2015 5th Australian Control Conference (AUCC)
November 5-6, 2015. Gold Coast, Australia

The Challenge of Stabilizing Control for Queueing Systems with Unobservable Server States

Yoni Nazarathy*‡, Thomas Taimre*, Azam Asanjarani*, Julia Kuhn*†, Brendan Patch*†, and Aapeli Vuorinen*.
*School of Mathematics and Physics, The University of Queensland, Australia.
†Korteweg-de Vries Institute for Mathematics, University of Amsterdam, The Netherlands.
‡Email: y.nazarathy@uq.edu.au

Abstract— **We address the problem of stabilizing control for complex queueing systems where servers follow unobservable Markovian environments. The controller needs to assign servers to queues without full information about the servers' states. A control challenge is to devise a policy that matches servers to queues in a way that takes state estimates into account and updates these estimates in the best way possible. Maximally attainable stability regions are non-trivial.**

We present the model, the control problem, and some preliminary methods for analysis and control. We illustrate basic phenomena and then focus on the simplest possible model having a single queue, a fixed state server, and a two state server. For this case, we begin analysis of a partially observable Markov decision process (POMDP) hinting at some structural properties. We also show how to use a quasi-birth–death (QBD) process for analysis and control.

I. Introduction

Methods for design and analysis of stabilizing controllers for queueing systems are widely studied with respect to applications in telecommunications, engineering, and operations research (e.g. [13]). The main theme is the sequential allocation of resources or servers (e.g. communication channels, transmitters, manufacturing machines) in an efficient manner to units requiring processing (e.g. file transfers, widgets). A primary goal, referred to as stability, is to ensure the quantity of units requiring processing remains finite.

In this paper we take first steps towards the study of a multi-server multi-queue system where the servers' states vary in a Markovian fashion and are **not explicitly observed**. Such a scenario has only been partially addressed in the literature. In [18] the stability region of a single-server multi-queue system with a fully observed server state that varies in an i.i.d. fashion is studied. In [6] this work is extended in the multi-server multi-queue context to the case where queue lengths and server states are infrequently observed. More recently, in [3] the stability region of a multi-server multi-queue fully observed system was characterised using a finite set of linear inequalities. In [10] a single-queue (of infinite size) multi-server system similar to ours was studied. Other related work arises, for example, in the context of (nearly) optimal opportunistic spectrum access in cognitive radio networks. Such problems have been considered in [11], [14], [15]; see also the survey [1] and the references therein. However, none of these models capture the same phenomena which we illustrate in this paper.

A key theme to consider is *exploration vs. exploitation*. A sensible controller must strike a trade-off between exploring servers about which there is little information and exploiting servers that are believed to be in a good state; see for example [8] for a survey. Devising such controllers is not a straightforward task. Our goal in this paper is to introduce some methodology that can potentially be used for the design and analysis of such controllers. We introduce a general model for which we illustrate some key phenomena; we also specialize to the simplest non-trivial model and analyze it in further detail using a partially observable Markov decision process (POMDP) approach [17] so as to obtain the maximal stability region. For this simple model, we also discuss how to analyze controller behavior with a finite-state controller modeled as a quasi-birth–death (QBD) process [9]. This way, one can further analyze and optimize controllers.

The remainder of this paper is structured as follows. In Section II we introduce our general system model as well as the simplest non-trivial model. In Section III we show key phenomena using simulation. In Section IV we introduce the POMDP associated with this model and hint on some structural properties. Then in Section V we show how a finite state controller (possibly approximating the POMDP based controller) can be analyzed using matrix analytic Markov methods. We conclude with an outlook in Section VI.

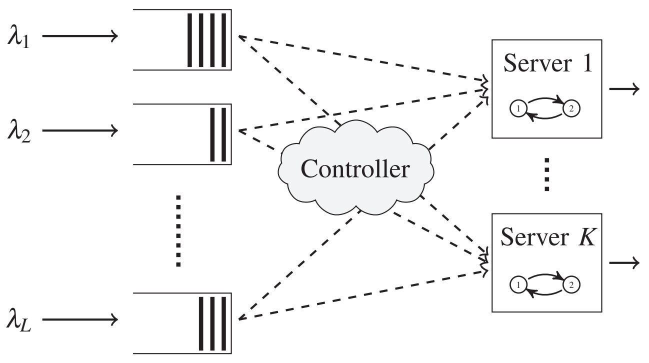

Fig. 1. A controller allocates K servers to L queues. Server states vary according to two state Markov chains and influence the service success probabilities. Server states are not directly observable.

II. System Model

Consider a situation as depicted in Fig. 1. Jobs arrive into one of L queues and are potentially served by one of K servers, according to some control policy. In each discrete time step $t \in \{0, 1, \dots\}$ the probability of a job arriving to

Queue i is λ_i, independent of all other events. After the potential arrivals, the controller allocates each non-empty queue a server at which to attempt service. Service of a job from Queue i by Server j at time t succeeds with a probability $\mu_{i,j}\big(X_j(t)\big)$ that depends only on the state of the server, $X_j(t)$. The latter evolves on $\{1,2\}$ according to a Markov chain, independently of all other events.

The two-state Markov chain is sometimes referred to as a Gilbert–Elliot (GE) Channel [16] having a transition matrix:

$$P^j_{\mathrm{GE}} = \begin{bmatrix} \overline{p} & p \\ q & \overline{q} \end{bmatrix} = \begin{bmatrix} 1-\gamma\overline{\rho} & \gamma\overline{\rho} \\ \overline{\gamma}\overline{\rho} & 1-\overline{\gamma}\overline{\rho} \end{bmatrix},$$

where we denote $\overline{x} := 1-x$. A standard parametrization of this Markov chain uses transition probabilities $p,q \in [0,1]$. Alternatively we may specify the stationary probability of being in state 2, denoted by $\gamma \in [0,1]$, together with the second eigenvalue of P^j_{GE}, denoted by $\rho \in \big[1-\min(\gamma^{-1},\overline{\gamma}^{-1}),\ 1\big]$. Then ρ quantifies the time-dependence of the chain — when $\rho = 0$ the chain is i.i.d., otherwise there is memory. The relationship between these parameterisations is given by $p = \gamma\overline{\rho}$, $q = \overline{\gamma}\overline{\rho}$, $\gamma = p/(p+q)$, and $\rho = 1-p-q$.

The key feature of our model is that the controller **does not** generally observe $X_j(t)$ directly. Instead, the controller is only aware whether a failure or a success of the service has taken place. This is represented by a random variable $I_j(t)$ the realization of which is equal to 0 in case of failure and equal to 1 in case of success. If at time t no queue was allocated to server j, then $I_j(t)$ is not available to the controller. The matter of stabilizing control when X_j is directly observed has been studied in [3]; however, their analysis does not apply to our case where server states are not observed explicitly.

In this paper, we call a control policy *stabilizing* if the associated Markov chain of the system (including the queues) is positive recurrent. For example, if $L \cdot \max_i \lambda_i < K \cdot \min_{i,j} \mu_{i,j}$, then any control policy that arbitrarily allocates servers to queues is stabilizing. But in general, as we illustrate in the paper, stability regions are non-trivial.

A key component used by the controller is the *belief state* for server j. For the two state Markov chain, this is denoted by $\omega_j(t) = \mathbb{P}\big(X_j(t) = 2\,|\,\text{Prior knowledge to time } t\big)$. In our case, the prior knowledge relevant for decision making is given by the sequence of random variables $I_j(t)$ for times t during which server j was selected. As we describe now, it is a simple matter to recursively update this sequence in a Bayesian manner. Given that $\omega_j(t) = \omega$, the believed chance of having success from the queue–server pair i,j at time t is $r_{i,j}(\omega) := \overline{\omega}\mu_{i,j}(1) + \omega\mu_{i,j}(2)$. Now it is easy to construct τ^j, τ^j_0, τ^j_1 which denote the belief updating operators given no observation, observation with $I_j = 0$, or observation with $I_j = 1$, respectively. Omitting subscripts i and j, and denoting $\mu_\ell = \mu_{i,j}(\ell)$ for $\ell = 1,2$, these operators are:

$$\tau(\omega) = \overline{q}\omega + p\overline{\omega} = \omega\rho + \gamma(1-\rho),$$
$$\tau_0(\omega) = \frac{\overline{q}\overline{\mu}_2\omega + p\overline{\mu}_1\overline{\omega}}{\overline{r}(\omega)}, \qquad \tau_1(\omega) = \frac{\overline{q}\mu_2\omega + p\mu_1\overline{\omega}}{r(\omega)}.$$

Observe that the fixed point of τ is the stationary probability γ. The fixed points of τ_0 and τ_1 are also of interest. When $\rho \neq 0$ and $\mu_1 \neq \mu_2$, τ_0 and τ_1 are (real) hyperbolic Möbius transformations of the form $(a\omega+b)/(c\omega+d)$ for $\omega \in [0,1]$. As such, they each have two distinct fixed points, one stable and one unstable. Here, excluding trivialities where $p,q \in \{0,1\}$, the stable fixed point of each lies in $(0,1)$ and is of the form $\big(a-d+\sqrt{(a-d)^2+4bc}\big)/2c$ (see also [12, Lemmas 2, 3]). For τ_0 we have $a = \overline{q}\overline{\mu}_2 - p\overline{\mu}_1$, $b = p\overline{\mu}_1$, $c = \overline{\mu}_2 - \overline{\mu}_1$, and $d = \overline{\mu}_1$; similarly for τ_1 with $\overline{\mu}_\ell$ replaced by μ_ℓ. Denote by ω_ℓ the stable fixed point of τ_ℓ.

Define now for the queue–server pair, i,j, the interval

$$\Omega = \Omega_{i,j} = [\min(\omega_0^{i,j},\omega_1^{i,j}),\ \max(\omega_0^{i,j},\omega_1^{i,j})] \subset [0,1].$$

where the fixed points $\omega_\ell^{i,j}$ are the stable fixed points of τ_ℓ for $\ell \in \{0,1\}$ using the parameters $\mu_{i,j}$ and P^j_{GE}.

Proposition 1: Apply any arbitrary infinite sequence of the mappings τ, τ_0, τ_1 to $\omega \in [0,1]$. Then the limit of any subsequence lies within Ω.
To see this we note that for any ω (and excluding trivialities by ensuring $\rho \neq 0$, and $p,q \notin \{0,1\}$) τ_ℓ is a contraction mapping on $[0,1]$ for $\ell \in \{0,1\}$. Moreover, $\gamma \in \Omega$.

Proposition 1 gives us a range Ω in which the belief states are ultimately contained. This range is useful for finding a corresponding range within which the throughput lies. We denote the *throughput range* of the queue–server pair i, j by $\Omega^\mu_{i,j} = \{r_{i,j}(\omega)\ :\ \omega \in \Omega_{i,j}\}$. Knowledge of $\Omega^\mu_{i,j}$ is useful for obtaining rough rules for maximally stabilizing policies.

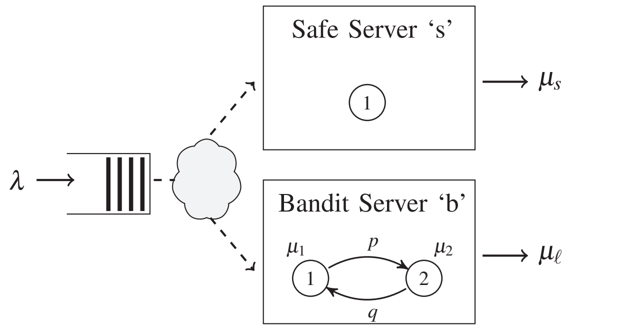

Fig. 2. The simplest (specialized) queueing system analyzed throughout this paper with the exception of Section III.

Consider the simplest non-trivial example of our model as in Fig. 2. In this case, there is only one queue (hence we omit the subscript i) and the server is either $j =$‘s’, with guaranteed success probability μ_s (a ‘Safe Server’) or $j =$‘b’, with state evolving according to P_{GE} (omitting the superscript j for readability; a ‘Bandit Server’, in reference to the GE Channel being a restless two-armed bandit). The parameters of this model are μ_s, μ_1, μ_2, and the two parameters of P_{GE} as well as the arrival rate λ. Assume w.l.o.g. that $\mu_1 < \mu_2$.

The control problem is non-trivial when $\mu_1 < \mu_s < \mu_2$. This can be seen as follows. The throughput range for ‘s’ is trivially $\Omega^\mu_{1,s} = \{\mu_s\}$ but $\Omega^\mu_{1,b}$ is a range that depends on μ_1, μ_2, and P_{GE}. When $\Omega^\mu_{1,b}$ lies entirely below μ_s, any sensible control policy will only choose 's'. Similarly when $\Omega^{\mu}_{1,b}$ lies entirely above μ_s any policy will only choose 'b'.

The interesting case is when $\mu_s \in \Omega^{\mu}_{1,b}$. In this case, it is plausible that a stabilizing policy will utilize the channel history and sometimes choose 'b' (hoping to get μ_2 and gain further information about the bandit) and sometimes choose 's' (believing that a choice of 'b' would yield μ_1 and should thus be avoided).

As an example, consider $\mu_1 = 0.2$, $\mu_s = 0.5$, and $\mu_2 = 0.8$. Note that for these values, if $\gamma = 0.5$, without further information, both servers are equally attractive. If $\rho = 0.4$ it turns out that for $\gamma \notin [0.3, 0.7]$ any sensible policy is degenerate (it always chooses 's' or always chooses 'b'). In this degenerate situation, the stability region is either $\lambda < \mu_s$ (if the policy always chooses 's') or $\lambda < r(\gamma)$ (if the policy always chooses 'b'). However, for $\gamma \in [0.3, 0.7]$ it is possible that the stability region is increased to $\lambda < \mu^*$ with $\mu^* > \max\big(\mu_s, r(\gamma)\big)$. For example for $\gamma = 0.5$ we find that $\mu^* = 0.5212$.

We elaborate more on this simple example in the sections that follow but first we return to the more general model and illustrate some further phenomena that may occur.

III. Some Key Phenomena

To illustrate some key behaviors exhibited by our model consider an example with $L = 2$ and $K = 3$ where, for all three servers, P^j_{GE} is set with $\gamma = 0.5$ and we vary ρ. Further, for each server j,

$$\mu_{1,j}(1) = 0.95\,,\ \mu_{1,j}(2) = 0.05\,,\ \mu_{2,j}(1) = 0.4\,,\ \mu_{2,j}(2) = 0.6\,.$$

We also fix $\lambda_2 = 0.525$ and vary λ_1.

To develop some intuition for the behavior of the system under these parameters, first assume that $\rho = 0$ and further set $\lambda_1 = 0$. In this case the system cannot be stabilized since $\omega_j(t) = \gamma$ for all t and

$$\mu^* = r_{2,j}(\gamma) = 0.5 < 0.525 = \lambda_2\,.$$

That is, since there is no memory, the controller cannot use any past information to choose a good allocation.

At the other extreme, assume momentarily the controller is able to view the states of the servers. In this case an optimal selection may always be made. Now since the long term proportion of servers which are in state 2 follows a binomial distribution with parameters 3 and 0.5, the long term proportion of time during which a good selection is not possible (only $\mu_{2,j}(1)$ is available for all j) is $1/8$ and the complement of that is the long term proportion of time during which a good selection is possible ($\mu_{2,j}(2)$ is available for some j). Hence in this case

$$\mu^* = 0.4/8 + 0.6 \times 7/8 = 0.575 > 0.525 = \lambda_2\,.$$

That is, with full information the system can be stabilised. Even though in practice the controller is not able to view the states of the servers, when $|\rho| \to 1$ the controller behaves as though this were the case since the server state changes sufficiently slowly that highly accurate state estimates can be found. The above discussion hints that the stability region depends on the switching speed of the server states. That is, fast changing server states can sometimes not be stabilized while slowly changing server states can be stabilized.

We now introduce another phenomenon associated with Queue 1 and set $\lambda_1 > 0$. Attempting service of jobs from this queue is more informative about server states than attempting service of jobs from Queue 2 since the failure and success probabilities are much closer to 0 and 1 respectively. Hence, by occasionally allocating this queue to a server the controller may be able to improve the quality of available information and ultimately make better choices.

We illustrate these behaviors through Monte Carlo simulation experiments. As a policy, we use an adaptation of Max-Weight (see [7]). At each time slot each queue is allocated to a unique server $j \in \{1,2,3\}$ so as to maximise $\mathbb{E}\left[\sum_{i=1}^{2} \mu_{i,j}(X_j(t))\,Q_i(t)\right]$, where $Q_i(t)$ is the size of queue $i \in \{1,2\}$ and the expected values are based on state estimates as described in the previous section. This is by no means a maximally stabilizing policy (finding such a policy remains an open question), yet by performing simulations using this policy we illustrate the key phenomena described above.

When performing Monte Carlo simulation of stability, a crucial quantity to examine (estimated from simulation runs) is the *drift* $\lim_{t\to\infty} Q_i(t)/t$. The latter is 0 (almost surely) for stable or critical systems and is positive otherwise. In this example, using a binomial distribution argument as above and denoting $[x]^+ = \max(x, 0)$, we believe that for $i = 1$ the drift is

$$[\lambda_1 - \frac{1}{8}(\mu_{1,j}(1) + \mu_{1,j}(2)) - \frac{6}{8}(\alpha_1\mu_{1,j}(1) + \overline{\alpha}_1\mu_{1,j}(2))]^+$$
$$= \left[\lambda_1 - \frac{13}{80}\alpha_1 - \frac{27}{40}\right]^+,$$

where α_i is the long run proportion of time that Queue i attempts service at a server in state i ("good state" for that queue) out of the total time that some of the servers are in different states. A similar expression holds for the drift of Queue 2, and upon combining these expressions we obtain:

$$\lim_{t\to\infty} \frac{Q_1(t) + Q_2(t)}{t} = \left[\lambda_1 - \frac{13}{80}\alpha_1 - \frac{27}{40}\right]^+ + \left[\frac{1}{10} - \frac{3}{20}\alpha_2\right]^+.$$

Note that if we employ a random policy then $\alpha_i = 1/2$ and if we can observe server states then $\alpha_i = 1$. In our Max-Weight based policy, α_i is not available analytically but is directly tied to stability through the equation above. In fact, through simulation experiments we verified the stated relationship between α_i and the drift in this equation for $\rho \in \{-0.99, -0.98, \ldots, 0.99\}$ and $\lambda_1 = 0.1, 0.45$.

For $\lambda_1 < 0.6750$ we believe the total drift is $[0.1 - 0.15\,\alpha_2]^+$, all of which is contributed from Queue 2, regardless of what α_1 is. This implies that we need $\alpha_2 > 2/3$ for stability. Figure 3 (a) shows that for $\rho = 0$ and $\lambda_1 = 0.1, 0.45$ the Max-Weight based policy appears to obtain $\alpha_2 = 1/2$, identical to the random policy. As $|\rho| \to 1$, however, we see that the policy starts to behave as though it is fully observing the states, obtaining an α_2 closer to 1. As expected, for $\lambda_1 = 0.45$ this occurs faster than for $\lambda_1 = 0.1$. For example,

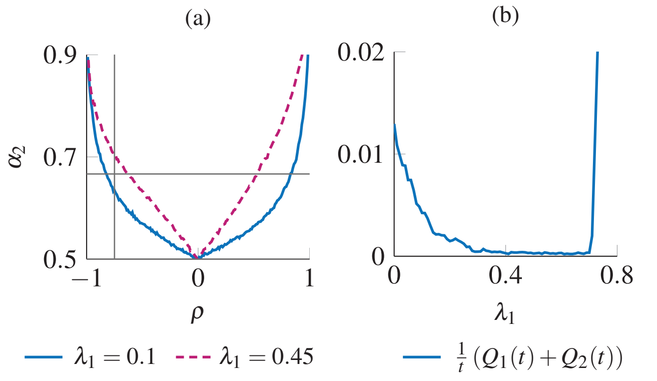

Fig. 3. (a) Estimated α_2 for $\rho \in (-1,1)$ with $\lambda_1 = 0.45$ (dashed) or $\lambda_1 = 0.1$ (solid); (b) Estimated drift for $\lambda_1 \in (0,0.8)$ with $\rho = -0.75$, $t = 200{,}000$.

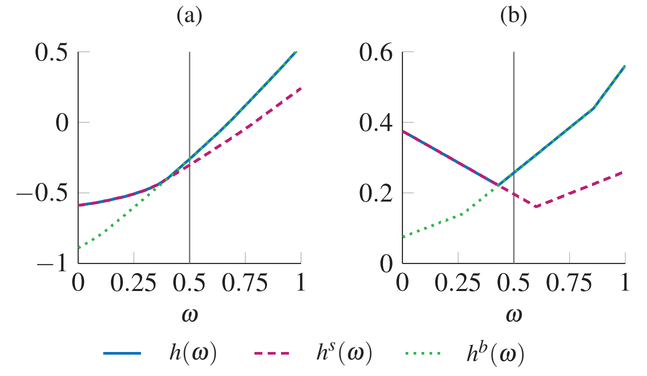

Fig. 4. Relative value functions with $\gamma = 0.5$ and $\mu_s = 0.5$, $\mu_1 = 0.2$, $\mu_2 = 0.8$. For (a) $\rho = 0.7$ (positively correlated), for (b) $\rho = -0.7$ (negatively correlated). The threshold of the myopic policy is indicated by the vertical line.

for $\rho = -0.75$ we obtain $\alpha_2 < 2/3$ when $\lambda_1 = 0.1$ and $\alpha_2 > 2/3$ for $\lambda_1 = 0.45$, thus suggesting that for $\rho = -0.75$ there exists a critical value of λ_1 for which stability occurs. This is supported by Fig. 3 (b), where we see that the drift is only estimated to be 0 for midrange values of λ_1. For high values of λ_1 there is a large amount of drift contributed from Queue 1. Of interest is the drift contributed from Queue 2 for low values of λ_1 — it is not commonly found that a system becomes more stable when some of the arrival rates are increased.

IV. Maximally Stabilizing Control for the Simplest Example

Having shown some of the phenomena that may occur in our system model, we now return to focus on the simpler example of Fig. 2. It is plausible that this system is stable only when $\lambda < \mu^*$, where μ^* is the throughput attained by a system without a queue but rather with an infinite supply of jobs. Hence the maximal stability region (the largest set of λ which can be stabilized) is dictated by the aforementioned maximal throughput μ^*. In contrast with the example of the previous section, the queues do not play a key role here.

Such a throughput optimizing problem has been studied for the degenerate case in which $\mu_1 = 0$ and $\mu_2 = 1$ (observation of I yields direct observation of X). This degenerate case is the so called one-armed subsidy problem associated with the Whittle index [19]. For this problem, [11] have shown that an optimal policy is a threshold policy where 's' is selected when $\omega(t) < \omega^*$, and otherwise 'b' is selected. This motivates the question whether our problem may also be solved by a threshold policy.

It is well-known that the optimal policy that maximizes the throughput follows from the average reward Bellman equation [4], $\mu^* + h(\omega) = \max\left\{h^s(\omega), h^b(\omega)\right\}$, where h is the relative value function, and

$$h^s(\omega) := \mu + h\big(\tau(\omega)\big),$$
$$h^b(\omega) := r(\omega) + \Big[r(\omega)\,h\big(\tau_0(\omega)\big) + \bar{r}(\omega)\,h\big(\tau_1(\omega)\big)\Big].$$

The optimal decision is then to choose 's' if and only if $h^s(\omega) \geq h^b(\omega)$. We can approximately compute h numerically by relative value iteration, on a grid on $[0,1]$. Extensive numerical experiments with a variety of different parameter choices suggest that the optimal policy is indeed threshold in nature (for an example see Fig. 4).

We now comment on why this should be true. Note that $\tau(0) = \tau_0(0) = \tau_1(0) = p$ and thus,

$$\mu^* + h(0) = \max\Big\{\mu_s + h(p),\, \mu_1 + \big[\bar{\mu}_1 h(p) + \mu_1 h(p)\big]\Big\};$$

implying that 's' is optimal. Similarly, we have $\tau(1) = \tau_0(1) = \tau_1(1) = \bar{q}$, whence 'b' is optimal. It is intuitive that h^b increases faster in ω than h^s — the larger ω the more weight is given to μ_2. (This is confirmed by Fig. 4; in fact it turns out that generally h^s is even decreasing for small ω when $\rho < 0$.) Then, because $h^s(0) > h^b(0)$ and $h^s(1) < h^b(1)$, there must be a single point (or interval) of intersection of h^b and h^s. Then (the smallest) ω such that $h^b(\omega) = h^s(\omega)$ is the threshold ω^*.

Proving that a threshold policy is optimal for similar problems has been of great interest in the stochastic optimal control community because it simplifies the finding of an approximate or even exact representation of the optimal policy. Besides [11] relevant work was done in [12], where a Markovian search problem is considered. The problem is of similar nature because the updating rules for the underlying Markov chain are Möbius transformations similar to ours; consequently, the Bellman equation takes a similar shape. However, the simplifying feature for the search problem is that the search ends once the target has been found, and thus, the random variable I does not play a role. Proving the threshold property remains an open challenge for now. Further related and relevant work is [5].

It is of interest to also consider the *myopic threshold policy* based on threshold $\omega^m = (\mu_s - \mu_1)/(\mu_2 - \mu_1)$, which only maximises the immediate expected reward and gives no priority to exploration. Assuming that the relative value function is convex (as can be seen in Fig. 4 and similar experiments), we establish the following intuitive relation between the optimal and the myopic policy.

Proposition 2: Assume h is convex. Then an optimal policy, π^*, chooses the bandit server no later than the myopic policy, π^m. That is, $\left\{\omega \,|\, \pi^m(\omega) = \text{'b'}\right\} \subseteq \left\{\omega \,|\, \pi^*(\omega) = \text{'b'}\right\}$.

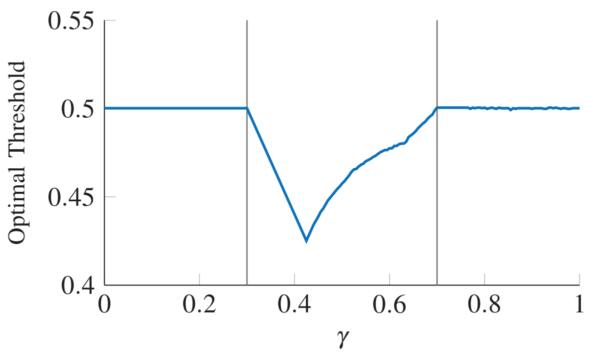

Fig. 5. Optimal threshold values for various values of γ, with $\mu_s = 0.5$, $\mu_1 = 0.2$, $\mu_2 = 0.8$, $p = 0.4$. The region Ω^μ is bounded by the vertical lines.

Proof: By convexity of the relative value function:

$$\begin{aligned}&(\omega\overline{\mu}_2 + \overline{\omega}\,\overline{\mu}_1)\,h\big(\tau_0(\omega)\big) + (\omega\mu_2 + \overline{\omega}\mu_1)\,h\big(\tau_1(\omega)\big)\\&\quad \geq h\left(\overline{q}\overline{\mu}_2\omega + p\overline{\mu}_1\overline{\omega} + \overline{q}\mu_2\omega + p\mu_1\overline{\omega}\right)\\&\quad = h(\omega\overline{q} + \overline{\omega}p) = h\big(\tau(\omega)\big)\,.\end{aligned}$$

This implies that $h^b(\omega) \geq h^s(\omega)$ if $\omega\mu_1 + \overline{\omega}\mu_0 \geq \mu_s$. Thus, the bandit server is the optimal choice for all $\omega > \omega^m$. ■

From Proposition 2 we have that if the optimal policy is indeed threshold in nature, then that threshold is no larger than ω^m. It can further be seen from figures such as Fig. 5 that indeed for $\gamma \in \Omega^\mu$ exploration pays off. Outside of Ω^μ we see that the optimal threshold is ω^m (since the policy is degenerate). We believe that when $\gamma \in \Omega^\mu$ the threshold is never myopic, that is, $\omega^* < \omega^m$.

V. QBD Modeling of a Finite State Controller

We now illustrate how matrix-analytic modeling (MAM) can be used to analyze a finite state controller that approximates an optimal controller. Assume the controller state at time t is $\psi(t)$ and takes values in the finite set $\{1,\ldots,M\}$. The controller action is (potentially) randomized based on a vector of probabilities $\mathbf{c}$ so that 'b' is chosen with probability $c_{\psi(t)}$ and otherwise the choice is 's'. The controller state is updated in a (potentially) randomized manner based on the $M \times M$ stochastic matrices S,P, and Q as follows: if 'b' was not selected (either because there were no jobs in the queue, or because 's' was selected), the distribution of the new state is $\big(S_{\psi(t),1},\ldots,S_{\psi(t),M}\big)$. Similarly, if 'b' was chosen and service was successful ($I = 1$), the distribution of the new state is $\big(P_{\psi(t),1},\ldots,P_{\psi(t),M}\big)$. Finally, if 'b' was chosen and the service failed ($I = 0$), the distribution of the new state is $\big(Q_{\psi(t),1},\ldots,Q_{\psi(t),M}\big)$. That is, the rows of S,P, and Q indicate how to (potentially randomly) choose the next controller state.

Now, given such a controller (S, P, Q, and $\mathbf{c}$), we construct a Markovian model of the system. The state of this model at time t is given by the queue length, bandit state, and controller state as follows:

$$Z(t) = \Big(\underbrace{Q(t)}_{\text{Level}}, \underbrace{\big(\overbrace{B(t)}^{\text{Bandit}}, \overbrace{\psi(t)}^{\text{Controller}}\big)}_{\text{Phase}}\Big) \in \{0,1,\ldots\} \times \{1,2\} \times \{1,\ldots,M\}.$$

When the states are lexicographically ordered, with first component countably infinite (levels) and the other components finite (phases), the resulting transition matrix is of the QBD form (see [9]):

$$\begin{bmatrix} S_0 & S_1 & 0 & \cdots & \cdots & \cdots & \cdots \\ P_{-1} & P_0 & P_1 & 0 & \cdots & \cdots & \cdots \\ 0 & P_{-1} & P_0 & P_1 & 0 & \cdots & \cdots \\ \vdots & 0 & P_{-1} & P_0 & P_1 & 0 & \cdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \ddots \end{bmatrix}, \qquad (1)$$

where S_0, S_1, P_{-1}, P_0, and P_1 are blocks of size $2M \times 2M$ as we construct now. For $\ell \in \{1,2\}$, denote

$$\begin{aligned}\widetilde{P}_\ell &= \mu_\ell\,\mathrm{diag}(\mathbf{c})P + \mu_s\,\mathrm{diag}(\overline{\mathbf{c}})S\,,\\ \widetilde{Q}_\ell &= \overline{\mu}_\ell\,\mathrm{diag}(\mathbf{c})Q + \overline{\mu}_s\,\mathrm{diag}(\overline{\mathbf{c}})S\,.\end{aligned}$$

Here $\widetilde{P}_\ell(i,j)$ is the probability of updating controller state from i to j when the server state is ℓ and transmission was successful. Similarly, $\widetilde{Q}_\ell(i,j)$ is the probability of doing so when transmission was unsuccessful. Recall that in both cases 'b' is used for service with probability c_i, and otherwise 's' is used. Now define the $2M \times 2M$ matrices,

$$\widetilde{P} = \begin{bmatrix} \overline{p}\widetilde{P}_1 & p\widetilde{P}_1 \\ q\widetilde{P}_2 & \overline{q}\widetilde{P}_2 \end{bmatrix},\ \widetilde{Q} = \begin{bmatrix} \overline{p}\widetilde{Q}_1 & p\widetilde{Q}_1 \\ q\widetilde{Q}_2 & \overline{q}\widetilde{Q}_2 \end{bmatrix},\ \widetilde{S} = \begin{bmatrix} \overline{p}S & pS \\ qS & \overline{q}S \end{bmatrix}.$$

These are combined to form the basic elements of the QBD transition probability matrix (1) of the (complicated) Markov chain Z as: $S_0 = \overline{\lambda}\widetilde{S}$, $S_1 = \lambda\widetilde{S}$, $P_{-1} = \overline{\lambda}\widetilde{P}$, $P_0 = \overline{\lambda}\widetilde{Q} + \lambda\widetilde{P}$, $P_1 = \lambda\widetilde{Q}$.

The virtue of modeling the system as a QBD is that we can use the body of knowledge and exploit existing algorithms in MAM (e.g. [2]), for analyzing the system and ultimately optimizing controllers. We now briefly outline some useful results for QBDs.

A key object for QBDs is the G-matrix, where the entry G_{ij} is the probability of first hitting phase j in level $n-1$ after starting at phase i in level n. This matrix is the minimal non-negative solution to the matrix equation $G = P_{-1} + P_0G + P_1G^2$. Much of QBD theory, algorithms, and software, deals with solving such a matrix equation, see for instance [2]. Another key related object is the matrix $R = P_1(I - P_0 - P_1G)^{-1}$. Positive recurrence (stability) and the stationary distribution of the QBD is directly linked to R. In particular, the QBD is positive recurrent if and only if the spectral radius of R is less than 1 (Corollary 6.2.4 of [9]). Further, assuming positive recurrence, the stationary distribution is of the so-called matrix-geometric form: $\pi_n = \pi_0R^n$, for $n \geq 0$. Here π_n is a row vector of length $2M$ giving the stationary distribution over the phases when the level is n. Hence solving the matrix quadratic equation for G essentially gives the stationary distribution (the normalizing vector π_0 can also be easily obtained).

A sufficient condition for stability (positive recurrence) is that, $\pi_\infty\big(P_1 - P_{-1}\big)\mathbf{1} < 0$, where π_∞ is the stationary distribution of the (finite) stochastic matrix $P_{-1} + P_0 + P_1$. This is also the stationary distribution of $\widetilde{P} + \widetilde{Q} + \widetilde{S}$ which

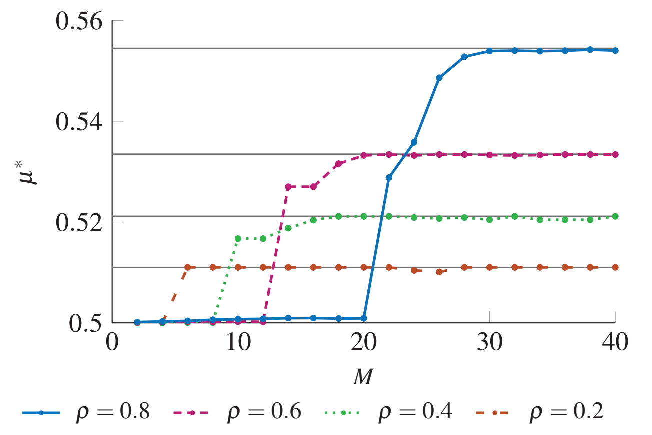

Fig. 6. Stability region achieved by finite state controllers for increasing M. The limiting horizontal lines are at μ^* as computed by means of relative valuate iteration of Section IV.

does not depend on λ. This property of our QBD allows us to represent the stability criterion as,

$$\lambda < \mu^* = \left(\frac{\pi_\infty \widetilde{Q}\mathbf{1}}{\pi_\infty \widetilde{P}\mathbf{1}} + 1 \right)^{-1},$$

with μ^* depending on the controller and system parameters but not depending on λ.

We now treat $\{1,\dots,M\}$ as a discretization of the belief state space Ω and encode discretized versions of the belief state updating operators τ, τ_0, and τ_1 in the matrices S, Q, and P respectively. To ensure irreducibility of the QBD, we add a small ($\varepsilon = 0.001/M$) probability component to each entry of S, Q and P and renormalize. We also set the elements of $\mathbf{c}$ to be $c_i = \mathbb{I}\{\frac{i}{M} \geq \omega^*\}$ (a thresholding), where $\mathbb{I}\{A\}$ is the usual indicator function of an event A.

This structure allows us to maximise μ^* by optimizing ω^* (over the M possibilities) so as to find the maximally stabilising controller with M controller states (and $2M$ QBD states). Indeed, as we show in Fig. 6, as M grows, this controller achieves the same stability region as the maximally stabilising controller of the previous section. In this figure we consider $\mu_1 = 0.2$, $\mu_s = 0.5$, $\mu_2 = 0.8$, and $\gamma = 0.5$, for $\rho = 0.2, 0.4, 0.6$, and 0.8. The optimal μ^* based on relative value iteration of the Bellman equation are 0.5110, 0.5212, 0.5335, and 0.5545, respectively.

The virtue of the QBD based controller is that only M states are needed (as opposed to the continuum $[0,1]$). Further, by exploiting the QBD structure of the discretized system, we may analyze its performance under such a controller. For example, by computing the matrix G for a given arrival rate λ we can examine the steady-state (matrix-geometric) distribution of the queue length based on π_n and may further optimize the controller.

VI. Outlook

We introduced a general model framework for studying control of queueing systems with partial observations of Markovian environments. Even the simplest possible model in our framework has non-trivial behavior which can potentially be rigorously analyzed in a POMDP setting. Proving structural properties remains an open challenge. In tackling the problem using a Markovian modeling framework, we illustrated the usefulness of the QBD process viewpoint.

For more complicated examples we showed through simulation that several interesting phenomena may occur. Determining the maximally stabilizing region and policy remains an open question in general. This preliminary work has illustrated some of the interesting behavior that arises when studying questions of stability and control for our model, the answers to which remain fruitful topics for further research.

Acknowledgement

YN is supported by Australian Research Council (ARC) grants DP130100156 and DE130100291. AA is supported by DE130100291. JK is supported by DP130100156.

References

[1] A. Asadi and V. Mancuso, A survey on opportunistic scheduling in wireless communications. IEEE Commun. Surveys Tuts., 15(4):1671–1688, 2013.

[2] D. A. Bini, B. Meini, S. Steffé, and B. Van Houdt, Structured Markov chains solver: software tools. Proceeding from SMCTools, 14, 2006.

[3] H. Halabian, I. Lambadaris, and C. H. Lung, Explicit characterization of stability region for stationary multi-queue multi-server systems. IEEE Trans. Autom. Control, 59(2):355–370, 2014.

[4] O. Hernández-Lerma and J. B. Lasserre, *Discrete-time Markov Control Processes: Basic Optimality Criteria*. Springer, 2012.

[5] L. Johnston and V. Krishnamurthy, Opportunistic file transfer over a fading channel: A POMDP search theory formulation with optimal threshold policies. IEEE Trans. Wireless Commun., 5(2):394–405, 2006.

[6] K. Kar, X. Luo and S. Sarkar, Throughput-optimal scheduling in multichannel access point networks under infrequent channel measurements. IEEE Trans. Wireless Commun., 7(7):2619–2629, 2008.

[7] F. Kelly and E. Yudovina, *Stochastic Networks*. Cambridge, 2014.

[8] J. Kuhn and Y. Nazarathy, *Wireless Channel Selection with Reward-Observing Restless Multi-armed Bandits*, draft book chapter, submitted, 2015.

[9] G. Latouche and V. Ramaswami, *Introduction to Matrix Analytic Methods in Stochastic Modeling*. SIAM, 1999.

[10] C. P. Li and M. J. Neely, Network utility maximization over partially observable Markovian channels. Performance Evaluation, 70(7):528–548, 2013.

[11] K. Liu and Q. Zhao, Indexability of restless bandit problems and optimality of Whittle index for dynamic multichannel access. IEEE Trans. Inf. Theory, 56(11):5547–5567, 2010.

[12] I. M. MacPhee and B. P. Jordan, Optimal search for a moving target. Probab. Eng. Inform. Sc., 9(2):159–182, 1995.

[13] B. M. Miller, Optimization of queuing system via stochastic control. Automatica, 45(6):1423–1430, 2009.

[14] J. Niño-Mora, A restless bandit marginal productivity index for opportunistic spectrum access with sensing errors, in *Network Control and Optimization*, R. Núñez-Queija and J. Resing, Eds. Springer, 2009, pp. 60–74.

[15] W. Ouyang, A. Eryilmaz, and N. B. Shroff, Asymptotically optimal downlink scheduling over Markovian fading channels. INFOCOM, 2012 Proceedings IEEE, pp. 1224–1232, 2012.

[16] P. Sadeghi, R. A. Kennedy, P. B. Rapajic, and R. Shams, Finite-state Markov modeling of fading channels: A survey of principles and applications. IEEE Signal Process. Mag., 25(5):57–80, 2008.

[17] R. D. Smallwood and E. J. Sondik, The optimal control of partially observable Markov processes over a finite horizon. Operations Research, 21(5):1071–1088, 1973.

[18] L. Tassiulas and A. Ephremides, Dynamic server allocation to parallel queues with randomly varying connectivity. IEEE Trans. Inf. Theory, 39(2):466–478, 1993.

[19] P. Whittle, Restless bandits: Activity allocation in a changing world. Journal of Applied Probability, 25:287–298, 1988.