoli@mit.edu).

for critical tasks remains indispensable. With scientific and technological advances in modeling human performance, there has been an increasing interest in formal methods for task management for human operators to increase the overall efficiency of human-in-the-loop systems.

In this paper, we consider applications where human operators have to persistently perform similar tasks, generated over time by some external process. Typical examples for such settings include remotely located human operators processing continuous stream of information from unmanned vehicles in a persistent surveillance mission, e.g., see [1], or workers processing jobs in a production line. We consider a queueing framework for such settings. Queueing theory is a framework to study systems with waiting lines, and it is used to model several scenarios in commerce, industry, health-care, public service and engineering domains. An extensive treatment of queueing systems can be found in several texts, e.g., see [2], [3].

Queueing methods for task management in the context of call centers and job floors have attracted a great deal of attention, e.g., see [4]. A typical feature of such *static* queue models is that, as long as the tasks are independent of each other, the performance of the operator on the tasks are also independent. However, it is reasonable to expect that, even if the tasks are independent of each other, the performance of the *human* server on those tasks could be correlated. For example, the cumulative performance of a human operator on two difficult tasks serviced one after the another would be different if the tasks are assigned to the operator with a break in between, as compared to assigning them without any break in between.

In this paper, we consider a novel model for a dynamical queue introduced in our earlier work [5], in which service times depend on the utilization history of the server. In other words, we model the server as a dynamical system, and let the service time be a function of the server state. Given this model, we consider the case in which new tasks arrive at a deterministic rate, and propose a task release control architecture that schedules the beginning of service of each task after its arrival. The model for state-dependent service times is inspired by, but not restricted to, a well known empirical law from psychology—the Yerkes-Dodson law [6]—which states that human performance increases with mental arousal up to a point, and decreases thereafter. Our model in this paper is in the same spirit as the one in [7], [8], where the authors consider a state-dependent queueing system whose service rate is first increasing and then decreasing as a function of the amount of outstanding work. However, our model differs in the sense that the service times are related to the utilization history rather than the outstanding amount of work. A similar model has also been reported in the human factors literature, e.g., see [9]. Recently, there has also been interest in incorporating error rates into the performance metric for humans in a queueing setup, e.g., see [10], [11].

The control architecture considered in this paper falls under the category of *task release control*, which has been typically used in production planning to control the release of jobs to a production system to deal with machine failures, input fluctuations, and variations in operator workload (see, e.g., [12], [13]). The task release control architecture is different from an *admission control* architecture, e.g., see [14], [8], where the objective is, given a measure of the *quality of service*, to determine criteria on the basis of which to accept or reject incoming tasks. In the setting of this paper, no task is dropped and the task release controller simply acts like a switch regulating access to the server and hence effectively determines the schedule for the beginning of service of each task after its arrival. We extensively focus on threshold based task release control policies that release task to the server only if the server state is below a certain fixed value.

It is informative to compare our approach with other approaches to task management for human operators. One such approach is *interface design* which has attracted considerable research effort, mainly from the human factors community, e.g., see [15], [16]. The idea behind such an approach is to give the operator options to work on a subset of outstanding tasks by displaying relevant information and then using cues, e.g., colors, to draw attention to most relevant tasks. A primary motivation for displaying such rich set of information to the operator is to maintain his/her *situational awareness* of the mission that could potentially improve operator performance. While we do not address this feature explicitly, the queueing based framework in this paper could be used in tandem with interface design approaches to control the amount of information displayed so that the operator does not get overwhelmed.

While this paper discusses the use of dynamical queues and task release control policies for human-in-the-loop systems, such a framework is finding increasing application in a variety of other domains, where the queue parameters are strongly dependent on some internal state. Examples include ramp metering congestion control of motorways, e.g., see [17], and air traffic control of national airspace systems, e.g., see [18].

The contributions of the paper are as follows. First, we present a novel dynamical queue framework for task management of a human operator engaged in a persistent mission, where the server characteristics are inspired by, but not restricted to, empirical laws relating human performance to utilization history. We then exactly characterize the throughput of the dynamical queue for the special case when all the tasks are homogeneous and show that the task release control policy that releases tasks to the server only when the server state is below an appropriately chosen threshold value is maximally stabilizing. The technical approach relies on characterizing a specific equilibrium class of the underlying dynamical system and exploiting its optimality. For the heterogeneous task case, we provide bounds on the throughput of the dynamical queue, where we prove a surprising result, i.e., that the throughput of the queue strictly increases with the introduction of heterogeneity. Finally, we provide preliminary empirical evidence to justify the dynamical queue framework for human operators.

The rest of the paper is organized as follows. In Section II, we describe the model for dynamical queueing systems, task release control policies and derive trivial bounds on the throughput. In Section III, we consider the special case when all the tasks are homogeneous, i.e., they bring the same deterministic amount of work. In Section IV, we consider the general case when the tasks bring heterogeneous amounts of work. In Section V, we present preliminary experimental evidence aimed at validating the dynamical queue framework for human operators. Finally, in Section VI, we give concluding remarks and describe some future work directions.

II. Dynamical Queue Model

Consider the following single-server queue model. Tasks arrive periodically, at rate λ, i.e., a new task arrives every $1/\lambda$ time units. The tasks are identical and independent of each other and each task brings w units of work, where w is an i.i.d. random variable whose probability distribution is f_W with bounded support $[\mathcal{W}_1, \mathcal{W}_2]$ for some $\mathcal{W}_1 > 0$ and $\mathcal{W}_2 \geq \mathcal{W}_1$. In the rest of the paper, we shall assume this bounded support assumption on f_W without explicitly repeating it. Let $\bar{w}$ be the mean of w with respect to f_W. Let $\delta_{\bar{w}}$ be the Dirac delta distribution centered at $\bar{w}$. We shall use the δ distribution for the scenario when the tasks are homogeneous. Note that the task arrival process under consideration is deterministic. We briefly discuss the implications of stochastic inter-arrival times in Section VI. The tasks must be serviced in the order of their arrival. We next state the dynamical model for the server, which specifies the state-dependent service times for the server.

A. Server Model

Let $x(t) \in [0, 1]$ be the server state at time t, and let $b : \mathbb{R} \to \{0, 1\}$ be such that $b(t)$ is 1 if the server is busy at time t, and 0 otherwise, where $\mathbb{R}$ is the set of real numbers. The evolution of $x(t)$ is governed by a simple first-order model:

$$\dot{x}(t) = \frac{b(t) - x(t)}{\tau}, \qquad x(0) = x_0, \tag{1}$$

where $\tau > 0$ is a time constant that determines the extent to which past utilization affects the current state of the server, and $x_0 \in [0, 1]$ is the initial condition. The quantity $x(t)$ denotes the *utilization ratio* of the server, i.e., the fraction of the recent history when the server was busy. Physically, $x(t)$ represents the perceived workload of the operator based on its recent utilization history. Equation (1) can be considered to be the continuum limit of the discrete-time exponential moving window average by re-writing the time derivative in Equation (1) from first principles:

$$x(t+\triangle t) \approx \left(1 - \frac{\triangle t}{\tau}\right) x(t) + \frac{\triangle t}{\tau} b(t).$$

Equation (1) is closely related to the moving window average model with time window τ. The simple moving window average model has been proposed in [9] for computing the utilization ratio. For other models of human mental workload, we refer the reader to [19]. The time constant τ corresponds to the inverse of the sensitivity of the operator to its recent utilization history: higher τ correspond to smaller sensitivity and lower τ correspond to higher sensitivity. Note that the set $[0, 1]$ is invariant under the dynamics in Equation (1) for any $\tau > 0$ and any $b : \mathbb{R} \to \{0, 1\}$.

The service times are related to the state $x(t)$ through a map $\mathcal{S} : [0, 1] \to \mathbb{R}_{>0}$, where $\mathbb{R}_{>0}$ is the set of positive real numbers. If a task is allocated to the server at state x, then the amount of time required to perform unit work is given by $\mathcal{S}(x)$. Therefore, if the amount of work associated with a task allocated to the server at state x is w, then the service time on that task is $w\mathcal{S}(x)$. This linear decomposition of the total service time into the amount of work associated with the task and the rate of performing work with respect to the initial server state is an approximation to a more realistic scenario where the rate of performing work also depends on the amount of work – such a model has been proposed in [8]. The linear decomposition that we use is reasonable especially for small heterogeneity in the tasks. In our framework, the controller cannot interfere with the server while it is servicing a task. Hence, the only way in which the server state can be controlled is by scheduling the beginning of service of tasks after their arrival. Such controllers are called task-release controllers and will be formally characterized later on. In this paper we assume that $\mathcal{S}(x)$ is positive valued, continuous, and convex. Let $\mathcal{S}_{\text{min}} := \min\{\mathcal{S}(x) \mid x \in [0, 1]\}$, and $\mathcal{S}_{\text{max}} := \max\{\mathcal{S}(0), \mathcal{S}(1)\}$.

The solution to Equation (1) is $x(t) = e^{-t/\tau}\left(\int_0^t \frac{1}{\tau} b(s) e^{s/\tau} ds + x_0\right)$. This implies that the server state $x(t)$ is increasing when the server is busy, i.e, when $b(t) = 1$, and decreasing when the server is not busy, i.e., when $b(t) = 0$. Note that $\mathcal{S}(x)$ is not necessarily monotonically increasing in x, since it has been noted in the human factors literature (e.g., see [6]) that, for certain cognitive tasks demanding persistence, the performance (which in our case would correspond to the inverse of $\mathcal{S}(x)$) could *increase* with the state x when x is small. This is mainly because a certain minimum level of mental arousal is required for good performance. A well-known empirical law capturing such characteristics is the Yerkes-Dodson law [6]. A loose experimental justification of this server model in the context of human-in-the-loop systems is

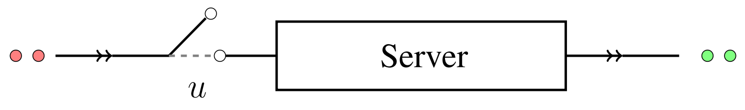


Fig. 1. Task release control architecture. The task release controller u has real-time information about the state of the server and accordingly acts like switch to regulate the allocation of tasks to the server. An outstanding task is allocated to the server if and only if the server is idle and u is in the ON state.

included in our earlier work [20], where $\mathcal{S}(x)$ for that setup was found to have a U-shaped profile. We shall use that particular $\mathcal{S}(x)$ from [20] for various numerical illustrations in this paper. We provide further experimental evidence for this model in Section V. It is important to note that the U-shaped relationship between the service time and the operator's utilization, as would be dictated, for example, by the Yerkes-Dodson law, falls within our assumptions on $\mathcal{S}(x)$, but it is not essential. In particular, our assumptions on $\mathcal{S}(x)$ also allow it to be monotonically increasing, decreasing, or even constant over $x \in [0, 1]$.

B. *Task Release Control Policy*

We now describe task release control policies for the dynamical queue. Without explicitly specifying its domain, a task release controller u acts like an on-off switch at the entrance of the queue, e.g., see Figure 1. In short, u is a task release control policy if $u(t) \in \{\text{ON}, \text{OFF}\}$ for all $t \geq 0$, and an outstanding task is assigned to the server if and only if the server is idle, i.e., when it is not servicing a task and $u = \text{ON}$. Let $\mathcal{U}$ be the set of all such task release control policies. Note that we allow $\mathcal{U}$ to be quite general in the sense that it includes control policies that are functions of λ, $\mathcal{S}$, $x(t)$, f_W, τ, etc.

C. *Example*

We present a simple example to illustrate the dynamical queue setup and the task release control architecture. Please refer to Figure 2 for a graphical illustration of this example. Let $\lambda = 0.02\, s^{-1}$, $n(0) = 2$, $x(0) = 0.6$, $\tau = 100\, s$, $\mathcal{S}(x) = 375x^2 - 375x + 150$ and $f_W = \delta_1$. Consider the task release control policy that is ON if and only if $x(t) \leq 0.7$ and OFF otherwise. Since $x(0) = 0.6 < 0.7$, $u(0) = \text{ON}$. Also, since $n(0) > 0$, a task is allocated to the server at $t = 0$. The time required for the server to service this task is $\mathcal{S}(0.6) = 60\, s$. The server state at the end of the service, as governed by Equation (1) is $x(60) = 0.7805$. Therefore, $u(60) = \text{OFF}$. Since the inter-task arrival time is $50\, s$, we have that $n(60) = 2 > 0$. No new task is allocated to the server at $t = 60\, s$ and it starts idling until its state reaches $x = 0.7$ at which point $u = \text{ON}$ and hence a new task is allocated to the server and the server state starts increasing again.

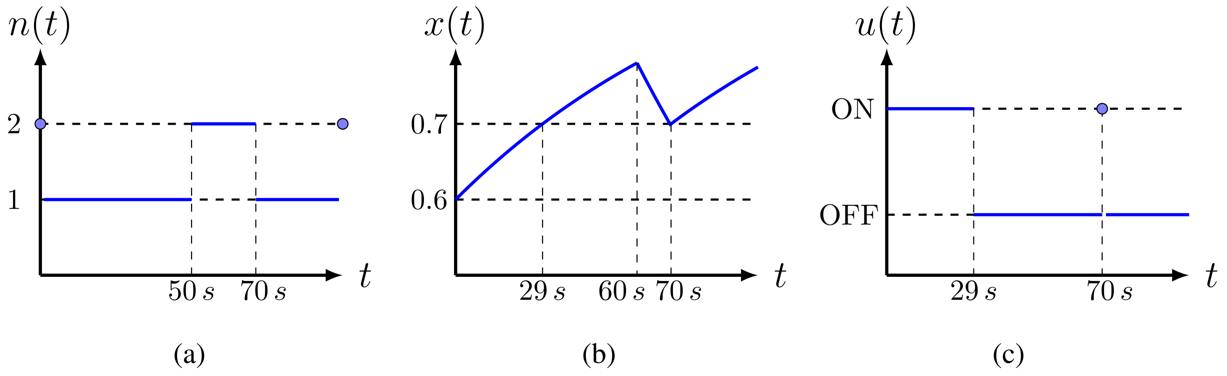


Fig. 2. Illustration of the evolution of a typical dynamical queue under the task release control architecture: (a) evolution of queue length $n(t)$, (b): evolution of the server state $x(t)$, and (c): evolution of the value of the control policy $u(t)$. The specifications of the dynamical queue used here are: $\lambda = 0.02\,s^{-1}$, $n(0) = 2$, $x(0) = 0.6$, $\tau = 100\,s$ and $\mathcal{S}(x) = 375x^2 - 375x + 150$, $f_W = \delta_1$. The task release control policy is such that it is ON if and only if $x(t) \leq 0.7$ and OFF otherwise.

D. *Objectives of the paper*

We now formally state the problem. For a given $\tau > 0$ and f_W, let $n_u(t, \tau, \lambda, f_W, x_0, n_0)$ be the queue length, i.e., the number of outstanding tasks, at time t, under task release control policy $u \in \mathcal{U}$, when the task arrival rate is λ and the server state and the queue length at time $t = 0$ are x_0 and n_0, respectively. Define the maximum stabilizable arrival rate for policy u as:

$$\lambda_{\text{max}}(\tau, f_W, u) := \sup\left\{\lambda \colon \limsup_{t\to+\infty} n_u(t, \tau, \lambda, f_W, x_0, n_0) < +\infty \quad \forall x_0 \in [0,1],\, n_0 \in \mathbb{N} \quad \text{a.s.}\right\}.$$

The quantity $\lambda_{\text{max}}(\tau, f_W, u)$ will also be referred to as the throughput under policy u. The maximum stabilizable arrival rate over all policies, or simply the throughput, is defined as $\lambda^*_{\text{max}}(\tau, f_W) := \sup_{u\in\mathcal{U}} \lambda_{\text{max}}(\tau, f_W, u)$. For a given $\tau > 0$ and f_W, a task release control policy u is called *maximally stabilizing* if, for any $x_0 \in [0,1]$, $n_0 \in \mathbb{N}$, $\limsup_{t\to+\infty} n_u(t, \tau, \lambda, f_W, x_0, n_0) < +\infty$ for all $\lambda \leq \lambda^*_{\text{max}}(\tau, f_W)$ almost surely. The primary objective in this paper is to compute the throughput and design a corresponding maximally stabilizing task release control policy for the dynamical queue whose server state evolves according to Equation (1), and where $\mathcal{S}(x)$ is positive, continuous, and convex.

In this paper, we extensively focus on a specific class of task release control policies – threshold policies. For a given $x^* \in [0,1]$, the x^*-threshold policy is defined as

$$u_{x^*}(t) = \begin{cases} \text{ON} & \text{if } x(t) \leq x^*, \\ \text{OFF} & \text{otherwise.} \end{cases}$$

We prove that an appropriate threshold policy is maximally stabilizing when the tasks are homogeneous and utilize the threshold policies to prove bounds on the throughput when the tasks are heterogeneous.

E. Simple bounds on the throughput

We start by deriving simple bounds on the throughput.

Proposition II.1. *For any $\tau > 0$ and f_W, we have that $\lambda^*_{max}(\tau, f_W) \in \left[(\bar{w}\mathcal{S}(1))^{-1}, (\bar{w}\mathcal{S}_{\min})^{-1}\right]$.*

Proof: The time between the start of service of successive tasks consists of two parts: the time to actively service a task, and the time when the server is idle, as governed by the task release control policy. The upper bound on the throughput is obtained by neglecting the idle times and by assuming that the server spends the least amount of time to service every task. The lower bound is proven by considering the trivial policy $u(t) \equiv \text{ON}$ as follows. Assume, by contradiction, that the queue length grows unbounded under this policy for some initial condition for an arrival rate $(\bar{w}\mathcal{S}(1))^{-1} - \epsilon$ for some $\epsilon > 0$. For a queue length growing unbounded, the server state exceeds $1 - \eta$ for any given $\eta > 0$ in some finite time T. Note that the queue length remains bounded until T. After T, all the service times per unit work are upper bounded by $\mathcal{S}(1) + \theta$ where $\theta \geq 0$ depends on η through the continuity of $\mathcal{S}(x)$. One can select η and hence θ such that

$$(\bar{w}\mathcal{S}(1) + \bar{w}\theta)^{-1} > (\bar{w}\mathcal{S}(1))^{-1} - \epsilon. \tag{2}$$

By the Strong Law of Large Numbers, with probability one, the average service time per task after T is upper bounded by $\bar{w}\mathcal{S}(1) + \bar{w}\theta$. Combining this with Equation (2), we get that, after T, the arrival rate is strictly less than the mean service time with probability one and hence the queue length cannot grow unbounded. This contradiction proves that the queue length remains bounded with probability one for an arrival rate $(\bar{w}\mathcal{S}(1))^{-1} - \epsilon$ for any ϵ and for any initial condition, which in turns proves that $\lambda^*_{\max}(\tau, f_W)$ is lower bounded by $(\bar{w}\mathcal{S}(1))^{-1}$. ■

The bounds obtained in Proposition II.1 can be shown to be tight for some simple cases. Consider first the case when $\mathcal{S} \equiv c$ for some constant $c > 0$. In this case, $\mathcal{S}(1) = \mathcal{S}_{\min} = c$ and hence Proposition II.1 implies that $\lambda^*_{\max}(\tau, f_W) = (\bar{w}c)^{-1}$ for all $\tau > 0$. Additionally, the trivial policy $u(t) \equiv \text{ON}$ is maximally stabilizing. Another simple case is when $\mathcal{S}(x)$ is non-increasing. In this case, $\mathcal{S}(1) = \mathcal{S}_{\min}$ and hence

Proposition II.1 implies that $\lambda^*_{\text{max}}(\tau, f_W) = (\bar{w}\mathcal{S}(1))^{-1}$ for all $\tau > 0$. One can show that the trivial policy $u(t) \equiv$ ON is maximally stabilizing in this case too.

We now derive tighter bounds on the throughput and design corresponding maximally stabilizing task release control policies.

III. Homogeneous Tasks

In this section, we consider the special case when the arriving tasks are homogeneous, i.e., every task brings in exactly the same deterministic amount of work with it. Formally, we let $f_W(w) = \delta_{\bar{w}}(w)$ for some $\bar{w} \in [\mathcal{W}_1, \mathcal{W}_2]$. We start by studying specific types of equilibria that are associated with the trivial policy $u(t) \equiv$ ON.

A. *One-task equilibria*

Let x_i be the server state at the beginning of service of the i-th task and let the queue length be zero at that instant. The server state upon the arrival of the $(i+1)$-th task is then obtained by integration of (1) over the time period $[0, 1/\lambda]$, with initial condition $x_0 = x_i$. Let x_i' denote the server state when it has completed service of the i-th task. Then, $x_i' = 1 - (1 - x_i)e^{-\bar{w}\mathcal{S}(x_i)/\tau}$. Assuming that $\bar{w}\mathcal{S}(x_i) \leq 1/\lambda$, we get that $x_{i+1} = x_i' e^{-(1/\lambda - \bar{w}\mathcal{S}(x_i))/\tau}$, and finally $x_{i+1} = (1 - (1 - x_i)e^{-\bar{w}\mathcal{S}(x_i)/\tau})e^{(\bar{w}\mathcal{S}(x_i) - 1/\lambda)/\tau} = \left(x_i - 1 + e^{\bar{w}\mathcal{S}(x_i)/\tau}\right) e^{-\frac{1}{\lambda\tau}}$. If λ, $\bar{w}$ and τ are such that $x_{i+1} = x_i$, then under the trivial control policy $u(t) \equiv$ ON, the server state at the beginning of all the tasks after and including the i-th task will be x_i and the queue length at most 1. We then say that the server is at *one-task equilibrium* at x_i. Therefore, for a given λ, $\bar{w}$ and τ, the one-task equilibrium server states correspond to $x \in [0, 1]$ that satisfy $x = \left(x - 1 + e^{\bar{w}\mathcal{S}(x)/\tau}\right) e^{-\frac{1}{\lambda\tau}}$ and $\mathcal{S}(x) \leq (\bar{w}\lambda)^{-1}$, i.e., $\mathcal{S}(x) = \frac{\tau}{\bar{w}} \log\left(1 - (1 - e^{\frac{1}{\lambda\tau}})x\right)$ and $\mathcal{S}(x) \leq (\bar{w}\lambda)^{-1}$. Let us define a function $\mathcal{R}$ as:

$$\mathcal{R}(x, \tau, \bar{w}, \lambda) := \frac{\tau}{\bar{w}} \log\left(1 - (1 - e^{\frac{1}{\lambda\tau}})x\right). \tag{3}$$

For a given $\tau > 0$ and $\lambda > 0$, define the set of one-task equilibrium server states as:

$$x_{\text{eq}}(\tau, \bar{w}, \lambda) := \{x \in [0, 1] \mid \mathcal{S}(x) = \mathcal{R}(x, \tau, \bar{w}, \lambda)\}. \tag{4}$$

Note that we did not include the constraint $\mathcal{S}(x) \leq (\bar{w}\lambda)^{-1}$ in the definition of $x_{\text{eq}}(\tau, \bar{w}, \lambda)$ in Equation (4). This is because this constraint can be shown to be redundant as follows. Equation (3) implies that, for any $\tau > 0$ and $\lambda > 0$, $\mathcal{R}(x, \tau, \bar{w}, \lambda)$ is strictly increasing in x and hence $\mathcal{R}(x, \tau, \bar{w}, \lambda) \leq \mathcal{R}(1, \tau, \bar{w}, \lambda) = (\lambda\bar{w})^{-1}$ for all $x \in [0, 1]$. Therefore, $\mathcal{S}(x_{\text{eq}}(\tau, \bar{w}, \lambda)) = \mathcal{R}(x_{\text{eq}}(\tau, \bar{w}, \lambda), \tau, \bar{w}, \lambda) \leq (\lambda\bar{w})^{-1}$.

The strict convexity of $\mathcal{S}(x) - \mathcal{R}(x, \tau, \bar{w}, \lambda)$ in x, which follows from the convexity assumption on $\mathcal{S}(x)$ and the strict concavity of $\mathcal{R}$ in x from Equation (3), implies that the cardinality of $x_{\text{eq}}(\tau, \bar{w}, \lambda)$ can take on values 0,1 and 2. For a given $\tau > 0$, $\bar{w} > 0$ and $\lambda > 0$, let $x_{\text{eq},1}(\tau, \bar{w}, \lambda)$ be the smaller element of $x_{\text{eq}}(\tau, \bar{w}, \lambda)$ if it is not empty and let $x_{\text{eq},2}(\tau, \bar{w}, \lambda)$ be the other element if the cardinality of $x_{\text{eq}}(\tau, \bar{w}, \lambda)$ is 2. Figure 3 illustrates these definitions through an example. One can show that $x_{\text{eq},1}(\tau, \bar{w})$ is a stable equilibrium point and $x_{\text{eq},2}(\tau, \bar{w})$, if it exists, is an unstable equilibrium point. Formally, one can show that, if $x_{\text{eq},1}(\tau, \bar{w})$ and $x_{\text{eq},2}(\tau, \bar{w})$ exist, then:

(i) for any $\tau > 0$ and $\bar{w} > 0$, the set $(x_{\text{eq},2}(\tau, \bar{w}), 1]$ is invariant and is not in the region of attraction of $x_{\text{eq},1}(\tau, \bar{w})$ or $x_{\text{eq},2}(\tau, \bar{w})$,

(ii) there exists a $\tau^* > 0$ such that for all $\tau > \tau^*$, the set $[0, x_{\text{eq},2}(\tau, \bar{w}))$ is invariant for all $\tau > \tau^*$. Moreover, in the limit as $\tau \to +\infty$, the set $[0, x_{\text{eq},2}(\tau, \bar{w}))$ is the region of attraction of $x_{\text{eq},1}(\tau, \bar{w})$.

We introduce a couple of additional definitions. For a given $\tau > 0$ and $\bar{w} > 0$, let

$$\begin{aligned} \lambda_{\text{eq}}^{\max}(\tau, \bar{w}) &:= \max\left\{\lambda > 0 \mid x_{\text{eq}}(\tau, \bar{w}, \lambda) \neq \emptyset\right\}, \\ x_{\text{th}}(\tau, \bar{w}) &:= x_{\text{eq},1}\left(\tau, \bar{w}, \lambda_{\text{eq}}^{\max}(\tau, \bar{w})\right). \end{aligned} \tag{5}$$

We now argue that the definitions in Equation (5) are well posed. Consider the function $\mathcal{S}(x) - \mathcal{R}(x, \tau, \bar{w}, \lambda)$. Since $\mathcal{R}(0, \tau, \bar{w}, \lambda) = 0$ for any $\tau > 0$, $\bar{w} > 0$ and $\lambda > 0$, and $\mathcal{S}(0) > 0$, we have that $\mathcal{S}(0) - \mathcal{R}(0, \tau, \bar{w}, \lambda) > 0$ for any $\tau > 0$, $\bar{w} > 0$ and $\lambda > 0$. Since $\mathcal{R}(1, \tau, \bar{w}, \lambda) = (\bar{w}\lambda)^{-1}$, $\mathcal{S}(1) - \mathcal{R}(1, \tau, \bar{w}, \lambda) < 0$ for all $\lambda < (\bar{w}\mathcal{S}_{\max})^{-1}$. Therefore, by the continuity of $\mathcal{S}(x) - \mathcal{R}(x, \tau, \bar{w}, \lambda)$, the set of equilibrium server states, as defined in Equation (4), is non-empty for all $\lambda < (\bar{w}\mathcal{S}_{\max})^{-1}$. Moreover, since $\mathcal{R}(x, \tau, \bar{w}, \lambda) \leq \mathcal{R}(1, \tau, \bar{w}, \lambda) = (\bar{w}\lambda)^{-1}$ for all $x \in [0, 1]$, $\mathcal{S}(x) - \mathcal{R}(x, \tau, \bar{w}, \lambda) \geq \mathcal{S}(x) - (\bar{w}\lambda)^{-1}$ for all $x \in [0, 1]$. Therefore, for all $\lambda > (\bar{w}\mathcal{S}_{\min})^{-1}$, the set of equilibrium states, as defined in Equation (4), is empty. Hence, $\lambda_{\text{eq}}^{\max}(\tau, \bar{w})$ and $x_{\text{th}}(\tau, \bar{w})$ are well defined.

In the rest of the paper, we will restrict our attention on those τ, $\bar{w} > 0$ and $\mathcal{S}(x)$ for which $x_{\text{th}}(\tau, \bar{w}) < 1$. Loosely speaking, this is satisfied when $\mathcal{S}(x)$ is increasing on some interval in $[0, 1]$ and the increasing part is *steep enough* (e.g., see Figure 3). It is reasonable to expect this assumption to be satisfied in the context of human operators whose performance deteriorates quickly at very high utilizations. The implications of the case when $x_{\text{th}}(\tau, \bar{w}) = 1$ are discussed briefly at appropriate places in the paper.

B. *Lower bound on the throughput*

We start by analyzing the throughput under a specific task release control policy. In particular, we consider the $x_{\text{th}}(\tau, \bar{w})$-threshold policy, where $x_{\text{th}}(\tau, \bar{w})$ is as defined in Equation (5).

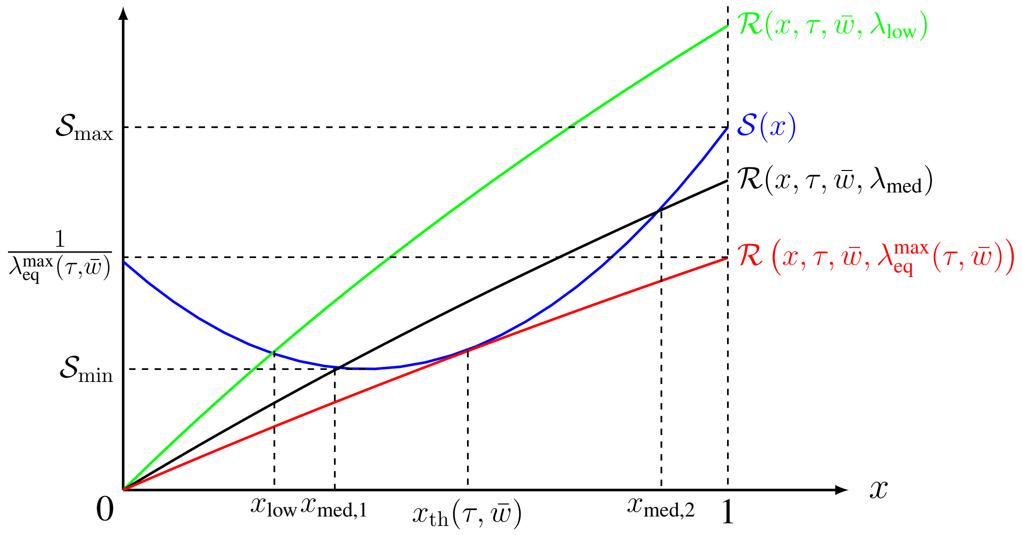


Fig. 3. A typical $\mathcal{S}(x)$ along with $\mathcal{R}(x,\tau,\bar{w},\lambda)$ for three values of λ: λ_{low}, λ_{med} and $\lambda_{\text{eq}}^{\max}(\tau,\bar{w})$ in the increasing order. Here, $x_{\text{low}} = x_{\text{eq},1}(\tau,\bar{w},\lambda_{\text{low}})$, $x_{\text{med},1} = x_{\text{eq},1}(\tau,\bar{w},\lambda_{\text{med}})$, $x_{\text{med},2} = x_{\text{eq},2}(\tau,\bar{w},\lambda_{\text{med}})$ and $x_{\text{th}}(\tau,\bar{w}) = x_{\text{eq},1}(\tau,\bar{w},\lambda_{\text{eq}}^{\max}(\tau,\bar{w}))$. Note that, since $x_{\text{th}}(\tau,\bar{w}) < 1$, $\lambda_{\text{eq}}^{\max}(\tau,\bar{w})$ is the value of λ at which $\mathcal{R}(x,\tau,\bar{w},\lambda)$ is tangent to $\mathcal{S}(x)$. The physical significance of x_{th} is that it is the equilibrium around which the total time (i.e., service time + idle time) for servicing a single task is minimum among all the equilibria points in $[0,1]$.

Theorem III.1. *For any $\tau > 0$, $\bar{w} > 0$, $x_0 \in [0,1]$, $n_0 \in \mathbb{N}$ and $\lambda \leq \lambda_{\text{eq}}^{\max}(\tau,\bar{w})$, if $x_{\text{th}}(\tau,\bar{w}) < 1$ then we have that $\limsup_{t\to+\infty} n_u(t,\tau,\lambda,\delta_{\bar{w}},x_0,n_0) < +\infty$ with u being the $x_{\text{th}}(\tau,\bar{w})$-threshold policy.*

The proof of this result, which can be found in [5], follows along the lines of the proof of Theorem IV.1, where we analyze threshold policies for the heterogeneous task case.

C. Upper bound on the throughput

We now prove that the $x_{\text{th}}(\tau,\bar{w})$-threshold policy is indeed maximally stabilizing by showing that no other task release control policy gives more throughput.

Theorem III.2. *For any $\tau > 0$, $\bar{w} > 0$, $x_0 \in [0,1]$, $n_0 \in \mathbb{N}$, $\lambda > \lambda_{\text{eq}}^{\max}(\tau,\bar{w})$ and $u \in \mathcal{U}$, if $x_{\text{th}}(\tau,\bar{w}) < 1$ then we have that $\limsup_{t\to+\infty} n_u(t,\tau,\lambda,\delta_{\bar{w}},x_0,n_0) = +\infty$.*

The proof of Theorem III.2 depends on a series of intermediate steps [5] that are summarized below. An important thing to note while understanding these steps is that any task release control policy can be equivalently described by the sequence of server states at the beginning of service of tasks resulting from its implementation.

(i) Consider first the following *constrained n-task static problem*: given n tasks, each with work $\bar{w}$, what is the fastest way for the dynamical server to service these tasks using a task release control policy, under the constraint that the initial and the final server state is x. We emphasize here that all the n tasks are initially enqueued and no new tasks arrive. One can then decompose the idle times and rearrange them to write the total time required for the n-task static problem to be equal to the sum of n constrained one-task static problems, each having the initial and final server states equal to each other but this state value possibly different across the n constrained one-task static problems. An illustration of such a decomposition and rearrangement procedure for a constrained 2-task static problem is provided in Figure 4. From the definition of one-task equilibria, one can lower bound the time for a constrained one-task static problem by $1/\lambda_{\text{eq}}^{\text{max}}(\tau, \bar{w})$. Therefore, the time for any constrained n-task static problem can be lower bounded by $n/\lambda_{\text{eq}}^{\text{max}}(\tau, \bar{w})$.

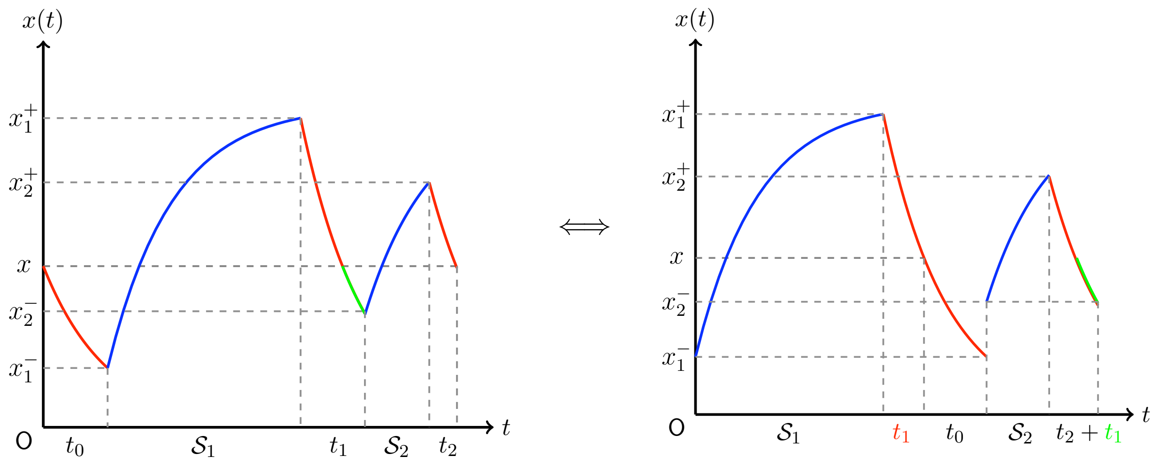


Fig. 4. Illustration of the decomposition and rearrangement procedure for a constrained 2-task static problem, i.e., the problem of servicing two tasks as quickly as possible under the constraint that the initial and final server states have to be same. The figure on the left shows the evolution of the server state under a generic task release control policy. The figure on the right shows the decomposition of the idle times and their rearrangement to get the total time as a sum of two 1-task problems, each with different boundary conditions. The active times are denoted by $\mathcal{S}_1$ and $\mathcal{S}_2$, whereas the idle times are denoted by t_0, t_1 and t_2 in the left as well as the right figures. The contiguous t_1 idle time on the left figure is split into two parts in the right figure. These two parts, again denoted by t_1 in red and green colors, are rearranged to get profile for two constrained 1-task problems with different boundary conditions.

(ii) We then use the lower bound from the constrained static problem to prove the bound for the original dynamic problem as follows. We first generalize our result for the static problem. In particular, we consider a *relaxed* n-task static problem, where the initial and final server states are not constrained to be equal but are constrained to lie within the interval $[x_L, x_U]$ for a given pair of constants x_L and x_U satisfying $0 < x_L < x_U < 1$. We show that, for any such x_L and x_U, the minimum time for the relaxed n-task static problem is lower bounded by $n/\lambda_{\text{eq}}^{\text{max}}(\tau, \bar{w}) + \kappa$, where the constant κ depends on the values of x_L and x_U. We then show the existence of a pair of x_L and x_U satisfying $0 < x_L < x_U < 1$ such that any task release control policy that lets the server state at the beginning of service of tasks to lie outside $[x_L, x_U]$ infinitely often cannot give a throughput that is more than $\lambda_{\text{eq}}^{\text{max}}(\tau, \bar{w})$. We finally collect these facts to show that the maximum throughput possible by a task release control policy is upper bounded by $\lambda_{\text{eq}}^{\text{max}}(\tau, \bar{w})$.

Theorems III.1 and III.2 imply that the throughput of the dynamical queue is $\lambda_{\text{eq}}^{\text{max}}(\tau, \bar{w})$, and that the $x_{\text{th}}(\tau, \bar{w})$-threshold policy is maximally stabilizing.

Remark III.3. (i) *A key step in the proof of Theorem III.2 is the rearrangement argument illustrated in Figure 4. This rearrangement argument holds for any server dynamics of the form $\dot{x} = f(x)$ during idling, with a continuous $f : [0, 1] \to \mathbb{R}$ such that $f(x) \neq 0$ for $x \in (0, 1]$ and $[0, 1]$ is invariant under the dynamics. This can be easily seen by noting that the time required by the server to idle from x_i to $x_f < x_i$ with x_i and x_f belonging to $(0, 1]$ is $\int_{x_i}^{x_f} \frac{1}{f(x)} dx$.*

(ii) *For a given $\bar{w}$, it is interesting to note the dependence of $\lambda_{\text{eq}}^{\text{max}}$ on τ. For any $\bar{w} > 0$, one can show that $\lambda_{\text{eq}}^{\text{max}}(\tau, \bar{w})$ is monotonically strictly decreasing in τ. Additionally, $\lim_{\tau \to 0^+} \lambda_{\text{eq}}^{\text{max}}(\tau, \bar{w}) = (\bar{w} \mathcal{S}_{\min})^{-1}$, and $\lim_{\tau \to +\infty} \lambda_{\text{eq}}^{\text{max}}(\tau, \bar{w}) = a$, where $a > 0$ is such that the line passing through the origin and having slope $\bar{w}/a$ is tangential to $\mathcal{S}$ in $(0, 1)$. An example plot of $\lambda_{\text{eq}}^{\text{max}}(\tau, \bar{w})$ is shown in Figure 5.*

(iii) *If $x_{\text{th}}(\tau, \bar{w}) = 1$, then one can show that, for any $\epsilon > 0$, there exists no stabilizing task release control policy for arrival rates greater than $\lambda_{\text{eq}}^{\text{max}}(\tau, \bar{w}) + \epsilon$, i.e., $\lambda_{max}^*(\tau, \bar{w}) \leq \lambda_{\text{eq}}^{\text{max}}(\tau, \bar{w}) + \epsilon$.*

(iv) *For any $\lambda < \lambda_{\text{eq}}^{\text{max}}(\tau, \bar{w})$, Theorem III.1 holds true for any x-threshold policy with $x \in [x_{\text{eq},1}(\tau, \bar{w}), x_{\text{eq},2}(\tau, \bar{w})]$, if $x_{\text{eq},2}(\tau, \bar{w})$ exists or $x \in [x_{\text{eq},1}(\tau, \bar{w}), 1]$ otherwise. It is possible to exploit this flexibility to design a threshold policy with dynamically changing threshold values to ensure that, for any $n_0 \in \mathbb{N}$ and $x_0 \in [0, 1]$, the queue length goes to zero in finite time for any $\lambda < \lambda_{\text{eq}}^{\text{max}}(\tau, \bar{w})$.*

D. Simulations

In this section, we report results from some numerical experiments. We select $\tau = 300\,s$, $\mathcal{S}(x) = 229x^2 - 267x + 99\,s$ and $\bar{w} = 1$. These values roughly correspond to the model estimated from

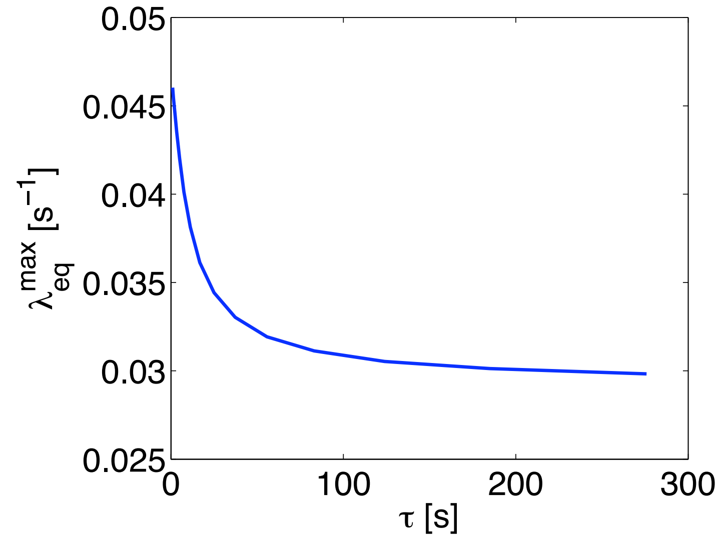


Fig. 5. Plot of $\lambda_{\text{eq}}^{\max}(\tau,\bar{w})$ versus τ for $\mathcal{S}(x) = 229x^2 - 267x + 99$ and $\bar{w} = 1$. The decreasing nature of the plot implies that the maximum throughput decreases with the decrease in the sensitivity of the operator.

experimental data, as reported in [20]. For these values, $x_{\text{th}}(\tau,\bar{w}) \approx 0.65$ and $\lambda^*_{\text{max}}(\tau,\bar{w}) \approx 0.03\,s^{-1}$. We set the arrival rate to be $\lambda = 0.025\,s^{-1}$. For this case, the stable equilibrium $x_{\text{eq},1}(\tau,\bar{w},\lambda)$ is 0.53 and the unstable equilibrium $x_{\text{eq},2}(\tau,\bar{w},\lambda)$ is 0.81. Following earlier remarks (just before Equation (5)) on the stability of the equilibria in $x_{\text{eq}}(\tau,\bar{w})$, Figure 6(a) shows the expected unstable behavior for initial condition $x_0 = 0.9$ under the trivial policy $u(t) \equiv$ ON. On the other hand, following Theorem III.1, Figure 6(b) shows that the queue is stable under the $x_{\text{th}}(\tau,\bar{w})$-threshold policy. Note that, following earlier remarks on the stability of the equilibrium points, if x_0 is in $[0, 0.81)$, then the queue could be stable even under $u(t) \equiv$ ON. We performed experiments with several such x_0 and observed that this indeed is the case.

We performed similar simulations for heterogeneous tasks. We again selected $\lambda = 0.025\,s^{-1}$ and $\tau = 300\,s$. We set f_W to be a uniform distribution over $[5, 45]$ and let $\mathcal{S}(x) = (229x^2 - 267x + 99)/25\,s$. These parameters ensure that the average state-dependent service times are same as the deterministic state-dependent service times selected for the simulations for homogeneous tasks. We observe that the queue is unstable under $u(t) \equiv$ ON even for x_0 in $[0, 0.81)$ while it is stable for all $x_0 \in [0, 1]$ under the $x_{\text{th}}(\tau,\bar{w})$-threshold policy, see Figure 7 for an example. In the next section, we develop a formal theory for the stability of the dynamical queue when the tasks are heterogeneous.

IV. Heterogeneous Tasks

In this section, we return to the general case when f_W is not necessarily the delta distribution. For this general case, we are not able to compute the throughput exactly but we provide meaningful bounds.

A. Lower bound on the throughput

We first prove a surprising lower bound.

Theorem IV.1. *For any f_W and $\tau > 0$, we have that*

$$\lambda^*_{max}(\tau, f_W) \geq \lambda^*_{max}(\tau, \delta_{\bar{w}}),$$

where the inequality is strict if and only if $f_W \neq \delta_{\bar{w}}$.

Proof: We first prove an upper bound on the mean inter-task time under a threshold policy, which will be critical in proving the main result. The time between servicing successive tasks under the x-threshold policy is the random variable $w\mathcal{S}(x) + \tau \log\left(\frac{1-(1-x)e^{-w\mathcal{S}(x)/\tau}}{x}\right)$. Due to the bounded support assumption of f_W, this random variable has a finite variance. The expected value can be written as

$$E_{f_W}\left[w\mathcal{S}(x) + \tau \log\left(\frac{1-(1-x)e^{-w\mathcal{S}(x)/\tau}}{x}\right)\right] = \bar{w}\mathcal{S}(x) + \tau E_{f_W}\left[\log\left(1-(1-x)e^{-w\mathcal{S}(x)/\tau}\right)\right] - \tau\log(x). \tag{6}$$

The function $1-(1-x)e^{-y/\tau}$ is strictly concave in y (except for $x = 1$) and non-negative for every $\tau > 0$. Since every non-negative strictly concave function is also logarithmically strictly concave, e.g., see [21], applying Jensen's inequality to $E_{f_W}\left[\log\left(x-1+e^{w\mathcal{S}(x)/\tau}\right)\right]$ and using Equation (6), one gets that for all $f_W \neq \delta_{\bar{w}}$,

$$E_{f_W}\left[w\mathcal{S}(x) + \tau \log\left(\frac{1-(1-x)e^{-w\mathcal{S}(x)/\tau}}{x}\right)\right] < \bar{w}\mathcal{S}(x) + \tau \log\left(\frac{1-(1-x)e^{-\bar{w}\mathcal{S}(x)/\tau}}{x}\right). \tag{7}$$

We now use this bound to prove the main result. Let

$$T := \min_{x\in[0,1]} E\left[w\mathcal{S}(x) + \tau \log\left(\frac{1-(1-x)e^{-w\mathcal{S}(x)/\tau}}{x}\right)\right], \tag{8}$$

where we let $E[\infty] := \infty$. Let x^* be the minimizer in Equation (8). Note that $x^* > 0$, because at $x = 0$, the value of the expectation is $+\infty$ and at any other value of x (say $x = 0.5$), the value of the expectation is finite. Using Equation (7) for $x = x_{\text{th}}(\tau, \bar{w})$, we get that $T < 1/\lambda^{\text{max}}_{\text{eq}}(\tau, \bar{w}) = 1/\lambda^*_{\text{max}}(\tau, \delta_{\bar{w}})$, where the equality follows from Theorems III.2 and III.1. Consider the evolution of the queue under the x^*-threshold policy for an arrival rate $1/T - \eta$ for any $\eta \in (0, 1/T - \lambda^*_{\text{max}}(\tau, \delta_{\bar{w}}))$. We now prove the boundedness of the queue length, thereby proving the theorem.

Let x_i and t_i be the server state and time instants respectively at the beginning of service of the i-th task. For brevity in notation, let $n(t)$ be the queue length at time t. For any $x_0 \in [0,1]$ and $n_0 \in \mathbb{N}$, considering the possibility when $x_0 > x^*$ we have that $n(t_1) = \max\{0, n_0 - 1, n_0 - 1 + \lfloor \lambda\, t_1 \log(x_0/x^*) \rfloor\}$. Consider the following two cases:

- **State 1:** $x_1 = x^*$. While $n(t_i) > 0$, we have that $x_{i+1} = x^*$ and $t_{i+1} - t_i = w\mathcal{S}(x^*)$, where w is independently and identically distributed according to f_W. Applying the Strong Law of Large Numbers to the sequence $t_i - t_{i-1}$, we have that, for every $\epsilon > 0$, there exists $n(\epsilon) > 0$ such that, for all $n \geq n(\epsilon)$,
$$\Pr\left(\Big|\frac{t_{n+1} - t_1}{n} - E_{f_W}\left[w\mathcal{S}(x^*) + \tau \log\left(\frac{x^* - 1 + e^{w\mathcal{S}(x^*)/\tau}}{x^*}\right)\right]\Big| < \epsilon\right) = 1.$$
This implies that, with $\epsilon := \frac{1}{2}\left(\frac{1}{\lambda} - \bar{w}\mathcal{S}(x^*)\right)$, for all $n \geq \max\left\{n(\epsilon), \frac{2n_0}{1 - \lambda\bar{w}\mathcal{S}(x^*)}\right\}$, we have that
$$t_{n+1} - t_1 < n\left(\frac{1}{\lambda} - \frac{n_0}{n\lambda}\right) = \frac{n - n_0}{\lambda}.$$
In other words, for any initial queue length n_0, with probability one, after service of finite number of tasks the queue length goes to zero and the server state drops below x^*. Thereafter, we appeal to the next case by resetting x_i and t_i as x_1 and t_1 respectively. Moreover, with these notations, $n(t_1)$ will be zero.
- **State 2:** $x_1 < x^*$. While the queue length is non-zero, the server is never idle. The maximum amount of continuous service time required for the server state to cross x^* starting from any $x_1 < x^*$ is upper bounded by $-\tau \log(1 - x^*) + \mathcal{W}_2\mathcal{S}_{\max}$. This is possibly followed by an idle time that is upper bounded by $-\tau \log x^*$, at the end of which the server state is x^*. Therefore, the maximum number of outstanding tasks when the server state reaches x^* is upper bounded by $n_1 + \lceil \tau \log(1 - x^*)\left((\mathcal{W}_2\mathcal{S}_{\max})^{-1} - \lambda\right)\rceil + \lceil -\lambda\tau \log x^* \rceil$. Thereafter, we appeal to the earlier case by resetting $x_1 = x^*$ and n_1 to be the number of outstanding tasks when the server state reaches x^*.

In summary, when the system is in State 1, the queue length decreases to zero with probability one at which point it enters State 2. When the system is in State 2, it stays in it for ever or eventually enters State 1 with bounded queue length. Collecting these facts, we arrive at the result. ■

Remark IV.2. (i) *Theorem IV.1 shows that, the throughput strictly increases with the introduction of stochasticity in service times. This is novel because not only does this imply that the throughput of the dynamical queue depends also on the second moment of the service times, but also that this dependence occurs in an unexpected way. To the best of our knowledge, such a phenomenon has not been reported for queueing systems so far.*

(ii) *A key step in the proof of Theorem IV.1 is Equation* (7) *where we use Jensen's inequality. A physical intuition behind this can be given as follows. The total time between successive tasks, as given by the expression inside the expectation operator on the left hand side of Equation* (7) *is composed of two parts: active time given by $w\mathcal{S}(x)$ and the idle time given by $\tau \log\left(\frac{1-(1-x)e^{-w\mathcal{S}(x)/\tau}}{x}\right)$. The active time is linear in w and hence it is unaffected by the expectation operator. However, the idle time is concave in w. In other words, the idle time has a diminishing marginal, i.e., the amount of idle time required by the server to have its state decreased by $\triangle x$ starting from an initial state x_i decreases for increasing x_i. This property of the idle time is key to the increase in throughput with the introduction of heterogeneity in the tasks, and hence heterogeneity in the idle times.*

(iii) *The concavity property that leads to Equation* (7) *is associated with the server dynamics and is independent of the convexity of $\mathcal{S}(x)$, and hence Equation* (7) *is valid for any $\mathcal{S}(x)$ that does not necessarily satisfy the convexity property assumed in this paper.*

B. Upper bound on the throughput

In this section, we derive an upper bound on the maximum throughput possible using any task release control policy.

Theorem IV.3. *For any f_W and $\tau > 0$, we have that*

$$\lambda^*_{max}(\tau, f_W) \leq \left(E_{f_W}\left[1/\lambda^{\max}_{\text{eq}}(\tau, w)\right]\right)^{-1}.$$

Proof: The proof is similar to that of Theorem III.2. The difference is in the time required for the constrained n-task static problem. Consider a set of n tasks associated with works $w_1, \ldots, w_n$, where each w_i is identically and independently sampled from f_W. It is desired to service these n tasks in the fastest possible way using a task release control policy under the constraint that the initial and final server state is x. The time required for the constrained 1-task problem is $w_1\mathcal{S}(x) + \tau \log\left(\frac{x-1+e^{w_1\mathcal{S}(x)/\tau}}{x}\right)$ which is lower bounded by $1/\lambda^{\max}_{\text{eq}}(\tau, w_1)$. Using the same decomposition and rearrangement approach as before, the time required for the constrained n-task problem can be lower bounded by $\sum_{i=1}^n 1/\lambda^{\max}_{\text{eq}}(\tau, w_i)$. Using Strong Law of Large Numbers, one can show that, with probability one, as $n \to +\infty$, the average time required per task is lower bounded by $E_{f_W}\left[1/\lambda^{\max}_{\text{eq}}(\tau, w)\right]$. The rest of the proof follows similarly. ■

Remark IV.4. (i) *Theorems IV.1 and IV.3 do not imply that a threshold policy is maximally stabilizing when the tasks are heterogeneous. However, the proof of Theorem IV.1 implies that the best threshold policy would correspond to the x^*-threshold policy, where x^* is the minimizer in Equation* (8).

(ii) *Theorem IV.3 holds true even for any f_W with unbounded support, but having finite mean and variance.*

C. Simulations

For $\tau = 300\,s$, f_W a uniform distribution over $[5, 45]$ and $\mathcal{S}(x) = (229x^2 - 267x + 99)/25\,s$, the lower bound, as given by Theorem IV.1 is computed to be about $0.031\,s^{-1}$ and the upper bound, as given by Theorem IV.3 is computed to be about $0.039\,s^{-1}$. Note that, these bounds are tighter than the bounds provided by Proposition II.1.

Theorem IV.1 suggests that, for appropriate f_W, one could possibly increase $\lambda_{\text{eq}}^{\max}(\tau, f_W)$ up to $(\bar{w}\mathcal{S}_{\min})^{-1}$ for all $\tau > 0$ and, hence, by Proposition II.1, one could achieve the maximum possible throughput. Figure 8 demonstrates that this is feasible through an illustrative extreme example. The solid curve in Figure 8, which represents the throughput curve, shows that for tasks with large heterogeneity, the throughput for a given threshold value x closely follows the inverse of $\bar{w}\mathcal{S}(x)$ which itself is the maximum possible throughput under the x-threshold policy. In particular, the throughput under the $\arg\min_{x\in[0,1]} \mathcal{S}(x)$-threshold policy is very close to $(\bar{w}\mathcal{S}_{\min})^{-1}$.

V. Experiments

In this section, we report preliminary empirical evidence aimed at validating the dynamical queue framework for human operators. The experiments involved human subjects sitting in front of a computer screen and answering multiple-choice analogy questions; see Figure 9 for a snapshot of the interface during one such question. The experiment started with a large number of questions which were allocated to the subjects one at a time in a queueing fashion, where the time between successive questions was controlled by us.

A total of 26 subjects with English as the primary spoken language were recruited for the experiments. However, most of them rated their command of English only to be a 3 on a scale of 1 to 5, with 5 being the most proficient. Most participants had also studied for or taken the Verbal Scholastic Assessment Test previously, though no one admitted to having a good skill set for analogy questions. The experiment with every subject lasted for 45 minutes. Before the start of the experiment, the subjects underwent an orientation familiarizing them with the interface by allowing them to practise a few sample questions. The subjects were also notified that their score at the end of the experiment would be computed by assigning one point for every correct answer and deducting one point for every incorrect answer. In addition to the remuneration for their participation, a reward was set for the subject with the maximum score at the end of the experiments, and all the subjects were notified of this during their orientation. The subjects were also explicitly instructed that their best strategy to maximize the score would be to maximize the number of correct answers while answering as many questions as quickly as possible. At the end of the experiments, the subjects were asked to take a feedback survey about their strategy on answering the questions, subjective assessment of the workload during the experiment, etc.

During the experiments, we did not show the amount of time spent or time remaining in the experiment to avoid any factor (apart from their own perceived workload) pressurizing the subjects. The number of questions in the queue and feedback on the subject's performance were not shown as well. The questions were chosen in a manner so that there was no bias in the difficulty of the questions at any point in the experiment. Additionally, the questions were queued in the same order for every subject for consistency. The subjects were not given the option to go back and change their answers upon submission and were required to provide an answer before moving on to the next question, with no option to skip questions.

Based on the analysis of pilot tests, we used $x_0 = 0.5$ and $\tau = 150\,s$ in our model in Equation (1) to estimate the state of all the subjects in real time. For every subject, the tasks were allocated to the subject under threshold based task release control policies, where the thresholds for the first 30 minutes were selected as follows: threshold value of 0.6 for the first 300 seconds, followed by the threshold value of 0.7 for the next 700 seconds, followed by the threshold value of 0.8 for the next 800 seconds. We recorded the times at which the questions were assigned, the times at which the questions were answered and whether the answers were correct or incorrect. The difference between the time at which a question was answered and the time at which the question was assigned is the total service time $w\mathcal{S}(x)$ for that question. Among the service times that were recorded in the first 30 minutes, we calculated the mean of the service times for which the initial server state was $x = 0.6 \pm 0.02$ and stored it as $\bar{w}\mathcal{S}(0.6)$. We repeated this procedure to get $\bar{w}\mathcal{S}(0.7)$ and $\bar{w}\mathcal{S}(0.8)$. We found the quadratic function passing through these three points to get the function $\bar{w}\mathcal{S}(x)$ over $x \in [0, 1]$, and used it to compute x_{th} according to Equation (5). The value of x_{th} was found to be around 0.7 for all the subjects. All these computations were done in real time. We implemented the x_{th}-threshold policy for the last 15 minutes of the experiment, and recorded the same data as in the first 30 minutes. The participants were not informed about the details or even the presence of task release control policies in the experiments. Among the 26 subjects, we found that 11 did not have service times with initial state greater than 0.7 at all, i.e., these subjects were answering the questions very quickly, possibly without enough deliberation over the questions. This was also reflected in their answers to the survey questions at the end of the experiments. Hence, we do not include their data in our analysis.

Our first objective is to compare the statistics of service times with different initial states (0.6, 0.7 and 0.8) and across different subjects. Additionally, we also compare the statistics of correctness (1 for correct and 0 for incorrect) of the answers with different initial server states and the subjects. For this purpose, we performed ANOVA test on the service times and correctness recorded for the subjects for the three initial state values (0.6, 0.7 and 0.8) using the `anovan` command from the statistics toolbox of MATLAB. For the service times, the p-values [1] corresponding to the effect of different different initial states as well as different subjects were calculated to be zero each, implying that the mean service time for at least one initial state is statistically significantly different than the mean service time for other initial states, as well as that the mean service time for at least one subject is statistically significantly different than the mean service times of other subjects. For correctness, the p-values corresponding to the effect of different initial states and different subjects were calculated to be 0.0031 and 0.0003 respectively. This also implies that the mean correctness (i.e., the ratio of correct answers) for at least one initial state is statistically significantly different than the mean correctness for other initial states, as well as that the mean correctness for at least one subject is statistically significantly different than the mean correctness for the other subjects. Next, for detailed comparisons, we performed the ANOVA tests on the service times and the correctness with respect to the initial states for every subject individually. The p-values from these tests are reported in Table I.

Table I implies that, for most of the subjects (10 out of 15), the mean service time for at least one initial state is statistically significant different than the mean service times for other initial states, with the state evolution as given by our model in Equation (1). The mean correctness does not show statistically significant difference with respect to initial server states for all subjects, except for subject number 20. Figure 10 contains box plots for the service times with respect to different initial states for subject numbers 25 (minimum p-value) and 13 (maximum p-value). In these box plots, as is the standard convention, the central red line is the median of the data set, the middle blue horizontal lines at the edges of the boxes are the 25th and 75th percentiles and the extreme black horizontal lines represent the most extreme data points not considering outliers; the outliers are plotted individually by the red '+' symbols. Note that, the box plot for subject number 25 suggests a U-shaped relationship, similar to the Yerkes-Dodson law, between the service times and the utilization ratio as defined by our model in Equation (1). Figure 11 is the histogram for the service time distribution for subject number 25 corresponding to initial state

[1]A p-value is the probability of obtaining a test statistic as extreme as the data set obtained if the (null) hypothesis that the data set has the same mean overall is true.

TABLE I

P-VALUES FROM THE ANOVA TESTS FOR THE COMPARISON OF SERVICE TIMES AND CORRECTNESS FOR DIFFERENT INITIAL STATES, PERFORMED FOR EVERY INDIVIDUAL SUBJECT

Subject No.	p-value for the service times	p-value for the correctness
3	0.0373	0.7914
4	0.0002	0.2951
6	0.0027	0.5181
8	0.0039	0.6185
9	0.1938	0.531
13	0.7908	0.4582
16	0.0132	0.6337
18	0.1537	0.7138
19	0.1524	0.6836
20	0.3553	0.0366
21	0.0027	0.0827
22	0	0.5321
24	0	0.515
25	0	0.0774
26	0.0001	0.7257

$x = 0.7$.

The analysis above suggests that service times in these experiments show dependence on the initial state and that the evolution of the state of the human subjects could be modeled by Equation (1). Hence, according to the theory developed in this paper, one could obtain maximum throughput under x_{th}-threshold task release control policy. Our next objective is to analyze the experimental data to see if the application of x_{th}-threshold task release control policy resulted in any significant improvement in *empirical* throughput over other threshold task release control policies, where the empirical throughput is defined as the number of tasks serviced divided by the time over which the control policy is implemented. Since the value of x_{th} was close to 0.7 for all the subjects, in order to calculate the empirical throughput under the x_{th}-threshold policy, we incorporate data collected during the last 15 minutes of the experiment as well as during the 700 seconds in the early part of the experiment when the 0.7-threshold task release control policy was implemented. Note that, this way of combining data for the x_{th}-threshold policy also compensates for the possible effect due to the ascending and then descending sequence of thresholds selected in the experiment. We compare the empirical throughput under the x_{th}-threshold policy with the empirical throughput computed from the data collected under other threshold policies, i.e., 0.6 and 0.8-threshold policies. We also compare the effect of x_{th}-threshold policy on empirical correctness in a similar fashion. The results are shown in Figures 12 and 13, where we compare the histograms for the empirical throughput and correctness, respectively, under the 0.6 and 0.8-threshold policies on one hand and the x_{th}-threshold policy on the other hand. Figures 12 and 13 suggests that the x_{th}-threshold policy does indeed result in greater throughput *and* greater mean correctness in comparison to other threshold policies.

VI. Conclusion and future work

In this paper, we presented a dynamical queue framework as a formal approach to task management for human operators. Inspired by empirical laws, we considered a novel dynamical queue model for human operators, where the service times are dependent on the state of a simple underlying dynamical system. We studied the stability of such dynamical queues under deterministic inter-arrival times. For homogeneous tasks, we proved that a task release control policy that releases a task to the server only when its state is below an appropriately chosen threshold value gives the maximum throughput. For heterogeneous tasks, we showed that the throughput strictly increases with the introduction of heterogeneity. The deterministic inter-arrival time assumption in our analysis is not binding and the results extend to the case where the inter-arrival times are sampled identically and independently from a common distribution having bounded variance. We also reported preliminary empirical evidence to justify the dynamical queue model for human operators.

The ability of appropriately designed threshold based task control policies to stabilize an otherwise unstable queue, and the associated throughput optimality results proven in this paper, provide a formal methodology and justification for similar approaches commonly adopted in practice, e.g., see [13]. From a scientific point of view, the increase in throughput due to heterogeneity in tasks is a novel phenomenon for queueing systems. This result provides an additional dimension to improving throughput of dynamical queues by repackaging the tasks until one has maximum heterogeneity across the repackaged tasks. Moreover, if one has to decide upon a quantization scheme for a large task that has to be completely by a human operator as quickly as possible, this result suggests that uniform quantization is the worst possible quantization.

In future, we plan to extend our analysis to characterize the average wait time of dynamical queues.

We also plan to extend our formulation and analysis to also incorporate accuracy of the job done by the operators in the performance metric. We intend to perform extensive experiments to develop a high fidelity dynamical model for human operators. Finally, we also plan to extend our framework to align it more closely to conventional state-dependent queues where the notion of server state is closely related to the amount of outstanding work rather than the past utilization.

Acknowledgments

The authors thank Christine Siew for conducting the experiments reported in the paper. K. Savla thanks Prof. Devavrat Shah for helpful discussions. This research was partially supported by the Michigan/AFRL Collaborative Center on Control Science, AFOSR grant no. FA 8650-07-2-3744, and by the C. S. Draper Career Development chair at the Massachusetts Institute of Technology.

References

[1] K. Savla, T. Temple, and E. Frazzoli, "Human-in-the-loop vehicle routing policies for dynamic environments," in *IEEE Conf. on Decision and Control*, pp. 1145–1150, 2008.

[2] L. Kleinrock, *Queueing Systems I: Theory*. Wiley-Interscience, 1975.

[3] S. Asmussen, *Applied Probability and Queues*. Springer, 2003.

[4] G. Koole and A. Mandelbaum, "Queueing models of call centers: An introduction," *Annals of Operations Research*, vol. 113, pp. 41–59, 2002.

[5] K. Savla and E. Frazzoli, "Maximally stabilizing admission control policy for a dynamical queue," *IEEE Trans. on Automatic Control*, vol. 55, no. 11, pp. 2655–2660, 2010. Available at `http://arxiv.org/abs/0909.3651`.

[6] R. M. Yerkes and J. D. Dodson, "The relation of strength of stimulus to rapidity of habit-formation," *Journal of Comparative Neurology and Psychology*, vol. 18, pp. 459–482, 1908.

[7] J. H. Dshalalow, ed., *Frontiers in Queuing Models and Applications in Science and Engineering*, ch. Queueing Systems with State Dependent Parameters. CRC press, Inc., 1997.

[8] R. Bekker and S. C. Borst, "Optimal admission control in queues with workload-dependent service rates," *Probability in the Engineering and Informational Sciences*, vol. 20, pp. 543–570, 2006.

[9] M. L. Cummings and C. E. Nehme, "Modeling the impact of workload in network centric supervisory control settings," in *2nd Annual Sustaining Performance Under Stress Symposium*, (College Park, MD), Feb. 2009.

[10] L. F. Bertuccelli, N. Pellegrino, and M. Cummings, "Choice tasks in modeling relooks in UAV search missions," in *American Control Conference*, (Baltimore, MD), pp. 2410–2415, 2010.

[11] V. Srivastava, R. Carli, F. Bullo, and C. Langbort, "Task release control for decision making queues," in *American Control Conference*, (San Francisco, CA), 2011. To appear.

[12] C. R. Glassey and M. G. C. Resende, "A scheduling rule for job release in semiconductor fabrication," *Operations Research Letters*, vol. 7, no. 5, pp. 213–217, 1988.

[13] J. W. M. Bertrand and H. P. G. V. Ooijen, "Workload based order release and productivity: a missing link," *Production Planning and Control*, vol. 13, no. 7, pp. 665–678, 2002.

[14] S. Stidham, “Optimal control of admission to queueing system,” *IEEE Trans. Automatic Control*, vol. 30, pp. 705–713, Aug 1985.

[15] M. L. Cummings and P. J. Mitchell, “Operator scheduling strategies in supervisory control of multiple uavs,” *Aerospace Science and Technology*, vol. 11, no. 4, pp. 339–348, 2007.

[16] S. Hameed, T. Ferris, S. Jayaraman, and N. Sarter, “Using informative peripheral visual and tactile cues to support task and interruption management,” *Human Factors*, vol. 51, no. 2, pp. 126–135, 2009.

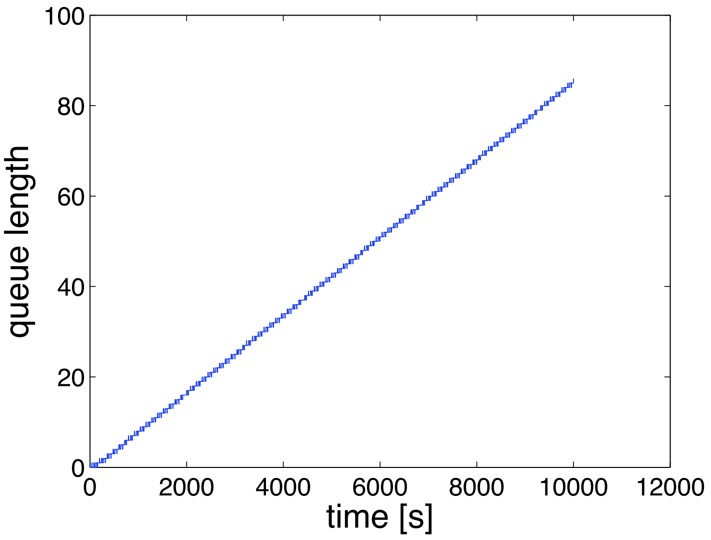
[17] F. P. Kelly and R. J. Williams, “Heavy traffic on a controlled motorway,” in *Probability and Mathematical Genetics: Papers in Honour of Sir John Kingman* (N. H. Bingham and C. M. Goldie, eds.), no. 378 in London Mathematical Society Lecture Notes Series, Cambridge University Press, 2010.

[18] J. Le Ny and H. Balakrishnan, “Distributed feedback control for an eulerian model of the national airspace system,” in *Proceedings of the American Control Conference*, (St. Louis, MO), pp. 2891–2987, 2009.

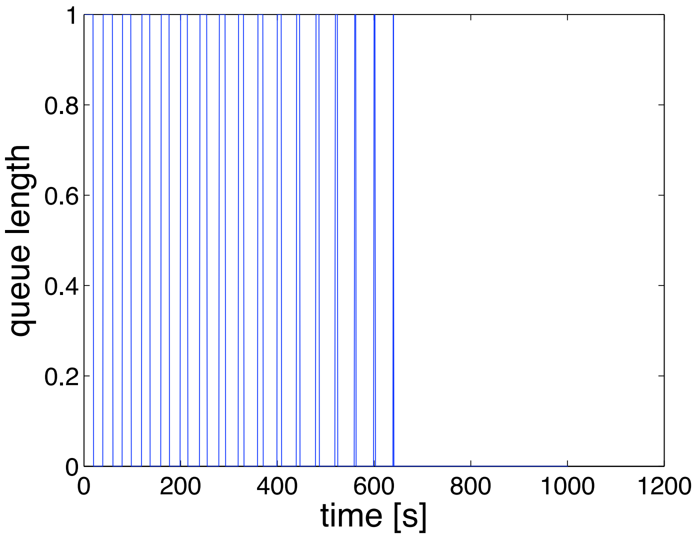
[19] P. A. Hancock and N. Meshkati, eds., *Human mental workload*. No. 52 in Advances in Psychology, Elsevier Science Publishers B. V., 1988.

[20] K. Savla, C. Nehme, T. Temple, and E. Frazzoli, “Efficient routing of multiple vehicles for human-supervised services in a dynamic environment,” in *AIAA Conf. on Guidance, Navigation, and Control*, (Honolulu, HI), 2008.

[21] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

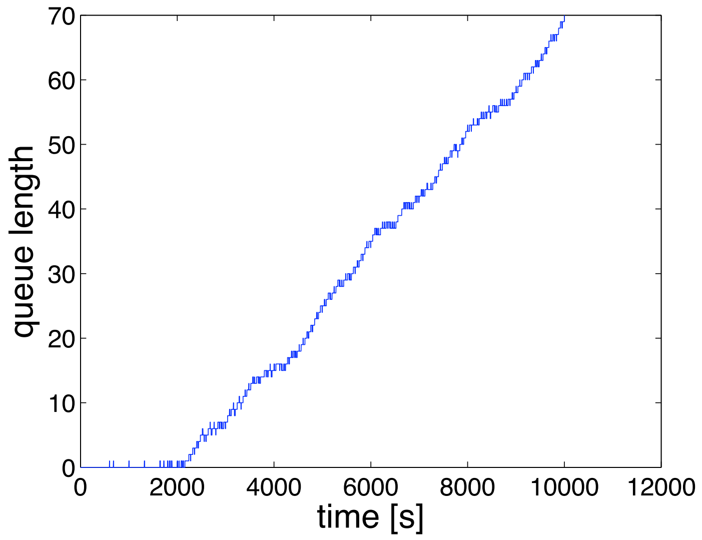


(a)

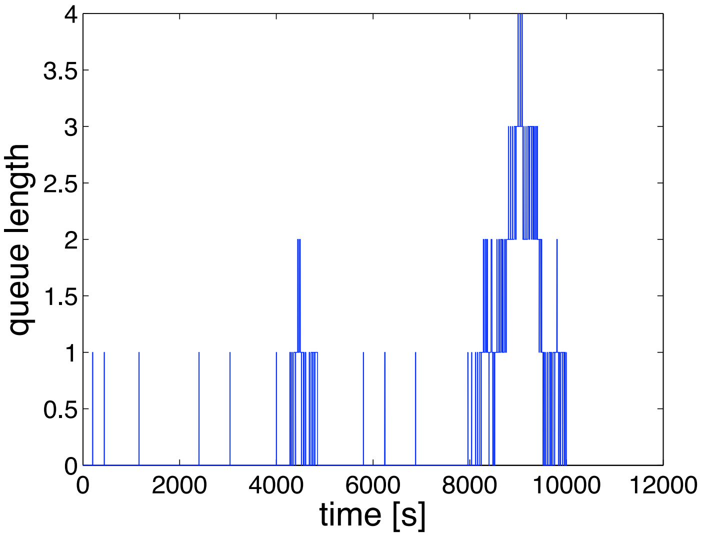


(b)

Fig. 6. Comparison of the queue length for homogeneous tasks under (a) $u(t) \equiv \text{ON}$ and (b) $x_{\text{th}}(\tau, \bar{w})$-threshold policy. Simulation parameters for both the cases are $\lambda = 0.025\,s^{-1}$, $\mathcal{S}(x) = 229x^2 - 267x + 99\,s$, $\bar{w} = 1$, $\tau = 300\,s$, $x_0 = 0.9$ and $n_0 = 0$. These figures illustrate that the throughput of a dynamical queue under the threshold policy is more than the throughput under the trivial policy $u(t) \equiv \text{ON}$.



(a)



(b)

Fig. 7. Comparison of the queue length during a sample run with heterogeneous tasks under (a) $u(t) \equiv \mathrm{ON}$ and (b) $x_{\mathrm{th}}(\tau, \bar{w})$-threshold policy. Simulation parameters for both the cases are $\lambda = 0.025\, s^{-1}$, $\mathcal{S}(x) = (229x^2 - 267x + 99)/25\, s$, f_W uniform distribution over $[5, 45]$, $\tau = 300\, s$, $x_0 = 0.5$ and $n_0 = 0$. These figures illustrate that the throughput of a dynamical queue under the threshold policy is more than the throughput under the trivial policy $u(t) \equiv \mathrm{ON}$.

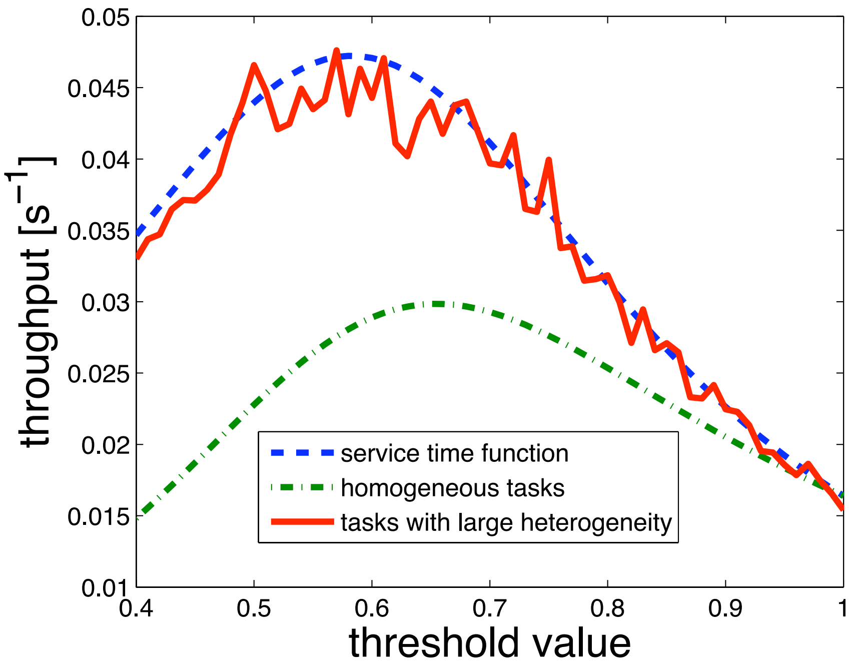
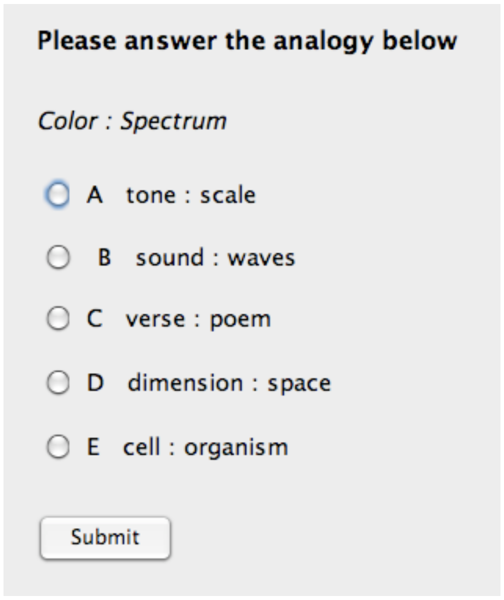


Fig. 8. Illustration of the maximum possible throughput under extreme task heterogeneity. The solid curve represents the throughput curve under threshold policies when $f_W(w)$ is a binary random variable that takes values 0.01 and 5000 with probabilities 0.995 and 0.005 respectively, and $\mathcal{S}(x) = (229x^2 - 267x + 99)/25\,s$; the dash-dotted curve represents the throughput curve under threshold policies when $f_W(w) = \delta_{25}(w)$ and the same $\mathcal{S}(x)$; the dashed curve represents the inverse of the function $\bar{w}\mathcal{S}(x)$, with $\bar{w} = 25$ and the same $\mathcal{S}(x)$. The peak of the dashed curve corresponds to the maximum possible throughput of the dynamical queue, as given by Proposition II.1, whereas the peak of the the dash-dotted curve corresponds to the maximum possible throughput when all the tasks are homogeneous, as given by Theorem III.2. The difference between these two peaks is the potential gain in throughput that one can obtain by having heterogeneity in the tasks. The fact that the solid curve closely follows the dashed curve, implies that the maximum throughout could be achieved from the queue by the corresponding composition of the tasks.



Please answer the analogy below

Color : Spectrum

- A tone : scale
- B sound : waves
- C verse : poem
- D dimension : space
- E cell : organism

Fig. 9. Experiment interface during a multiple choice analogy question.

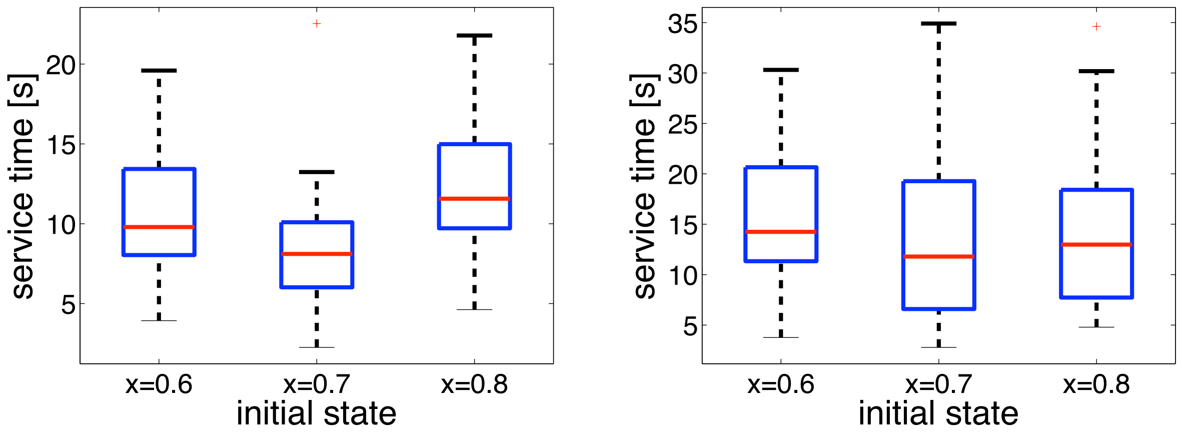


Fig. 10. Box plot for comparison of service times with different initial states for subject numbers 25 (left) and 13 (right).

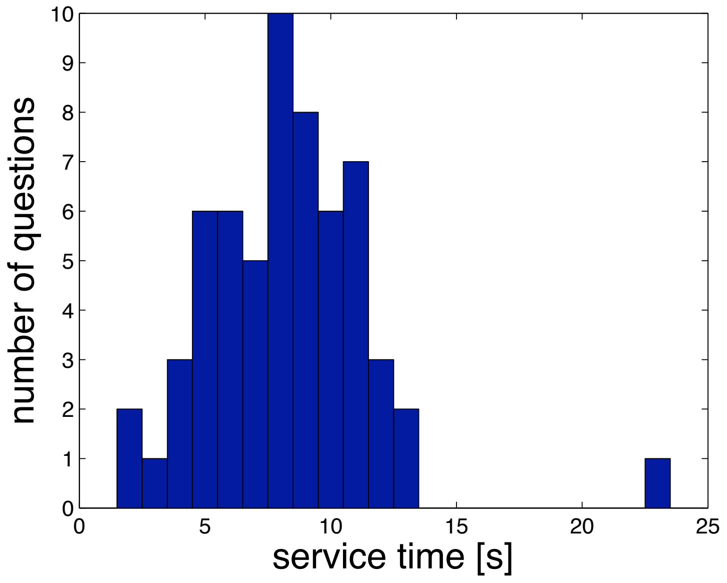


Fig. 11. Histogram for service time distribution with initial state $x = 0.7$ for subject number 25.

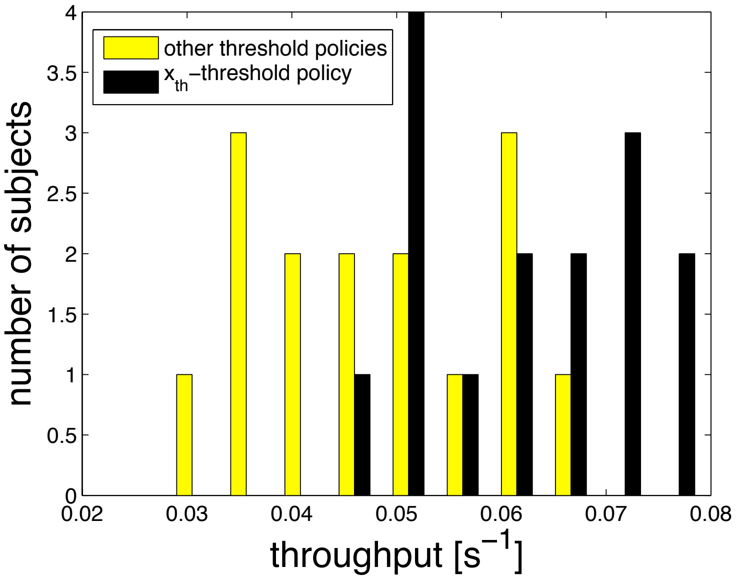


Fig. 12. Comparison between the histograms of the empirical throughput under other (i.e., 0.6 and 0.8) threshold policies on one hand and the $x_{\rm th}$-threshold policy on the other hand.

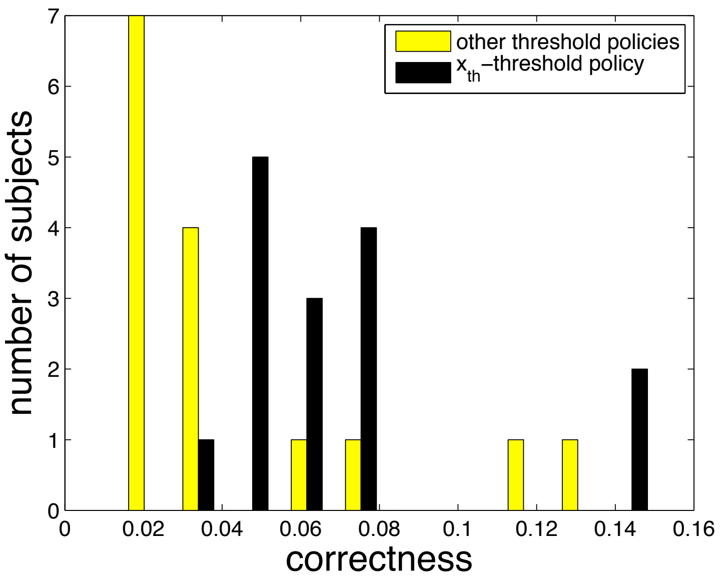


Fig. 13. Comparison between the histograms of the correctness under other (i.e., 0.6 and 0.8) threshold policies on one hand and the $x_{\rm th}$-threshold policy on the other hand.