Customers routing in a queueing system with two heterogeneous servers in speed and in quality of resolution

Benjamin Legros • Oualid Jouini

Laboratoire Genie Industriel, CentraleSupélec, Université Paris-Saclay, Grande Voie des Vignes, 92290 Chatenay-Malabry, France

benjamin.legros@centraliens.net • oualid.jouini@centralesupelec.fr

December 18, 2015

Abstract

Heterogeneous servers, in manufacturing and service systems, may have different speeds and different quality levels for the provided service or good. For a two-server queueing model, we formulate the routing problem of minimizing a weighted sum of the expected time in the system and the throughput of unsatisfied customers. Using a Markov decision process approach, we prove that the optimal routing of customers in service is of threshold type on the number of customers in the queue. This result is an extension of the known one when only the heterogeneity in speed of the servers is considered.

Keywords: Dynamic programming; optimal routing; queueing systems; Markov decision process; threshold policy; heterogeneous servers; quality of resolution.

1 Introduction

The operation speed usually interacts with the quality of the provided service. In some cases, a high speed means a hurry and not enough attention, which leads to a poor quality. In other cases, high speed may be related to a well trained and experienced human capacity, which implies high quality. Examples with speed-quality interaction include healthcare systems where the treatment length interacts with the health deterioration level after the treatment, call centers where the call conversation duration is correlated to the call resolution probability, make-to-order manufacturing companies with customized products, high-end restaurants, just to name a few.

We consider the problem of dynamically and optimally controlling a queueing system with two heterogeneous servers in speed and in quality of service. Managers are then concerned at the same time by the customer waiting time and the quality of the provided service. In what follows, we review the literature related to this paper.

The slow server problem. This work is most closely related to the slow server problem literature, which has a long history. Although a non-idling policy with priority for the faster server
minimizes waiting times, it can not be optimal for sojourn times. Krishnamoorthi (1963) was the first to consider the slow server problem and showed the optimal policy in the two-server case under elementary assumptions. He showed that the fast server should be always used, and the slow server should be only used when the fast server is busy and the number of customers waiting in the queue exceeds a given threshold. Next, the slow server problem has been considered rigorously using Markov decision process (MDP) approaches or sample path arguments. The rigorous proof of the optimal policy can be found in Larsen and Agrawala (1983), Lin and Kumar (1984) and Walrand (1984). Using value iteration, Koole (1995) provides a simpler version of the proof of the optimal threshold policy. Viniotis and Ephremides (1988) consider various extensions of the result, for example, for the case of Erlang servers. Rykov and Efrosinin (2009) also extend the proof of the optimality of a threshold policy including service costs.

Results concerning the optimal policy for more than two servers are much more challenging to obtain. The growing dimensionality of the underlying state space accounts for the difficulty. Weber (1993) uses a coupling argument to show that whenever a job is routed, it should always be routed to the fastest available server; but he only provides a conjecture that the optimal routing follows a state-dependent threshold policy. Two papers claim having proved the optimality of the state-dependent threshold policy. The first one, Rykov (2001), uses value iteration to show that the optimal value function satisfies monotonicity properties. The second one, Luh and Viniotis (2002) uses a linear programming formulation and sample path analysis. However, de Véricourt and Zhou (2006) proved the incompleteness of the proofs provided in these two papers.

Other papers have considered the search for a good routing heuristic through performance analysis. In the two-server case, Rubinovitch (1985) compares different non-idling queueing systems so as to determine whether or not the slow servers should be used in order to minimize the expected sojourn time in the system. He shows that under a threshold on the traffic intensity, the slow server should be removed. Cabral (2005) extends this result to the multi-server case with uninformed customers. Some researchers have also investigated the problem under various asymptotic regimes (Teh and Ward, 2002; Armony, 2005; Atar, 2008; Atar and Shwartz, 2008; Armony and Ward, 2010).

Researchers have addressed the extension of the slow server problem to queueing systems with unreliable servers. Ozkan and Kharoufeh (2014) differentiate the two servers by their service rates and reliability attributes. The slower server is perfectly reliable while the faster server is subject to random failures. The objective is to minimize the long-run average number of customers in the system. Using a Markov decision process approach, the authors prove that it is always optimal to route customers to the fast server when it is available, irrespective of its failure and repair rates.
For the slow server, the optimal policy is a threshold policy that depends on the queue length. Other related references include Efrosinin (2013); Ozkan and Kharoufeh (2015).

The slow server problem with quality of resolution. Despite its prevalence in practice, the literature has rarely addressed the slow server routing problem by including service quality related factors. Two exceptions, belonging to the call center operations management literature, are de Véricourt and Zhou (2005), and Zhan and Ward (2014). de Véricourt and Zhou (2005) analyze the routing problem in a call center where a customer immediately calls back when her problem is not appropriately resolved using an MDP formulation. The call quality is defined through a call resolution probability, i.e., the probability that the customer is satisfied and does not call back for the same problem. Servers have different call resolution probabilities and different service rates. They address the dynamic control problem under the objective of minimizing the expected total time of call resolution. For the two-server case, they prove that a threshold policy is optimal. A call should be routed to the server with the highest resolution rate (resolution probability times service rate) whenever possible. The other server will be used only when the number of waiting calls in the queue exceeds a certain threshold. Partial characterization of the optimal policy and practical heuristics are given for the multi-server case.

The resolution rate policy is however shown to perform poorly under an objective that involves the callback probability (Mehrotra et al., 2012). Under the asymptotic many server quality and efficiency driven regime, Zhan and Ward (2014) extend the analysis of de Véricourt and Zhou (2005), by considering similar modeling and assumptions, but a more general objective measured as a weighted sum of the expected waiting time and the callback rate. They approximate this asymptotic problem by a diffusion control problem. The efficiency of the analytic diffusion solution is then validated through simulation.

Although callbacks may be an appropriate measure of quality in some contexts such as technical call centers, it is not the case for many other manufacturing and service systems where unsatisfied customers defect rather than comeback to the system. To the contrary to the existing literature, an unsuccessful service in our modeling does not lead to the customer return. Examples include commercial call centers where agents have various selling abilities, or make-to-order manufacturing firms where an unsatisfied customer with a long lead time may switch to competitors (Hall and Porteus, 2000).

Contributions. Using an MDP approach, we address the optimal routing of customers in service for the two-server problem under the objective of minimizing a weighted sum of the expected time
spent in the system and the unsatisfied customer rate. We prove that the optimal routing is of
treshold type based on the number of customers in the queue. Under a given threshold, only one
server should work and the other one remains idle. Above this threshold, both servers should serve
jobs. This is similar to the known result when only the heterogeneity in speed of the servers is
considered.

The complexity added by the quality of resolution is that one server is not necessarily better than
the other one. The value of our analysis in comparison with existing ones is that no assumptions
are made on the preference for one given server. The approach for the proof of the threshold policy
is divided in three steps. In the first step, we prove that the number of busy servers should increase
with the queue size. In the second step, we prove that for the infinite horizon performance, having
the two servers idling is not optimal. Finally in the third step, we prove that there cannot be
changes in the prioritization of one server as a function of the system states.

Structure of the paper. In Section 2, we provide the problem formulation. Using an MDP
approach, we prove in Section 3 that the optimal policy is of threshold type. We finally give a brief
conclusion.

2 Problem Formulation

Consider a queueing system with a single customer type and two parallel servers, servers 1 and 2.
Customers arrive, at a single first come first served (FCFS) infinite queue, according to a Poisson
process with rate $\lambda$. Service times are independent and exponentially distributed with rate $\mu_i$
for server $i$, $i \in \{1, 2\}$. Once server $i$ completes a service, the customer is either satisfied with
probability $1 - \alpha_i$, or unsatisfied with probability $\alpha_i$, $i \in \{1, 2\}$. An unsatisfied customer defects,
and this is considered as a loss of goodwill. To ensure stability, we assume that $\lambda < \mu_1 + \mu_2$. The
stationary performance measures of interest are the customer expected time spent in the system,
denoted by $E(R)$, and the production rate (throughput) of server $i$, denoted by $T_i$, $i \in \{1, 2\}$.

Consider now the set of all non-preemptive non-anticipating FCFS routing policies for the
routing of customers in service. At any point of time, we want to decide for the first customer in
the queue (if any) whether to keep her in the queue, or to serve her by an available server (if any).
We combine two objectives to have a trade-off between minimizing the time spent in the system
and maximizing customer satisfaction about the provided service. Concretely, the goal is to find
the optimal routing policy which minimizes the following weighted sum

$$\alpha_1 T_1 + \alpha_2 T_2 + c R E(R),$$

(1)
where the coefficient $c_R$ ($c_R \geq 0$) translates the relative importance given, by the system manager, to the expected time spent in the system compared to the throughput of unsatisfied customers. Without loss of generality, the cost per unsatisfied customer is 1.

We propose to formulate the routing problem as an MDP and next use the value iteration technique to prove the form of the optimal policy. We formulate the problem via the definition of states, the transition structure and the possible actions. In order to completely separate transitions and actions we allow for idling possibilities, i.e., after an arrival or a service completion there is no automatic routing in service.

**States definition.** Let us denote by $x$ the number of customers in the queue, $x \geq 0$. The state of the servers is described through the symbols 0, 1, 2 and $1/2$. State 0 is a situation where the two servers are idle. State $i$ is a situation where only server $i$ is working, $i \in \{1, 2\}$. State $1/2$ corresponds to a situation where the two servers are working. A state of the system is thus completely defined by the couple $(i, x)$, $i \in \{0, 1, 2, 1/2\}$ and $x \geq 0$.

**Transitions.** We denote the transition rate from state $(i, x)$ to state $(i', x')$ by $q_{(i,x),(i',x')}$. Hence for $i, i' \in \{0, 1, 2, 1/2\}$ and $x, x' \geq 0$, we have

$$q_{(i,x),(i',x')} = \begin{cases} 
\lambda, & \text{if } i' = i, x' = x + 1, \text{ for } i \in \{0, 1, 2, 1/2\}, x \geq 0, \\
\mu_1, & \text{if } i = 1, i' = 0 \text{ or } i = 1/2, i' = 2 \text{ and } x' = x, \text{ for } x \geq 0, \\
\mu_2, & \text{if } i = 2, i' = 0 \text{ or } i = 1/2, i' = 1 \text{ and } x' = x, \text{ for } x \geq 0, \\
0, & \text{otherwise,}
\end{cases}$$

which corresponds to arrivals and service departures.

**Actions.** At each instant of time when at least one customer is in the queue and one server is idling, we are allowed to serve or not this customer and eventually to decide which server to choose. A cost of 1 is counted per unsuccessful service. This may discourage to route a customer automatically to the first available server. On the other hand, waiting customers incur costs. It is therefore important not to postpone too much a start of service. Hence, decisions have to be taken in situations where (i) at least one customer is in the queue and (ii) at least one server is idle. We have to decide:

- How many customers should be routed in service (0, 1 or 2)?
- In the case where only one customer should be served, then which server should be preferred?

Since our model is a continuous-time model, we need to show that uniformization works so as to make this continuous-time problem discrete. In all states where the two servers are busy, there
are three possible events: an arrival or a service completion from server 1 or from server 2. The rate out of each of these states is \( \lambda + \mu_1 + \mu_2 \). Yet, in all states where only one server is busy, there are only two possible events: an arrival or a service completion from the busy server. When the two servers are idling only an arrival can occur. By adding fictitious transitions from a state to itself we allow that the rate out of each state is \( \lambda + \mu_1 + \mu_2 \), without exception, for every policy.

We are considering long-term average costs. It is then optimal to schedule customers only at service completion and arrival times. We consider the embedded discrete-time Markov decision chain by looking at the system only at transition instants. They occur according to a Poisson process with rate \( \lambda + \mu_1 + \mu_2 \). The instantaneous holding costs for the embedded chain count for the whole period until the next transition. If it is optimal to keep a server idle at a given time, then the action remains optimal until the next event in the system. This result follows directly from the continuous-time Bellman equation (Puterman (1994), Chapter 11).

We choose to formulate a 2-step value function, in order to separate transitions and actions and simplify the involved expressions. We define the dynamic programming value functions \( U_n(\ldots) \) and \( V_n(\ldots) \) over \( n \geq 0 \) steps, depending on the state of the system (the first variable is the state of the servers, and the second variable is the number of customers in the queue). We take \( U_0(\ldots) = 0 \) and \( V_0(\ldots) = 0 \). Next, we can express \( V_{n+1}(\ldots) \) in \( V_n(\ldots) \) in the following way. First, the holding costs until the next jump are incurred by the cost function \( c(\ldots) \) defined as \( c(0, x) = \frac{c_R}{\lambda} x \), \( c(i, x) = \frac{c_R}{\lambda} (x + 1) \) for \( i = 1, 2 \) and \( c(1/2, x) = \frac{c_R}{\lambda} (x + 2) \) to account for the time spent in the system by a given customer. A cost of 1 is counted per unsuccessful service. Then one of three events can happen: an arrival with probability \( \frac{\lambda}{\lambda + \mu_1 + \mu_2} \), a departure from server 1 with probability \( \frac{\mu_1}{\lambda + \mu_1 + \mu_2} \) or a departure from server 2 with probability \( \frac{\mu_2}{\lambda + \mu_1 + \mu_2} \). We assume without loss of generality that \( \lambda + \mu_1 + \mu_2 = 1 \), such that the rate out of each state is equal to 1; then we can consider the rates to be transition probabilities. This allows for the convergence of \( V_{n+1}(\ldots) - V_n(\ldots) \) to the minimal long-run average costs as \( n \) tends to infinity (Puterman (1994), Section 8.10).

We have

\[
\begin{align*}
V_{n+1}(0, x) &= \frac{c_R}{\lambda} x + \lambda U_n(0, x + 1) + (\mu_1 + \mu_2) U_n(0, x), \\
V_{n+1}(1, x) &= \frac{c_R}{\lambda} (x + 1) + \lambda U_n(1, x + 1) + (\mu_1 U_n(0, x) + \alpha_1) + (\mu_2 U_n(1, x), \\
V_{n+1}(2, x) &= \frac{c_R}{\lambda} (x + 1) + \lambda U_n(2, x + 1) + (\mu_1 U_n(2, x) + \mu_2 U_n(0, x) + \alpha_2), \\
V_{n+1}(1/2, x) &= \frac{c_R}{\lambda} (x + 2) + \lambda U_n(1/2, x + 1) + (\mu_1 U_n(2, x) + \alpha_1) + (\mu_2 U_n(1, x) + \alpha_2),
\end{align*}
\]
with

\[ U_{n+1}(0, x) = \left\{ \begin{array}{ll} V_{n+1}(0, 0), & \text{for } x = 0 \\ \min(V_{n+1}(0, 1), V_{n+1}(1, 0), V_{n+1}(2, 0)), & \text{for } x = 1 \\ \min(V_{n+1}(0, x), V_{n+1}(1, x-1), V_{n+1}(2, x-1), V_{n+1}(1/2, x-2)), & \text{for } x > 1, \end{array} \right. \]

\[ U_{n+1}(i, x) = \left\{ \begin{array}{ll} V_{n+1}(i, 0), & \text{for } x = 0 \\ \min(V_{n+1}(i, x), V_{n+1}(1/2, x-1)), & \text{for } x > 0, \end{array} \right. \]

for \( i \in \{1, 2\} \), and

\[ U_{n+1}(1/2, x) = V_{n+1}(1/2, x), \]

for \( n, x \geq 0 \).

For each \( n > 0 \) and every state \((i, x) (i \in \{0, 1, 2, 1/2\} \text{ and } x \geq 0)\) there is a minimizing action:
serve two customers, serve a customer with server \( i \) \((i = 1 \text{ or } i = 2)\) or do not serve any customer.

For fixed \( n \) \((n > 0)\) we call this function

\[ \{0, 1, 2, 1/2\} \times \mathbb{N} \rightarrow \{\text{serve 2 customers, serve one customer with server } i, \text{ do not serve}\}, \]

a policy. As \( n \) tends to infinity, this policy converges to the average optimal policy, that is, the policy that minimizes the long-run expected average costs.

**Remark.** We assume in our modeling that a job cannot switch from one server to the other during service. In the opposite case, the optimal policy is known from George and Harrison (2001) since the system can be seen as an adjustable service rate queueing model. Server \( i \) working alone is associated to a cost of \( \alpha_i \mu_i \) and a service rate of \( \mu_i \) \((i = 1, 2)\) and both servers working together corresponds to a cost of \( \alpha_1 \mu_1 + \alpha_2 \mu_2 \) and a service rate of \( \mu_1 + \mu_2 \). The optimal policy would then be a two-thresholds policy. Under a first threshold the server with the lowest product \( \alpha_i \mu_i \) \((i = 1, 2)\) would be working alone, between this first threshold and a second one the server with the highest service rate would be working alone and above this second threshold both servers should work together.

### 3 Optimal Routing

One way of obtaining the long-term average optimal actions is to use the value iteration technique, by recursively evaluating \( V_n \) using Equation (2), for \( n \geq 0 \). In Theorem 1, we prove by induction on the value function that the optimal policy is of threshold type. We prove that if some structural properties defining the threshold structure of the optimal policy are satisfied for \( V_n \) then these properties are also satisfied for \( V_{n+1} \). Hence, we show that these properties hold for every \( n \), and
therefore the long-run average optimal policy is also of threshold type. The proof of the theorem is divided into three steps. They are first commented below.

**Step 1.** The first step consists of proving that the number of server which should be active follows a threshold policy. A threshold policy is characterized by the fact that if serving a customer is optimal in \( x \), then serving a customer is also optimal in \( x + 1 \). Sufficient conditions for this are

\[
V_n(0; x + 1) - V_n(i, x) \geq 0 = \Rightarrow V_n(0; x + 2) - V_n(i, x + 1) \geq 0,
\]

and

\[
V_n(i, x + 1) - V_n(1/2, x) \geq 0 = \Rightarrow V_n(i, x + 2) - V_n(1/2, x + 1) \geq 0,
\]

for \( i = 1, 2 \). These conditions are satisfied if

\[
V_n(0; x + 2) + V_n(i, x) - V_n(0, x + 1) - V_n(i, x + 1) \geq 0,
\]

and

\[
V_n(i, x + 2) + V_n(1/2, x) - V_n(i, x + 1) - V_n(1/2, x + 1) \geq 0,
\]

for \( i = 1, 2 \). The difference compared to Lin and Kumar (1984) or Koole (1995) is that we do not make a specific assumption on which server should be prioritized. The difficulty in the choice for server 1 or server 2 can be seen in the difference

\[
V_{n+1}(1, x) - V_{n+1}(2, x) = \lambda(U_n(1, x + 1) - U_n(2, x + 1)) + \mu_1(U_n(0, x) - U_n(2, x)) + \mu_2(U_n(1, x) - U_n(0, x)) + \mu_1\alpha_1 - \mu_2\alpha_2,
\]

for \( x \geq 0 \). With the last term, one could think of the natural routing control that routes customers in priority to the server that has the lowest unsuccessful service rate (minimum of \( \mu_i\alpha_i \) for \( i \in \{1, 2\} \)). However this simple rule does not propagate through value iterations because of the term \( \mu_1(U_n(0, x) - U_n(2, x)) + \mu_2(U_n(1, x) - U_n(0, x)) \) which can be either positive or negative and further assumptions should be made to determine which server to prioritize.

**Step 2.** This step consists of proving, under an infinite horizon, that having the two servers idling at the same time cannot be optimal, as long as a waiting customer represents a strictly positive cost for the system. However, this statement cannot be proven by induction since both servers idling can be optimal under a finite horizon.

For small \( n \), not serving customers is often optimal: the costs of holding a customer in the queue over a short period can be cheaper than the costs of unsuccessful services. As an example, consider the problem with parameter values \( \lambda = 0.13, \mu_1 = 2, \mu_2 = 5, \alpha_1 = 0.1, \alpha_2 = 1 \) and \( c_R = 0.005 \).
Using Equations (2) for \( n = 5 \), we deduce that it is optimal not to serve any customer.

Since we are considering long-run average performance, it would be tempting to first state that it is not optimal to have the two servers idling at the same time and next rewrite the definition of \( U_n \) such that \( U_n(0, x) = \min(V_n(1, x - 1), V_n(2, x - 1), V_n(1/2, x - 2)) \) for \( x > 1 \) and \( U_n(0, 1) = \min(V_n(1, 0), V_n(2, 0)) \). It would lead to prove less structural properties in the induction step from \( V_n \) to \( U_n \). Yet, this would force the system to choose non-optimal decisions under finite horizon, which breaks some structural properties required in Step 1 to prove the threshold structure. For instance, as shown later in the induction from \( V_n \) to \( U_n \) in the proof for Relation (6), the last case where \( U_n(0, x + 2) = V_n(j, x + 1) \) and \( U_n(i, x) = V_n(i, x) \) could not be proven.

**Step 3.** Assuming a threshold policy for the number of used servers (Step 1) and non-idling policies for both servers (Step 2), this last step consists of proving that there cannot be any changes in the preference for one server when only one server should work.

**Theorem 1** The optimal routing for the two heterogeneous servers in speed and in quality follows a threshold policy on the queue size defined by a threshold \( u \). If \( 0 \leq x < u \) the optimal action is to maintain only one server idling, and if \( x \geq u \) it is optimal to have both servers busy.

**Proof.**

**Step 1.** We define the class of functions \( F \) from \( \{0, 1, 2, 1/2\} \times \mathbb{N} \) to \( \mathbb{R} \) as follows: \( f \in F \) if for \( x \geq 0 \), we have

\[
\begin{align*}
  f(i, x + 1) &\geq f(i, x), \text{ for } i = 0, 1, 2, 1/2, \\
  f(1/2, x) &\geq f(i, x) \geq f(0, x), \text{ for } i = 1, 2, \\
  f(i, x + 2) + f(1/2, x) &\geq f(i, x + 1) + f(1/2, x + 1), \text{ for } i = 1, 2, \\
  f(0, x + 2) + f(i, x) &\geq f(0, x + 1) + f(i, x + 1), \text{ for } i = 1, 2, \\
  f(1/2, x + 1) + f(i, x) &\geq f(i, x + 1) + f(1/2, x), \text{ for } i = 1, 2, \\
  f(i, x + 1) + f(0, x) &\geq f(0, x + 1) + f(i, x), \text{ for } i = 1, 2, \\
  f(i, x + 1) + f(j, x) &\geq f(0, x + 1) + f(1/2, x), \text{ for } i, j = 1, 2, i \neq j, \\
  f(0, x) &+ f(1/2, x) \geq f(1, x) + f(2, x).
\end{align*}
\]

If Relation (5) is true for \( V_n \), then \( V_n(i, x + 2) - V_n(1/2, x + 1) \geq V_n(i, x + 1) - V_n(1/2, x) \), for \( x \geq 0 \). If \( V_n(i, x + 1) - V_n(1/2, x) \geq 0 \), we thus deduce that \( V_n(i, x + 2) - V_n(1/2, x + 1) \geq 0 \), for \( x \geq 0 \). Consequently, if using server \( j \) is optimal when \( x \) customers are in the queue and server \( i \) is busy (\( i \neq j \)), then using server \( j \) is also optimal when \( x + 1 \) customers are in the queue and
server \(i\) is busy \((i \neq j)\). With Relation (6), the same observation holds for server \(j\) when server \(i\) is idle. Relations (5) and (6) for \(V_n\) \((n \geq 0)\) are then sufficient to prove that the optimal policy is of threshold type.

Observe that summing up Relation (5) and Relation (6) in which we replace \(x\) by \(x + 1\), we obtain

\[
f(0, x + 3) + f(1/2, x) \geq f(1/2, x + 1) + f(0, x + 2). \tag{11}
\]

Note also that summing up Relation (5) and Relation (7) leads to the convexity in \(x\) of \(f(i, x)\); summing up Relation (5) in which we replace \(x\) by \(x + 1\) and Relation (7) leads to the convexity in \(x\) of \(f(1/2, x)\); and summing up Relation (6) and Relation (8) leads to the convexity in \(x\) of \(f(0, x)\).

In Table 1 we summarize the required relations to prove each relation in the propagation from \(V_n\) to \(U_n\) (minimizing actions) and in the propagation from \(U_n\) to \(V_{n+1}\).

In what follows we prove by induction on \(n\) that both \(V_n\) and \(U_n\) are in \(\mathcal{F}\). For \(x \geq 0\), \(V_0(., x) = U_0(., x) = 0\). Then \(V_0, U_0 \in \mathcal{F}\).

**Induction from \(V_n\) to \(U_n\).** Assume now that for a given \(n \geq 0\), \(V_n \in \mathcal{F}\), and let us prove that \(U_n \in \mathcal{F}\). **Relation (3):** We have for \(x \geq 0\),

\[
U_n(0, x) \leq V_n(0, x), \tag{12}
\]

\[
U_n(0, x) \leq V_n(i, x - 1), \text{ for } x \geq 1, \tag{13}
\]

\[
U_n(0, x) \leq V_n(1/2, x - 2), \text{ for } x \geq 2. \tag{14}
\]

If \(U_n(0, x + 1) = V_n(0, x + 1)\), then combining Inequality (12) with Relation (3) for \(V_n\) proves Relation (3) for \(U_n\). If \(U_n(0, x + 1) = V_n(i, x)\), then combining Inequality (13) with Relation (3) in the case \(x \geq 1\) for \(V_n\) proves Relation (3) for \(U_n\). In the case \(x = 1\), combining Inequality (12) with Relation (4) for \(V_n\) proves Relation (3) for \(U_n\). For \(x \geq 1\), if \(U_n(0, x + 1) = V_n(1/2, x - 1)\), then combining Inequality (14) in the case \(x \geq 2\) with Relation (3) for \(V_n\) proves Relation (3) for \(U_n\). In
the case \( x = 1 \), combining Inequality (13) with Relation (4) for \( V_n \) proves Relation (3) for \( U_n \).

We have, for \( x \geq 0 \) and \( i \in \{1, 2\} \),

\[
U_n(i, x) \leq V_n(i, x),
\]
(15)

\[
U_n(i, x) \leq V_n(1/2, x - 1), \text{ for } x \geq 1.
\]
(16)

If \( U_n(i, x + 1) = V_n(i, x + 1) \), then combining Inequality (15) with Relation (3) for \( V_n \) proves Relation (3) for \( U_n \). If \( U_n(i, x + 1) = V_n(1/2, x) \), then combining Inequality (16) with Relation (3) in the case \( x \geq 1 \) for \( V_n \) proves Relation (3) for \( U_n \). In the case \( x = 1 \), combining Inequality (15) with Relation (4) for \( V_n \) proves Relation (3) for \( U_n \).

**Relation (4):** We have, for \( x \geq 0 \) and \( i \in \{1, 2\} \),

\[
U_n(0, x) \leq V_n(0, x),
\]
(17)

\[
U_n(0, x) \leq V_n(i, x - 1), \text{ for } x \geq 1.
\]
(18)

If \( U_n(i, x) = V_n(i, x) \), then combining Inequality (17) with Relation (4) for \( V_n \) proves Relation (4) for \( U_n \). For \( x \geq 1 \), if \( U_n(i, x) = V_n(1/2, x - 1) \) then combining Inequality (18) with Relation (4) for \( V_n \) proves Relation (4) for \( U_n \).

We have, for \( x \geq 0 \) and \( i \in \{1, 2\} \),

\[
U_n(i, x) \leq V_n(1/2, x - 1) \text{ for } x \geq 1.
\]
(19)

Also, \( U_n(1/2, x) = V_n(1/2, x) \). Then combining Inequality (19) with Relation (3) for \( V_n \) proves Relation (4) for \( U_n \).

**Relation (5):** We have, for \( x \geq 0 \) and \( i \in \{1, 2\} \),

\[
U_n(i, x + 1) + U_n(1/2, x + 1) \leq V_n(i, x + 1) + V_n(1/2, x + 1),
\]
(20)

\[
U_n(i, x + 1) + U_n(1/2, x + 1) \leq V_n(1/2, x) + V_n(1/2, x + 1).
\]
(21)

Also, \( U_n(1/2, x) = V_n(1/2, x) \). If \( U_n(i, x + 2) = V_n(i, x + 2) \), then combining Inequality (20) with Relation (5) for \( V_n \) proves Relation (5) for \( U_n \). If \( U_n(i, x + 2) = V_n(1/2, x + 1) \), then Inequality (21) proves Relation (5) for \( U_n \).
Relation (6): We have, for $x \geq 0$ and $i \in \{1, 2\}$,

\begin{align}
U_n(0,x + 1) + U_n(i,x + 1) & \leq V_n(0,x + 1) + V_n(i,x + 1), \\
U_n(0,x + 1) + U_n(i,x + 1) & \leq V_n(0,x + 1) + V_n(1/2,x), \\
U_n(0,x + 1) + U_n(i,x + 1) & \leq V_n(1/2,x - 1) + V_n(1/2,x) \text{ for } x \geq 1, \\
U_n(0,x + 1) + U_n(i,x + 1) & \leq V_n(i,x) + V_n(1/2,x), \\
U_n(0,x + 1) + U_n(i,x + 1) & \leq V_n(i,x) + V_n(i,x + 1).
\end{align}

If $U_n(0,x + 2) = V_n(0,x + 2)$ and $U_n(i,x) = V_n(i,x)$, then combining Inequality (22) with Relation (6) for $V_n$ proves Relation (6) for $U_n$. For $x \geq 1$, if $U_n(0,x + 2) = V_n(0,x + 2)$ and $U_n(i,x) = V_n(1/2,x - 1)$, then combining Inequality (23) with Relation (11) for $V_n$ proves Relation (6) for $U_n$. For $x \geq 1$, if $U_n(0,x + 2) = V_n(1/2,x)$ and $U_n(i,x) = V_n(1/2,x - 1)$, then Inequality (24) proves Relation (6) for $U_n$. If $U_n(0,x + 2) = V_n(1/2,x)$ and $U_n(i,x) = V_n(i,x)$, then Inequality (25) proves Relation (6) for $U_n$. If $U_n(0,x + 2) = V_n(i,x + 1)$ and $U_n(i,x) = V_n(i,x)$, then Inequality (26) proves Relation (6) for $U_n$. For $x \geq 1$, if $U_n(0,x + 2) = V_n(i,x + 1)$ and $U_n(i,x) = V_n(1/2,x - 1)$, then combining Inequality (25) with Relation (5) for $V_n$ proves Relation (6) for $U_n$. If $U_n(0,x + 2) = V_n(j,x + 1)$ and $U_n(i,x) = V_n(i,x)$, then combining Inequality (23) with Relation (9) for $V_n$ proves Relation (6) for $U_n$.

Relation (7): We have, for $x \geq 0$ and $i \in \{1, 2\}$,

\begin{align}
U_n(1/2,x) + U_n(i,x + 1) & \leq V_n(1/2,x) + V_n(i,x + 1), \\
U_n(1/2,x) + U_n(i,x + 1) & \leq 2V_n(1/2,x).
\end{align}

Also, $U_n(1/2,x) = V_n(1/2,x)$. If $U_n(i,x) = V_n(i,x)$, then combining Inequality (27) with Relation (7) for $V_n$ proves Relation (7) for $U_n$. For $x \geq 1$, if $U_n(i,x) = V_n(1/2,x - 1)$, then combining Inequality (28) with the convexity in $x$ of $V_n(1/2,x)$ proves Relation (7) for $U_n$.

Relation (8): We have for $x \geq 0$ and $i \in \{1, 2\}$

\begin{align}
U_n(0,x + 1) + U_n(i,x) & \leq V_n(0,x + 1) + V_n(i,x), \\
U_n(0,x + 1) + U_n(i,x) & \leq 2V_n(i,x), \\
U_n(0,x + 1) + U_n(i,x) & \leq V_n(1,x) + V_n(2,x), \\
U_n(0,x + 1) + U_n(i,x) & \leq V_n(1/2,x - 1) + V_n(i,x), \text{ for } x \geq 1, \\
U_n(0,x + 1) + U_n(i,x) & \leq 2V_n(1/2,x - 1), \text{ for } x \geq 1.
\end{align}

If $U_n(i,x + 1) = V_n(i,x + 1)$ and $U_n(0,x) = V_n(0,x)$, then combining Inequality (29) with Relation
(8) for $V_n$ proves Relation (8) for $U_n$. If $U_n(i,x+1) = V_n(1/2,x)$ and $U_n(0,x) = V_n(0,x)$, then combining Inequality (31) with Relation (10) for $V_n$ proves Relation (8) for $U_n$. If $U_n(i,x+1) = V_n(i,x+1)$ and $U_n(0,x) = V_n(i,x)$, then combining Inequality (30) with Relation (3) for $V_n$ proves Relation (8) for $U_n$. If $U_n(i,x+1) = V_n(1/2,x)$ and $U_n(0,x) = V_n(i,x)$, then combining Inequality (30) with Relation (4) for $V_n$ proves Relation (8) for $U_n$. For $x \geq 1$, if $U_n(i,x+1) = V_n(i,x+1)$ and $U_n(0,x) = V_n(1/2,x-1)$, then combining Inequality (32) with Relation (5) for $V_n$ proves Relation (8) for $U_n$. For $x \geq 1$, if $U_n(i,x+1) = V_n(1/2,x)$ and $U_n(0,x) = V_n(1/2,x-1)$, then Inequality (33) with the convexity in $x$ of $V_n(1/2,x)$ proves Relation (8) for $U_n$.

Relation (9): We have, for $x \geq 0$ and $i,j = 1,2(i \neq j)$,

\[
U_n(0,x+1) + U_n(1/2,x) \leq V_n(0,x+1) + V_n(1/2,x), \tag{34}
\]

\[
U_n(0,x+1) + U_n(1/2,x) \leq V_n(j,x) + V_n(1/2,x), \tag{35}
\]

\[
U_n(0,x+1) + U_n(1/2,x) \leq V_n(i,x) + V_n(1/2,x), \tag{36}
\]

\[
U_n(0,x+1) + U_n(1/2,x) \leq V_n(1/2,x-1) + V_n(1/2,x), \text{ for } x \geq 1. \tag{37}
\]

Consider the case $i,j \in \{1,2\}$ and $i \neq j$. If $U_n(i,x+1) = V_n(i,x+1)$ and $U_n(j,x) = V_n(j,x)$, then combining Inequality (34) with Relation (9) for $V_n$ proves Relation (9) for $U_n$. If $U_n(i,x+1) = V_n(1/2,x)$ and $U_n(j,x) = V_n(j,x)$, then Inequality (35) proves Relation (9) for $U_n$. If $U_n(i,x+1) = V_n(i,x+1)$ and $U_n(j,x) = V_n(1/2,x-1)$, then combining Inequality (36) with Relation (7) for $V_n$ proves Relation (9) for $U_n$. For $x \geq 1$, if $U_n(i,x+1) = V_n(1/2,x)$ and $U_n(j,x) = V_n(1/2,x-1)$, then Inequality (37) proves Relation (9) for $U_n$.

Relation (10): We have, for $x \geq 0$,

\[
U_n(1,x) + U_n(2,x) \leq V_n(1,x) + V_n(2,x), \tag{38}
\]

\[
U_n(1,x) + U_n(2,x) \leq V_n(1/2,x-1) + V_n(2,x), \text{ for } x \geq 1, \tag{39}
\]

\[
U_n(1,x) + U_n(2,x) \leq 2V_n(1/2,x-1), \text{ for } x \geq 1. \tag{40}
\]

Also, $U_n(1/2,x) = V_n(1/2,x)$. If $U_n(0,x) = V_n(0,x)$, then combining Inequality (38) with Relation (10) for $V_n$ proves Relation (10) for $U_n$. For $x \geq 1$, if $U_n(0,x) = V_n(2,x-1)$, then combining Inequality (39) with Relation (7) for $V_n$ proves Relation (10) for $U_n$. The case $U_n(0,x) = V_n(1,x-1)$ ($x \geq 1$) is identical. For $x \geq 2$, if $U_n(0,x) = V_n(1/2,x-2)$, then combining Inequality (40) with the convexity in $x$ of $V_n(1/2,x)$ proves Relation (10) for $U_n$. 

13
Induction from $U_n$ to $V_{n+1}$. Assume now that for a given $n \geq 0$, $U_n \in \mathcal{F}$. We next show that $V_{n+1} \in \mathcal{F}$.

Relations (3): We have, for $x \geq 0$,

$$V_{n+1}(0, x + 1) - V_{n+1}(0, x) = \lambda(U_n(0, x + 2) - U_n(0, x + 1)) + (\mu_1 + \mu_2)(U_n(0, x + 2) - U_n(0, x + 1)) + \frac{cR}{\lambda}.$$  

Since Relation (3) holds for $U_n$, the terms proportional to $\lambda$ and $\mu_1 + \mu_2$ are positive. We thus conclude that Relation (3) is true for $V_n$.

We have, for $x \geq 0$ and $i \in \{1, 2\}$,

$$V_{n+1}(i, x + 1) - V_{n+1}(i, x) = \lambda(U_n(i, x + 2) - U_n(i, x + 1)) + \mu_i(U_n(0, x + 1) - U_n(0, x))$$

$$+ \mu_j(U_n(i, x + 1) - U_n(i, x)) + \frac{cR}{\lambda}.$$  

Since Relation (3) holds for $U_n$, the terms proportional to $\lambda$ and $\mu_i$ and $\mu_j$ are positive. We deduce that Relation (3) is true for $V_n$.

We have, for $x \geq 0$ and $i \in \{1, 2\}$,

$$V_{n+1}(1/2, x + 1) - V_{n+1}(1/2, x) = \lambda(U_n(1/2, x + 2) - U_n(1/2, x + 1)) + \mu_i(U_n(2, x + 1) - U_n(2, x))$$

$$+ \mu_j(U_n(1, x + 1) - U_n(1, x)) + \frac{cR}{\lambda}.$$  

Since Relation (3) holds for $U_n$, the terms proportional to $\lambda$ and $\mu_i$ and $\mu_j$ are positive. Thus Relation (3) is true for $V_n$ in this case.

Relation (4): We have, for $x \geq 0$ and $i \in \{1, 2\}$,

$$V_{n+1}(i, x) - V_{n+1}(0, x) = \lambda(U_n(i, x + 1) - U_n(0, x + 1)) + \mu_i U_n(i, x) - U_n(0, x)) + \frac{cR}{\lambda}.$$  

Since Relation (4) holds for $U_n$, the terms proportional to $\lambda$ and $\mu_j$ are positive. Moreover, $\mu_i \alpha_i \geq 0$. Thus Relation (4) is true for $V_n$ in this case.

We have, for $x \geq 0$ and $i \in \{1, 2\}$,

$$V_{n+1}(1/2, x) - V_{n+1}(i, x) = \lambda(U_n(1/2, x + 1) - U_n(i, x + 1)) + \mu_i U_n(j, x) - U_n(0, x)) + \mu_j \alpha_j + \frac{cR}{\lambda}.$$  

Since Relation (4) holds for $U_n$, the terms proportional to $\lambda$ and $\mu_i$ are positive. Moreover, $\mu_j \alpha_j \geq 0$. Thus, Relation (4) holds for $V_n$. 

14
Relation (5): We have, for $x \geq 0$,

$$V_{n+1}(i, x + 2) + V_{n+1}(1/2, x) - V_{n+1}(i, x + 1) - V_{n+1}(1/2, x + 1)$$

$$= \lambda(U_n(i, x + 3) + U_n(1/2, x + 1) - U_n(i, x + 2) - U_n(1/2, x + 2))$$

$$+ \mu_i(U_n(0, x + 2) + U_n(j, x) - U_n(0, x + 1) - U_n(j, x + 1))$$

$$+ \mu_j(U_n(i, x + 2) + U_n(i, x) - 2U_n(i, x + 1)),$$

for $i, j = 1, 2$ and $i \neq j$. Since Relation (5) holds for $U_n$, the term proportional to $\lambda$ is positive. Since Relation (6) holds for $U_n$, the term proportional to $\mu_i$ is positive. Since $U_n(i, x)$ is convex in $x$, the term proportional to $\mu_j$ is positive. Thus, Relation (5) is true for $V_n$.

Relation (6): We have, for $x \geq 0$,

$$V_{n+1}(0, x + 2) + V_{n+1}(i, x) - V_{n+1}(0, x + 1) - V_{n+1}(i, x + 1)$$

$$= \lambda(U_n(0, x + 3) + U_n(i, x + 1) - U_n(0, x + 2) - U_n(i, x + 2))$$

$$+ \mu_i(U_n(0, x + 2) + U_n(0, x) - 2U_n(0, x + 1))$$

$$+ \mu_j(U_n(0, x + 2) + U_n(i, x) - U_n(0, x + 1) - U_n(i, x + 1)),$$

for $i, j = 1, 2$ and $i \neq j$. Since Relation (6) holds for $U_n$, the terms proportional to $\lambda$ and $\mu_j$ are positive. Since $U_n(0, x)$ is convex in $x$, the term proportional to $\mu_i$ is positive. We therefore deduce that Relation (6) holds for $V_n$.

Relation (7): We have, for $x \geq 0$,

$$V_{n+1}(1/2, x + 1) + V_{n+1}(i, x) - V_{n+1}(i, x + 1) - V_{n+1}(1/2, x)$$

$$= \lambda(U_n(1/2, x + 2) + U_n(i, x + 1) - U_n(i, x + 2) - U_n(1/2, x + 1))$$

$$+ \mu_i(U_n(j, x + 1) + U_n(0, x) - U_n(0, x + 1) - U_n(j, x)),$$

for $i, j = 1, 2$ and $i \neq j$. Since Relation (7) holds for $U_n$, the term proportional to $\lambda$ is positive. Since Relation (8) holds for $U_n$, the term proportional to $\mu_i$ is positive. Therefore, Relation (7) is true for $V_n$.

Relation (8): We have, for $x \geq 0$,

$$V_{n+1}(i, x + 1) + V_{n+1}(0, x) - V_{n+1}(0, x + 1) - V_{n+1}(i, x)$$

$$= \lambda(U_n(i, x + 2) + U_n(0, x + 1) - U_n(0, x + 2) - U_n(i, x + 1))$$

$$+ \mu_j(U_n(i, x + 1) + U_n(0, x) - U_n(0, x + 1) - U_n(i, x)),$$

for $i, j = 1, 2$ and $i \neq j$. Since Relation (8) holds for $U_n$, the terms proportional to $\lambda$ and $\mu_j$ are positive. Therefore, Relation (8) is true for $V_n$. 

15
Relation (9): We have, for \( x \geq 0 \),

\[
V_{n+1}(i, x+1) + V_{n+1}(j, x) - V_{n+1}(0, x+1) - V_{n+1}(1/2, x) \\
= \lambda(U_n(i, x+2) + U_n(j, x+1) - U_n(0, x+2) - U_n(1/2, x+1)) \\
+ \mu_j(U_n(i, x+1) + U_n(0, x) - U_n(0, x+1) - U_n(i, x)),
\]

for \( i, j = 1, 2 \) and \( i \neq j \). Since Relation (9) holds for \( U_n \), the term proportional to \( \lambda \) is positive. Since Relation (8) holds for \( U_n \), the term proportional to \( \mu_j \) is positive. Therefore, Relation (9) holds for \( V_n \).

Relation (10): We have, for \( x \geq 0 \),

\[
V_{n+1}(0, x) + V_{n+1}(1/2, x) - V_{n+1}(1, x) - V_{n+1}(2, x) = \lambda(U_n(0, x+1) + U_n(1/2, x+1) - U_n(1, x+1) - U_n(2, x+1)).
\]

Since Relation (10) holds for \( U_n \), the term proportional to \( \lambda \) is positive. Therefore, Relation (10) is true for \( V_n \). This finishes the proof by induction of the first step of the proof.

Step 2. Under an infinite horizon, having the two servers idling at the same time cannot be optimal. Consider the first customer in the queue. If the two servers are idle, the decision not to serve this customer simply delays the decision to the next event (we need to serve this customer anyway). The probability not to be successful in the customer treatment remains identical at the next event, however the waiting cost will increase. Thus, idling the two servers at the same time cannot be optimal in the long-run.

Step 3. In what follows, we prove that there cannot be any changes in the preference for one server if only one server should work. In other words, we prove that if server \( i \) \( (i = 1, 2) \) is preferred when one job is in the system, then server \( i \) should always work.

Consider a given \( n \) for which the computation of the value function leads to the non-optimality of both servers idling. We therefore have \( U_n(0, 1) = \min(V_n(1, 0), V_n(2, 0)) \). If \( U_n(0, 1) = V_n(1, 0) \) (preference for server 1), then from Relation (9) for \( x = 0 \), \( i = 2 \) and \( j = 1 \), we obtain \( V_n(2, 1) \geq V_n(1/2, 0) \). This implies that if two jobs are in the system, it is either optimal to only use server 1 or to use both servers. Using server 2 only is then not optimal. From Relation (5) for \( i = 2 \), we may write \( V_n(2, x+2) - V_n(1/2, x+1) \geq V_n(2, x+1) - V_n(1/2, x) \) for \( x \geq 0 \). Thus for \( x \geq 0 \), we have \( V_n(2, x+1) - V_n(1/2, x) \geq V_n(2, 1) - V_n(1/2, 0) \geq 0 \). This proves that using server 2 only is
never the optimal strategy. The same reasoning holds if \( U_n(0, 1) = V_n(2, 0) \) (preference for server 2). This finishes Step 3 and the proof of the Theorem.

**Numerical illustration.** In Figure 1, we illustrate the optimal policy as a function of the parameter \( \lambda \). We compute \( V_n(., .) \) using Equation (2) and stop the iterations until the following criterion is met

\[
\max_{i, x} \{ V_{n+1}(i, x) - V_n(i, x) \} - \min_{i, x} \{ V_{n+1}(i, x) - V_n(i, x) \} < \epsilon,
\]

for \( \epsilon = 10^{-6} \). Figure 1(a) illustrates a situation where server 2 is prioritized since this server is at the same time the fastest and the most efficient. Figure 1(b) illustrates a situation where server 1 is prioritized although this server is the slowest. As expected we observe in both situations that the higher is the arrival rate, the higher is the number of states where it is optimal to have the two servers working.

![Figure 1: Optimal Thresholds (\( \mu_1 = 2, \mu_2 = 5, c_R = 0.005\lambda \))](image)

Note that there is no simple expression for the optimal threshold nor simple criterion for which server to prioritize. This can be seen in the iterative computation of the value function. If \( \mu_1 > \mu_2 \), we obtain for \( n = 2 \),

\[
V_2(2, 0) - V_2(1, 0) = \frac{c_R}{\lambda} (\mu_1 - \mu_2) + (1 + \lambda)(\alpha_2\mu_2 - \alpha_1\mu_1) + \mu_1\mu_2(\alpha_2 - \alpha_1).
\]

This expression already gives the idea that if server 1 has at the same time the highest service rate (\( \mu_1 > \mu_2 \)), the lowest unsuccessful throughput (\( \alpha_1\mu_1 < \alpha_2\mu_2 \)) and the lowest probability of an unsuccessful service (\( \alpha_1 < \alpha_2 \)), then this server should be prioritized. This is however only a necessary condition. As \( n \) increases, the expression of \( V_n(2, 0) - V_n(1, 0) \) does not allow to determine a simple necessary and sufficient criterion for prioritizing server 1.
The same complexity holds when it comes to determining if the two servers should both work. It can be seen in the following expression, for \( n = 2 \) and \( \mu_1 > \mu_2 \),

\[
V_2(1/2, 0) - V_2(1, 1) = -\frac{cr}{\lambda} (\mu_2 + \lambda) + \alpha_2\mu_2(2 - \mu_2) - \alpha_1\mu_1^3.
\]

This expression indicates that the slower is the fastest server (server 1 here) or the more successful is the slowest server, the more likely the choice would be for having both servers working.

In Table 2 we summarize the effect of the routing choices on the performance measures.

<table>
<thead>
<tr>
<th>Queue size condition</th>
<th>Decision epoch</th>
<th>Decision</th>
<th>Decision Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x &lt; u )</td>
<td>2 servers idle after a service completion or upon a task arrival and ( x = 0 )</td>
<td>Route a task to the server which minimizes ( \alpha \mu )</td>
<td>Reduce unsatisfied throughput</td>
</tr>
<tr>
<td>( x &lt; u )</td>
<td>2 servers idle after a service completion or at a task arrival and ( x = 0 )</td>
<td>Route a task to the faster server</td>
<td>Reduce service time</td>
</tr>
<tr>
<td>( x \geq u )</td>
<td>1 server idle after a service completion or at a task arrival</td>
<td>Route a task to the remaining idle server</td>
<td>Reduce waiting time</td>
</tr>
</tbody>
</table>

### 4 Conclusion

As a conclusion, the optimal job routing for the two heterogeneous server problem with quality of resolution follows a threshold policy on the queue size, defined by the threshold \( u (u \geq 0) \). After a service completion, if two servers are idle and \( x \) jobs are in the queue \((0 < x < u)\) or after a job arrival at an empty queue and the two servers are idling, then it is optimal to route a job to only one of the two servers. The server prioritization depends on the relative importance given to the expected time spent in the system in comparison with the unsatisfied throughput. After a service completion or an arrival, if \( x \geq u \), then it is optimal to have the two servers busy.

The optimal routing has an intuitive explanation. When the queue size is small, the expected waiting time is small for arriving jobs and the main concern of the manager would then be to minimize the unsatisfied customers throughput or the service times. Above the threshold on the queue size, the major problem for the manager becomes the waiting time and the two servers are then both requested to reduce it.

An interesting but challenging future research is to extend the results to the multi-server case. It would be also interesting to consider the control problem in a more general context with multiple customer types and/or abandonment.
References


