MASTER THESIS

# Coupling Method for the K-Competing Queues Problem with Abandonments

*Author:*
Semjons AVDEJEVS

*Supervisor:*
Dr. F.M. SPIEKSMA

*Advisor:*
H. BLOK, MSc


MATHEMATISCH INSTITUUT, UNIVERSITEIT LEIDEN

To be defended on 24 August 2015

**Abstract**

This thesis is focused on the K-Competing Queues problem, which seeks for an optimal way to share a common resource among multiple user classes. The basic version of this problem, where users have unlimited patience, can be solved to optimality by using variations of the well-known $c\mu$ rule.

The more relevant version of the problem, where user patience is limited, does not yet have an optimal solution. However, some authors have been able to identify special instances, where it is optimal to prioritise one user class over the others. By employing a simple coupling technique, we give an extension to the set of instances, where a full priority policy is optimal.

Along with our search for optimality, we also attempt to give an approximate solution by a number of heuristics. As we will see from the numeric simulations, the best of all known heuristics turns out to be the one that solves the fluid approximation of the problem to optimality.

# Contents

# 1 Introduction

Queues are a common phenomenon in systems where a certain common resource cannot immediately fulfill every user's needs. They can negatively affect system performance for a number of reasons. Firstly, a user waiting for service is an idling user who can potentially be better employed by doing something else. Secondly, accommodating and keeping track of users waiting for service might come at a certain cost. These issues give rise to a natural question: what can be done to improve system performance?

One of the optimisation problems that tackles this question is the so-called *$K$-competing Queues Problem*. In it, each user that enters the system is assumed not to be unique in the sense that he belongs to a certain user class, in which he shares a number of key parameters with the other users of the same class. Users belonging to each of $K$ classes have certain random inter-arrival times, random service requirements, fixed service rates and fixed holding cost per unit time. Given that a server can process no more than one user at a time, the goal is to minimise the average or discounted holding cost over a finite or an infinite time horizon.

If one does not give users the opportunity to abandon then an optimal solution to this problem is to prioritise the user class with the highest $c\mu$ index, where $c_i$ is the holding cost per user per unit time and $1/\mu_i$ is the mean service time for a user of class $i \in \{1, .., K\}$. This policy is also known as the $c\mu$ rule. Its optimality was shown in [3] with additional results in [5, 9, 10].

However, in practice it is not always the case that users are willing to wait for service forever. To simulate this behavior, our model is modified so that each individual user in the system has a certain class-dependent impatience threshold. This greatly increases complexity of the system as state transition rates are no longer bounded. This means that many of the proof techniques that are convenient in the case without abandonements, are no longer applicable.

Due to its relevance to call centers, healthcare and Internet, a large wave of literature on this problem has been published over the recent years. Among the approaches used to find an optimal solution are value iteration in [4, 6] and coupling techniques in [6]. There are also authors who seek for approximate solutions by relaxing the problem [2] or by solving its fluid approximation to optimality [7].

This thesis is structured as follows. In Section 2 we introduce the necessary notation and solve the problem without abandonments to optimality. In Section 3 we derive sufficient conditions on the input parameters, under which the problem with abandonments can be solved to optimality. In Section 4 we introduce some of the most common heuristics and analyse their performance on a series of instances.

# 2 Model without abandonments, optimality of the $c\mu$ rule

## 2.1 Problem formulation

We consider a queueing system, in which there are $K$ different types (also referred to as classes) of customers (also somtimes referred to as jobs) and one server (also referred to as machine).

For each $i \in \{1, .., K\}$, the arrivals of type-$i$ customers into the system occur according to independent random processes with mean inter-arrival times $1/\lambda_i$.

Every customer who enters the system requires a certain random amount of server time before he is ready to leave. For each customer of type $i$, the service requirement is independent of his arrival time and of other customers and has mean $1/\mu_i$. We assume that the server cannot simultaneously serve multiple customers. If a customer of type $i$ is not served immediately after his arrival, then he joins queue $i$ of customers of the same class in an infinite capacity waiting room.

While present in the system, each customer of type $i$ induces a holding cost of $c_i \geq 0$ per time unit. The goal is to find a service policy that minimises the running cost of the system.

## 2.2 Running costs

We represent each customer $j$ who enters the system by a tuple $(E_j, S_j, I_j)$ where $E_j$ is his arrival time, $S_j$ is his service requirement and $I_j$ is his impatience threshold. In this section the latter can be either dropped or assumed to be $\infty$. We use the notation $\omega$ to represent one fixed realisation of these random variables over all possible customers over time horizon $[0, \infty)$. We will further refer to elements $\omega$ as *outcomes*.

Let $\Omega$ be the set of all possible outcomes $\omega$ equipped with a sigma-algebra $\mathcal{F}$ of all outcome combinations that are relevent for our analysis. Then, knowing the distributions of inter-arrival times, service requirements and impatience thresholds, we accordingly define a probability measure $\mathbb{P}$ on $\mathcal{F}$ and thus obtain a probability space $(\Omega, \mathcal{F}, \mathbb{P})$ that we are going to work on throughout the rest of the thesis.

Given a particular outcome $\omega \in \Omega$, we define policy $\pi$ as a stationary (i.e. non-random) state-dependent order, in which the server processes available customers. We assume that the exact service requirements and impatience thresholds for any particular customer are not known to the server until they are explored by serving or non-serving. We further assume that at time $t \in \mathbb{R}^+$ each service policy $\pi$ bases its decisions only on events (arrivals, impatience departures and service completions) that occurred over the time interval $[0, t)$, i.e. is non-anticipative. We use $\mathcal{P}$ to denote the set of all non-anticipative policies $\pi$.

For a fixed outcome $\omega \in \Omega$ and policy $\pi \in \mathcal{P}$ we use $n_i^\pi(t, \omega)$ to denote the number of customers of type $i \in \{1, .., K\}$ present in the system at time $t \in \mathbb{R}^+$. We suppose that at time $0$ the system starts from a given state $x^0 \in \mathbb{N}_0^K$, that is $n_i^\pi(0, \omega) = x_i^0$ for all $i \in \{1, .., K\}$, $\omega \in \Omega$ and $\pi \in \mathcal{P}$.

Using the definitions above, the running cost of policy $\pi$ for a fixed outcome $\omega$ for any time interval $[a, b) \subseteq [0, \infty)$ can be defined as

$$C_\omega^\pi[a, b) := \int_a^b \left( \sum_{i=1}^K c_i n_i^\pi(t, \omega) \right) dt$$

with the expected value

$$C^\pi[a, b) := \mathbb{E}\left[C_\omega^\pi[a, b)\right] = \mathbb{E}\left[ \int_a^b \left( \sum_{i=1}^K c_i n_i^\pi(t, \omega) \right) dt \right] = \int_a^b \left( \sum_{i=1}^K c_i \mathbb{E}\left[n_i^\pi(t, \omega)\right] \right) dt, \tag{1}$$

where expectation $\mathbb{E}$ is taken with respect to all outcomes $\omega \in \Omega$.

Looking for a policy with the lowest expected running cost over the infinite time horizon is thus equivalent to looking for $\pi \in \mathcal{P}$ that minimises $C^\pi[0,\infty) := \limsup_{T\to\infty} C^\pi[0,T)$ over $\pi \in \mathcal{P}$. However, as the number of new arrivals is $\mathbb{P}$-almost surely unbounded, so will be the integrals in formula (1) for $C^\pi[0,\infty)$.

We therefore consider an $\alpha$-discounted version of cost $C$ defined for $\alpha \in [0,\infty)$ as

$$S_\alpha(\pi) := \limsup_{T\to\infty} \frac{\int_0^T e^{-\alpha t}\left(\sum_{i=1}^K c_i \mathbb{E}\left[n_i^\pi(t,\omega)\right]\right) dt}{\int_0^T e^{-\alpha t} dt}.$$

For $\alpha = 0$ the above turns into the expected average running cost

$$S_0(\pi) := \limsup_{T\to\infty} \frac{\int_0^T \left(\sum_{i=1}^K c_i \mathbb{E}\left[n_i^\pi(t,\omega)\right]\right) dt}{T} = \limsup_{T\to\infty} \frac{C^\pi[0,T)}{T} \tag{2}$$

which is going to be our primary minimisation objective throughout the rest of the thesis.

It is worth mentioning that for a model without abandonments the minimisation criterion (2) makes sense only in the case of underload, that is when the system load

$$\rho := \sum_{i=1}^K \frac{\lambda_k}{\mu_k} < 1.$$

Otherwise, for any policy $\pi \in \mathcal{P}$ the server's processing rate will be insufficient to cope with all of the arrivals. This will cause one or multiple expected queue lengths $\mathbb{E}\left[n_i^\pi(t,\omega)\right]$ to increase in $t$, in which case the limit in (2) will not exist. We therefore make the following assumption throughout the rest of this section.

**Assumption 1.** $\rho < 1$

Also note that in cases where customer abandonments are present, the expected queue lengths are bounded from above for any policy $\pi$ and any system load $\rho$. Too see why this is true, pick a feasible policy $\pi_0$ that keeps the server constantly idle. In this case, the system turns into $K$ independent $M/M/\infty$ systems, each with arrival rate $\lambda_i$ and departure rate $\beta_i$ ($i \in \{1,..,K\}$). From [1] we know that $M/M/\infty$ systems are stable, i.e. for all $i \in \{1,..,K\}$:

$$\lim_{t\to\infty} \mathbb{E}\left[n_i^{\pi_0}(t,\omega)\right] = \frac{\lambda_i}{\beta_i}$$

and hence $S_0(\pi_0) = \sum_{i=1}^K {}^{c_i\lambda_i}/_{\beta_i} < \infty$. As $\pi_0$ represents the worst possible policy, for any policy $\pi \in \mathcal{P}$ we have $S_0(\pi) \le S_0(\pi_0) < \infty$. This means that our cost criterion (2) is well-defined for any system load $\rho$ and no assumptions on $\rho$ are necessary.

## 2.3 Optimality criteria

**Definition 2.3.1.** *We call a policy $\pi^* \in \mathcal{P}$ **optimal** in the average sense (further: optimal) if*

$$S_0(\pi^*) = \inf_{\pi\in\mathcal{P}} S_0(\pi).$$

**Definition 2.3.2.** *We call a policy $\pi \in \mathcal{P}$ **non-idling** if it never keeps the server idle in a non-empty state of the system.*

**Definition 2.3.3.** *We define the set of all non-idling policies by*

$$\mathcal{N} := \{\pi \in \mathcal{P} \ : \ \pi \text{ is non-idling}\}.$$

We now show that it is sufficient to only consider those policies that are non-idling.

**Proposition 2.3.1.** *If service preemptions are allowed then the following holds:*

$$\inf_{\pi\in\mathcal{N}} S_0(\pi) = \inf_{\pi\in\mathcal{P}} S_0(\pi). \tag{3}$$

Before proving Proposition 2.3.1 we will first introduce some convenient notation and prove an auxiliary result.

**Definition 2.3.4.** *Let $P_1 = (x^1, \pi_1)$ and $P_2 = (x^2, \pi_2)$ be two $\mathbb{N}_0^K$-valued random processes on the probability space $(\Omega, \mathcal{F}, \mathbb{P})$ that start in states $x^1, x^2 \in \mathbb{N}_0^K$ and follow policies $\pi_1, \pi_2$, respectively.*

- *For outcome $\omega \in \Omega$ and $i \in \{1, .., K\}$, we use $n_i^{\pi_1}(t, \omega)$ and $n_i^{\pi_2}(t, \omega)$ to denote the number of type-$i$ customers in $P_1$ and $P_2$ at time $t$, respectively.*
- *For outcome $\omega \in \Omega$ and time interval $[a, b) \subseteq [0, \infty)$ we define the cost difference between process $P_1$ and $P_2$ as*

$$\Delta C_\omega^{P_1,P_2}[a, b) := \int_a^b \left( \sum_{i=1}^K c_i \left( n_i^{\pi_1}(t, \omega) - n_i^{\pi_2}(t, \omega) \right) \right) dt \tag{4}$$

*and say that Process $P_2$ has the same or lower expected total running cost in comparison to Process $P_1$ if*

$$\mathcal{C}^{P_1,P_2} := \mathbb{E}\left[\Delta C_\omega^{\pi_1,\pi_2}[0, \infty)\right] \geq 0,$$

*provided that $\mathcal{C}^{P_1,P_2}$ is well-defined.*

**Definition 2.3.5.** *Let processes $P_1$ and $P_2$ be defined in Definition 2.3.4.*

- *We say that stationary policies $\pi_1$ and $\pi_2$ are **identical** if for any system state $x \in \mathbb{N}_0^K$, the state-dependent actions $a^{\pi_1}(x)$ and $a^{\pi_2}(x)$ satisfy $a^{\pi_1}(x) = a^{\pi_2}(x)$.*
- *We say that policies $\pi_1$ and $\pi_2$ **execute the same actions** if for system states $x_1(t)$ and $x_2(t)$ at time $t$, respectively, we have $a^{\pi_1}(x_1(t)) = a^{\pi_2}(x_2(t))$.*

**Lemma 2.3.1.** *Let $P_1 = (x, \pi_1)$ be the random process that starts from some state $x \in \mathbb{N}_0^K$ at time $0$ and follows a stationary policy $\pi_1$. Suppose there exists a random process $P_1' = (x, \pi_1')$ such that*

- *$\mathcal{C}^{P_1,P_1'} \geq (>) 0$;*
- *for $\mathbb{P}$-almost any outcome $\omega \in \Omega$ the non-stationary policy $\pi_1'$ deviates from $\pi_1$ on a finite number of customers;*
- *for $\mathbb{P}$-almost any outcome $\omega \in \Omega$ the deviation above causes state difference over a finite time interval.*

*Then there exists a stationary policy $\pi_2$ such that $S_0(\pi_1) \geq (>) S_0(\pi_2)$.*

*Proof.* Let $S \subset \mathbb{N}_0^K$ be the set of states, after hitting which at some time $t_1$, policy $\pi_1'$ deviates from policy $\pi_1$. Let $t_1' > t_1$ be the time, from which the processes $P_1$ and $P_1'$ no longer have state difference.

Observe that for any pair of neighbouring states $x, y \in \mathbb{N}_0^K$, the transition rate between these states in both directions is bounded from below by a positive costant

$$d = \min\{\min_{i\in\{1,..K\}} \beta_i, \min_{i\in\{1,..,K\}} \lambda_i\}$$

for any service policy $\pi \in \mathcal{P}$. Because our system is Markovian, each of the states $x \in \mathbb{N}_0^K$ will be hit infinitely many times irrespective of the service policy.

Therefore, the process $P_1$ is expected to hit set $S$ infinitely many times beyond time $t_1'$ so it is always possible to perform another similar deviation at some time $t_2 \geq t_1'$. If the first deviation (strictly) reduced the expected running cost over a finite time interval $[t_1, t_2)$ then it also (strictly) reduced the expected average running cost over the same interval. As deviation at time $t_2$ will also (strictly) reduce the expected average running cost over some finite time interval $[t_2, t_3)$ then a stationary policy $\pi_2$ that always follows the deviated route must have a (strictly) lower expected average running cost in comparison to policy $\pi_1$, i.e. $S_0(\pi_2)(<) \leq S_0(\pi_1)$. □

Using Definition 2.3.4 and Lemma 2.3.1, we now begin the proof of Proposition 2.3.1.

*Proof of Proposition 2.3.1.* To show that (3) holds, it is sufficient to show that for any process $P_1 = (x, \pi_1)$ with $\pi_1 \in \mathcal{P} \setminus \mathcal{N}$ one can find a policy $\pi_2 \in \mathcal{N}$ such that process $P_2 = (x, \pi_2)$ satisfies $\mathcal{C}^{P_1,P_2} \geq 0$.

Given process $P_1 = (x, \pi_1)$ we construct a process $P_1' = (x, \pi_1')$ that avoids idling longer than process $P_1$ and satisfies $\mathcal{C}^{P_1,P_1'} \geq 0$. By repeatedly performing the same procedure and pushing potential idle periods forward in time, one obtains a process $P_2 = (x, \pi_2)$ with $\mathcal{C}^{P_1,P_2} \geq 0$ that satisfies $\pi_2 \in \mathcal{N}$.

We distinguish two types of outcomes $\omega \in \Omega$:

1. Suppose that outcome $\omega \in \Omega$ is such that policy $\pi_1$ never idles the server in a non-empty system. Let policy $\pi_1'$ be an exact copy of policy $\pi_1$ on the outcome $\omega$. Then, $\Delta C_\omega^{P_1,P_1'}[0, \infty) = 0$.

2. Suppose that outcome $\omega \in \Omega$ is such that policy $\pi_1$ allows idling in a non-empty system. Let $[t_1(\omega), t_2(\omega))$ be the first such idle period under policy $\pi_1$. Define policy $\pi_1'$ as follows:

   - for $t \in [0, t_1(\omega))$ let policy $\pi_1'$ copy the actions of policy $\pi_1$;
   - for $t \in [t_1(\omega), t_2(\omega))$ let policy $\pi_1'$ serve available customers in an arbitrary order;
   - for $t \in [t_2(\omega), \infty)$ let policy $\pi_1'$ copy actions of policy $\pi_1$, whenever possible. That is, whenever policy $\pi_1$ serves a type-$i$ customer, policy $\pi'$ also serves a type-$i$ customer if such a customer is available. Otherwise policy $\pi'$ idles.

   As policy $\pi_1'$ copies actions of policy $\pi_1$ over time slot $[0, t_1(\omega))$, processes $P_1$ and $P_1'$ do not differ. This implies
   $$\Delta C_\omega^{P_1,P_1'}[0, t_1(\omega)) = 0. \tag{5}$$
   As policy $\pi_1'$ keeps the server busy as long as possible over time slot $[t_1(\omega), t_2(\omega))$ while policy $\pi_1$ idles, we have
   $$n_i^{\pi_1'}(t, \omega) \leq n_i^{\pi_1}(t, \omega) \quad \text{for any } t \in [t_1(\omega), t_2(\omega)] \text{ and } i \in \{1, .., K\}, \tag{6}$$
   which, using (4), yields
   $$\Delta C_\omega^{P_1,P_1'}[t_1(\omega), t_2(\omega)) \geq 0. \tag{7}$$
   By (6) we have $n_i^{\pi_1'}(t_2(\omega), \omega) \leq n_i^{\pi_1}(t_2(\omega), \omega)$, i.e. at time $t_2(\omega)$ the process $P_1'$ has at most the same number of customers in the system in comparison to process $P_1$. Because from time $t_2(\omega)$ onwards processes $P_1$ and $P_1'$ execute similar actions, whenever possible, we have
   $$n_i^{\pi_1'}(t, \omega) \leq n_i^{\pi_1}(t, \omega) \quad \text{for any } t \in [t_2(\omega), \infty] \text{ and } i \in \{1, .., K\}.$$
   Using (4) again:
   $$\Delta C_\omega^{P_1,P_1'}[t_2(\omega), \infty) \geq 0. \tag{8}$$
   Putting (5), (6) and (8) together we obtain $\Delta C_\omega^{P_1,P_1'}[0, \infty) \geq 0$.

Taking the expectation with respect to both types of outcomes $\omega$, we obtain $\mathcal{C}^{P_1,P_1'} \geq 0$. Moreover, policy $\pi_1'$ avoids idling longer than policy $\pi_1$ for any outcome $\omega \in \Omega$. □

## 2.4 Optimality of the $c\mu$-rule

Suppose that the number of customer classes $K = 2$.

**Proposition 2.4.1.** *Suppose the classes are labelled such that $c_1\mu_1 \geq c_2\mu_2$ and*

- *the inter-arrival times for class $i$ customers are exponentially distributed with rate $\lambda_i$ ($i = 1, 2$);*
- *the service times for class $i$ customers are exponentially distributed with rate $\mu_i$ ($i = 1, 2$);*
- *service preemptions are allowed.*

*Then it is optimal to always prioritise class 1 whenever customers of both types are present in the system.*

*Proof.* Because the transition rates are bounded from above by $d := \sum_{i=1}^{2} \lambda_i + \max\{\mu_1, \mu_2\}$, we uniformise our Markov process and assume that all arrivals and service completions occur at the ends of discrete time slots of length $\tau$ with certain probabilities. We then let $\tau \to 0$ to obtain the desired result for the continuous-time model.

Pick an arbitrary starting state $x \in \mathbb{N}^2$ and let $j_1$ and $j_2$ be the customers at the heads of the class 1 and class 2 queues at time 0. Define two random processes $P_1 = (x, \pi_1)$ and $P_2 = (x, \pi_2)$, as follows:

- during $[0, \tau)$ policy $\pi_1$ serves customer $j_1$ and policy $\pi_2$ serves customer $j_2$;
- during $[\tau, 2\tau)$ policy $\pi_1$ serves customer $j_2$ and policy $\pi_2$ serves customer $j_1$;
- during $[2\tau, \infty)$ both policies copy the actions of an arbitrary policy $\pi$. That is, whenever policy $\pi$ serves a type-$i$ customer, policies $\pi_1, \pi_2$ also serve a type-$i$ customer if such a customer is available. Otherwise policies $\pi_1$ and $\pi_2$ idle.

By construction, both customers $j_1$ and $j_2$ are served for exactly $\tau$ time units over time slot $[0, 2\tau)$. As the service time distribution of $j_1$ and $j_2$ has the memoryless property, at time $2\tau$ their remaining service time is identical (in distribution) under both processes $P_1$ and $P_2$. As both $P_1$ and $P_2$ resume from the same state and follow the same policy, for any outcome $\omega \in \Omega$ there is no state difference between the processes from time $2\tau$ onwards. This translates into

$$\Delta C_\omega^{P_1,P_2}[2\tau, \infty) = 0 \quad \text{for any } \omega \in \Omega,$$

which, in turn, implies

$$\mathbb{E}\left[\Delta C_\omega^{P_1,P_2}[2\tau, \infty)\right] = 0. \tag{9}$$

Using the memoryless property again, we see that swapping customers $j_1$ and $j_2$ between the time slots $[0, \tau)$ and $[\tau, 2\tau)$ makes them equally like to complete their service at the end of assigned time slots under $\pi_1$ and $\pi_2$. Therefore, both customers $j_1$ and $j_2$ will be present for the entire time slot $[0, 2\tau)$ in both processes unless $j_1$ completes his service at time $\tau$ in process $P_1$ and/or $j_2$ completes his service at time $\tau$ in process $P_2$. These events have probabilities $1 - e^{-\mu_1\tau}$ and $1 - e^{-\mu_2\tau}$, respectively. Hence

$$\mathbb{E}\left[\Delta C_\omega^{P_1,P_2}[0, 2\tau)\right] = c_2(1 - e^{-\mu_2\tau}) - c_1(1 - e^{-\mu_1\tau}),$$

which together with (9) implies

$$\mathcal{C}^{P_1,P_2} = c_2(1 - e^{-\mu_2\tau}) - c_1(1 - e^{-\mu_1\tau}).$$

If $\mathcal{C}^{P_1,P_2} \leq 0$ then a similar swap of processing order can be made for all pairs of adjacent time slots where policy $\pi_2$ puts service of a class 2 customer ahead of a class 1 customer. This way, one obtains the policy $\pi^*$, which fully prioritises class 1. As no change of the service order will improve the expected cost of $\pi^*$, it must be optimal.

We now want inequality

$$\begin{aligned} c_2(1 - e^{-\mu_2\tau}) - c_1(1 - e^{-\mu_1\tau}) &\leq 0 \\ c_1(1 - e^{-\mu_1\tau}) &\geq c_2(1 - e^{-\mu_2\tau}) \end{aligned} \tag{10}$$

to hold as $\tau$ is approaching zero. As both sides of (10) are continuous in $\tau$ and equal to 0 when $\tau = 0$ then a sufficient condition for (10) to hold as $\tau \to 0$ is for the derivative of the left-hand side to exceed the derivative of the right-hand side at $\tau = 0$. These derivatives are, respectively, $c_1\mu_1$ and $c_2\mu_2$, which concludes the proof. □

Note that the swapping technique executed in the proof of Proposition 2.4.1 can be applied to any pair of customer classes to show that it is optimal to prioritise class with the higher $c\mu$-index over a class with the lower $c\mu$-index. Hence, for an arbitrary number of classes $K$ we have the following corollary.

**Corollary 2.4.1.** *Suppose $K$ customer classes are labelled such that $c_i\mu_i \geq c_j\mu_j$ for $1 \leq i < j \leq K$ and*

- *the inter-arrival times for class $i$ customers are exponentially distributed with rate $\lambda_i$ ($i \in \{1, .., K\}$);*
- *the service times for class $i$ customers are exponentially distributed with rate $\mu_i$ ($i \in \{1, .., K\}$);*
- *service preemptions are allowed.*

*Then it is optimal to always prioritise class with a lower index whenever customers of two or more types are simultaneously present in the system.*

# 3 Optimal policies for model with abandonments

We now shift our attention to a variation of the $K$-competing queues model, where customers have limited patience. We simulate this behaviour by assuming that no class $i$ customer ($i \in \{1, .., K\}$) is willing to spend more than a random amount of time with mean $1/\beta_i$ in the system. Whenever a customer reaches his impatience threshold, he abandons the system irrespectively of being in the waiting queue or at the server.

Even for a relatively simple case, in which class-dependent service times and impatience thresholds are exponentially distributed and $K = 2$, it was shown in [4] that the optimal policy need not always prioritise one of the queues. In the same paper, it was shown by the means of value iteration that the optimal action explicitly depends on the current state of the system.

So far, all attempts to tackle even this simplified problem in its full generality have proven unsuccessful. Therefore, in this section we only concentrate on identifying instances, for which a full priority policy, also called *index policy*, can indeed be proven to be optimal. We first consider the special case that the service times of both classes have the same exponential distribution. Then we move on to harder modifications where the service time distribution of both classes is arbitrary or different. The proof technique that we employ, is rather naive and only gives a crude bound. However, it does show that for certain combinations of the input parameters it is indeed optimal to give full priority to one of customer classes.

## 3.1 Equal service times

We first concentrate on the case that the service time distribution is the same for both customer classes.

### 3.1.1 Exponential case

Suppose that the service times for both classes are equal and are exponentially distributed with rate $\mu$.

With the inter-arrival times, service times and impatience thresholds being exponentially distributed, our problem can be classified as a Markov Decision Problem (MDP). One way to tackle MDPs is by using the well-known Bellman optimality equation, which for our specific problem reads

$$\begin{aligned}(\lambda_1 + \lambda_2 + \beta_1 x_1 + \beta_2 x_2 + \mu)\, u(x) + g = \quad & c_1 x_1 + c_2 x_2 \\ & + \lambda_1 u(x + e_1) + \lambda_2 u(x + e_2) \\ & + \beta_1 x_1 u(x - e_1) + \beta_2 x_2 u(x - e_2) \\ & + \mu \min\{u(x - e_1), u(x - e_2)\},\end{aligned} \tag{11}$$

where

- $x \in \mathbb{N}^2$ is the current state of the system;
- $u(x)$ is the relative value function;
- $g$ is the minimum expected mean running cost.

Similarly to [6], we interpret relative function $u$ as follows. Given $x \in \mathbb{N}_0^2$ and two processes $P_0 = (0, \pi^*)$, $P_x = (x, \pi^*)$, each of which follows a policy $\pi^*$ that is optimal with respect to the overall cost, $u(x) := \mathcal{C}^{P_x, P_0}$. This way, we also get $u(0) = 0$.

As in Theorem 2.5 of [6], to describe an optimal policy it is sufficient to compare values of $u(x - e_1)$ and $u(x - e_2)$ for all states $x \in \mathbb{N}^2$. In this subsection, we will show that under certain combinations of the input parameters, one has $u(x - e_1) \leq u(x - e_2)$ for any $x \in \mathbb{N}^2$. Thus always serving class 1 customers is optimal. We do this in a more general manner that will prove to be useful later on in our work.

**Proposition 3.1.1.** *Suppose that*

- *the inter-arrival times for each type of customer have an arbitrary distribution;*

- *the impatience thresholds of class $i$ customers are exponentially distributed with rate $\beta_i$ ($i = 1, 2$) and satisfy $\beta_1 \leq \beta_2$;*
- *both types of customer have the same service time distribution with mean $1/\mu$;*
- *the holding costs $c_i$ per time unit per customer of class $i$ ($i = 1, 2$) satisfy $c_1 \geq c_2$.*

*Then $u(x - e_1) \leq u(x - e_2)$ for any $x \in \mathbb{N}^2$, i.e. the optimal total running cost for a system starting from state $x - e_1$ is always lower compared to an optimal total running cost of a system starting from state $x - e_2$.*

*Proof.* Define two processes on the same probability space. Process $P_2$ starts in the state $x - e_2$ and follows an optimal policy $\pi_2$. Process $P_1$ starts in state $x - e_1$ and the policy $\pi_1$ that it follows will be explicitly given below.

As the processes $P_1$ and $P_2$ are defined on the same probability space, we can assume that they see the same arrivals. That is, whenever an arrival occurs, the relative position between the processes remains the same.

Also assume that the first $(x_1 - 1, x_2 - 1)$ customers and all new arrivals have the same impatience thresholds and service requirements under both processes. This leaves us to consider one extra type-2 customer for process $P_1$ and one extra type-1 customer for process $P_2$. We call these customers $j_2$ and $j_1$, respectively.

For customers $j_1$ and $j_2$, use the following notation:

- $I_1 \sim \text{Exp}(\beta_1)$ for the random impatience threshold of customer $j_1$;
- $I_2 \sim \text{Exp}(\beta_2)$ for the random impatience threshold of customer $j_2$;
- $T$ for the random service requirement that we may assume to be the same for both customers $j_1$ and $j_2$.

Because both $I_1$ and $I_2$ have exponential distributions, they can be stochastically ordered. Suppose that $I_2 := \min\{I_1, I_3\}$, where $I_3 \sim \text{Exp}(\beta_2 - \beta_1)$, independent of $I_1, I_2$. This way, customer $j_1$ of process $P_2$ does not abandon the system before customer $j_2$.

Using the above, we can now give an explicit definition to policy $\pi_1$ followed by process $P_1$.

- The moment that policy $\pi_2$ serves customer $j_1$, policy $\pi_1$ is to
  - idle, if customer $j_2$ has already abandoned via impatience;
  - serve customer $j_2$, if he is still present.
- Whenever $\pi_2$ serves any customer other than $j_1$, policy $\pi_1$ is to copy its action. This is always possible because of the above assumptions.

We use the notation of Definition 2.3.4 and assume that policy $\pi_2$ starts service of customer $j_1$ at time $0$. Our goal is to bound $\mathcal{C}^{P_2,P_1}$ from below. Given a well-defined policy $\pi_1$, we now distinguish two types of outcomes $\omega \in \Omega$:

1. Suppose $\omega \in \Omega$ is such that customer $j_2$ of process $P_1$ abandons at time $I_2(\omega)$ due to impatience. Then precisely one of two events below occurs in process $P_2$.

   - Customer $j_1$ also abandons at time $I_2(\omega)$.
     In this case, both processes arrive at the same state at time $I_2(\omega)$. As process $P_1$ copies actions of process $P_2$ then the running costs for both processes from time $I_2(\omega)$ onwards are going to be identical, i.e.
     $$\Delta C_\omega^{P_2,P_1}[I_2(\omega), \infty) = 0. \tag{12}$$
     By the construction of processes $P_1$ and $P_2$, the only difference in state over time slot $[0, I_2(\omega))$ is induced by customers $j_1$ and $j_2$, both of whom abandon at time $I_2(\omega)$. Thus,
     $$\Delta C_\omega^{P_2,P_1}[0, I_2(\omega)) = c_1 I_2(\omega) - c_2 I_2(\omega) \geq 0. \tag{13}$$
     Combining (12) and (13) we obtain
     $$\Delta C_\omega^{P_2,P_1}[0, \infty) \geq (c_1 - c_2) I_2(\omega) \geq 0.$$

- Customer $j_1$ does not depart at time $I_2(\omega)$ but at some time $U(\omega) > I_2(\omega)$.
  Suppose that the abandonment of customer $j_2$ at time $I_2(\omega)$ puts process $P_1$ in some state $y \in \mathbb{N}_0^2$. Then at time $I_2(\omega)$ process $P_2$ is in a state $y + e_1$.
  As at time $I_2(\omega)$ process $P_1$ is in a smaller state compared to process $P_2$ and copies its actions, we have $\Delta C_\omega^{P_2,P_1}[I_2(\omega), \infty) \geq 0$. Using the same argument that leads to equality (13) in the previous case yields
  $$\Delta C_\omega^{P_2,P_1}[0, \infty) \geq (c_1 - c_2) I_2(\omega) \geq 0.$$

2. Now suppose that outcome $\omega \in \Omega$ is such that customer $j_2$ of process $P_1$ departs by having filled his service requirement at time $T(\omega)$.
   As customer $j_1$ of process $P_2$ has not abandoned the system before time $T(\omega)$, he also completes his service requirement at time $T(\omega)$ in process $P_2$. This pair of service completions will put both processes in the same state at time $T(\omega)$. As both processes will see the same events and execute the same actions from time $T(\omega)$ onwards, then $\Delta C_\omega^{P_2,P_1}[T(\omega), \infty) = 0$.
   Also, by construction of process $P_1$, the only difference in state between it and $P_2$ over time slot $[0, T(\omega))$ is induced by customers $j_1$ and $j_2$, both of whom depart at time $T(\omega)$. Thus
   $$\Delta C_\omega^{P_2,P_1}[0, T(\omega)) = c_1 T(\omega) - c_2 T(\omega) \geq 0$$
   and
   $$\Delta C_\omega^{P_2,P_1}[0, \infty) = (c_1 - c_2) T(\omega) \geq 0.$$

We have now established that for any outcome $\omega \in \Omega$: $\Delta C_\omega^{P_2,P_1}[0, \infty) \geq 0$. This immediately leads to $\mathcal{C}^{P_2,P_1} \geq 0$. As process $P_1 = (x - e_1, \pi_1)$ follows a potentially suboptimal policy $\pi_1$, the process $P_1^* = (x - e_1, \pi_1^*)$ where $\pi_1^*$ is an optimal policy, satisfies $\mathcal{C}^{P_1,P_1^*} \geq 0$. This implies $\mathcal{C}^{P_2,P_1^*} \geq 0$.
As processes $P_2$ and $P_1^*$ start from respective states $x - e_2$ and $x - e_1$ and employ optimal policies, by the previous inequality we have $u(x - e_1) \leq u(x - e_2)$. This concludes the proof. □

Proposition 3.1.1 and optimality equation (11) immediately lead to the following corollary.

**Corollary 3.1.1.** *Suppose that*

- *the inter-arrival times of class $i$ customers are exponentially distributed with rate $\lambda_i$ ($i = 1, 2$);*
- *the impatience thresholds of class $i$ customers are exponentially distributed with rate $\beta_i$ ($i = 1, 2$) and satisfy $\beta_1 \leq \beta_2$;*
- *both types of customer have the same exponential service time distribution with rate $\mu$;*
- *the holding costs $c_i$ per time unit per customer of class $i$ ($i = 1, 2$) satisfy $c_1 \geq c_2$.*

*Then a service policy that prioritises class 1 customers over class 2 customers is optimal.*

Observe that the proof of Proposition 3.1.1 holds true for both preemptive and non-preemptive service disciplines. While we still have the relevant machinery directly at hand, we will prove two auxiliary results, which will come in handy in Section 3.1.2.

**Lemma 3.1.1.** *Suppose that conditions of Proposition 3.1.1 hold. Then processes $P_1^* = (x - e_1, \pi_1^*)$ and $P_2^* = (x - e_2, \pi_2^*)$, both of which follow optimal policies, satisfy*

$$\mathcal{C}^{P_2^*,P_1^*} \geq (c_1 - c_2) \cdot \mathbb{E}\left[I_2 \mathbf{1}_{\{I_2 < T\}}\right],$$

*where $I_2$ stands for an impatience threshold of a type-2 customer and $T$ is his random service requirement.*

*Proof.* We use the variables defined in the proof of Proposition 3.1.1.

As processes $P_2$ and $P_2^*$ both start in the same state $x - e_2$ and both follow optimal policies, $\mathcal{C}^{P_2,P_2^*} = 0$. As the process $P_1$ follows a potentially suboptimal policy, then for processes $P_1$ and $P_1^*$ both starting in state $x - e_1$ we have $\mathcal{C}^{P_1,P_1^*} \geq 0$. This yields

$$\mathcal{C}^{P_2^*,P_1^*} = \mathcal{C}^{P_2,P_1^*} = \mathcal{C}^{P_2,P_1} + \mathcal{C}^{P_1,P_1^*} \geq \mathcal{C}^{P_2,P_1}.$$

Our goal is therefore to estimate the last term from below.

Split the space $\Omega$ into disjoint sets $A$ and $B$.

- For any outcome $\omega \in A$, customer $j_2$ of process $P_1$ leaves the system at time $I_2(\omega)$ via impatience. By the previous proof, for such $\omega$ we have $\Delta C_\omega^{P_2,P_1}[0,\infty) \geq (c_1 - c_2)I_2(\omega)$.
- For any outcome $\omega \in B$, customer $j_2$ of process $P_1$ leaves the system at time $T(\omega)$ via a service completion.
  By the previous proof, for such $\omega$ we have $\Delta C_\omega^{P_2,P_1}[0,\infty) \geq (c_1 - c_2)T(\omega)$.

This allows for the following lower bound:

$$\begin{aligned} \mathcal{C}^{P_2,P_1} &:= \mathbb{E}\left[\Delta C_\omega^{P_2,P_1}[0,\infty)\right] \\ &= \mathbb{P}(A)\cdot\mathbb{E}\left[\Delta C_\omega^{P_2,P_1}[0,\infty)|\omega\in A\right] + \mathbb{P}(B)\cdot\mathbb{E}\left[\Delta C_\omega^{P_2,P_1}[0,\infty)|\omega\in B\right] \\ &\geq \mathbb{P}(A)\cdot\mathbb{E}\left[(c_1-c_2)I_2(\omega)|\omega\in A\right] + \mathbb{P}(B)\cdot\mathbb{E}\left[(c_1-c_2)T(\omega)|\omega\in B\right] \\ &= (c_1-c_2)\cdot\left(\mathbb{P}(A)\cdot\mathbb{E}\left[I_2(\omega)|\omega\in A\right] + \mathbb{P}(B)\cdot\mathbb{E}\left[T(\omega)|\omega\in B\right]\right). \end{aligned} \tag{14}$$

As our knowledge of process $P_2$ is limited to it using *some* optimal policy, we do not have sufficient information on the process $P_1$ to explicitly compute $\mathbb{P}(A)$, $\mathbb{P}(B)$, $\mathbb{E}\left[I_2(\omega)|\omega\in A\right]$ and $\mathbb{E}\left[T(\omega)|\omega\in B\right]$. However, we can obtain some lower bounds:

- The state $x$ in the formulation of Proposition 3.1.1 can be arbitrarily large. Therefore, the probability that customers $j_1$ and $j_2$ ever reach service under any policy can be arbitrarily small and we cannot provide an estimate better than $\mathbb{P}(B) \geq 0$. This reduces the estimate (14) to

$$\mathcal{C}^{P_2,P_1} \geq (c_1-c_2)\cdot\mathbb{P}(A)\cdot\mathbb{E}\left[I_2(\omega)|\omega\in A\right].$$

- $\mathbb{P}(A)$ and $\mathbb{E}\left[I_2(\omega)|\omega\in A\right]$ depend explicitly on the policy used by the process $P_1$. Both of these values are at their lowest if process $P_1$ is to start processing customer $j_2$ immediately at time $0$. Assuming that this is the case, we obtain a lower bound

$$\begin{aligned} \mathcal{C}^{P_2,P_1} &\geq (c_1-c_2)\mathbb{P}(A)\cdot\mathbb{E}\left[I_2(\omega)|\omega\in A\right] \\ &\geq (c_1-c_2)\mathbb{P}(I_2<T)\mathbb{E}(I_2|I_2<T) \\ &= (c_1-c_2)\cdot\mathbb{E}\left[I_2\mathbf{1}_{\{I_2<T\}}\right]. \end{aligned}$$

□

**Lemma 3.1.2.** *Recall the conditions of Proposition 3.1.1 and suppose that $c_1 < c_2$. Then processes $P_1^* = (x-e_1,\pi_1^*)$ and $P_2^* = (x-e_2,\pi_2^*)$, both of which follow optimal policies, satisfy*

$$\mathcal{C}^{P_2^*,P_1^*} \geq \frac{c_1-c_2}{\beta_2}.$$

*Proof.* We copy the proof of Lemma 3.1.1 up to the point where estimate (14) is obtained.

For any outcome $\omega \in B$, the service completion time $T(\omega)$ of customer $j_2$ under process $P_1$ cannot exceed his impatience threshold $I_2(\omega)$. As $(c_1 - c_2) < 0$ and then our further estimate goes as follows:

$$\begin{aligned} \mathcal{C}^{P_2,P_1} &\geq (c_1-c_2)\cdot\left(\mathbb{P}(A)\cdot\mathbb{E}\left[I_2(\omega)|\omega\in A\right] + \mathbb{P}(B)\cdot\mathbb{E}\left[T(\omega)|\omega\in B\right]\right) \\ &\geq (c_1-c_2)\cdot\left(\mathbb{P}(A)\cdot\mathbb{E}\left[I_2(\omega)|\omega\in A\right] + \mathbb{P}(B)\cdot\mathbb{E}\left[I_2(\omega)|\omega\in B\right]\right) \\ &= (c_1-c_2)\cdot\mathbb{E}\left[I_2(\omega)|\omega\in\Omega\right] \\ &= \tfrac{c_1-c_2}{\beta_2}. \end{aligned}$$

□

### 3.1.2 Non-exponential case

Suppose that the service times for both classes are equal but not exponentially distributed. Because the service time need not have the memoryless property, the system is no longer Markovian, which does not allow to use the Bellman equation (11).

Throughout this subsection, we use $T$ to denote the random service time and assume that it has a density $f_T$ on $[0,\infty)$ and $\mathbb{E}T = \frac{1}{\mu}$.

**Non-preemptive case**

We first assume that service preemptions are not allowed. As will be discussed in the next subsection, this assumption is vital for our relatively simple coupling argument to produce any results.

**Definition 3.1.1.** *Suppose that the service time $T$ has density $f_T : [0, \infty) \to \mathbb{R}_0^+$. For any $z \in [0, \sup\{Supp(f_T)\})$, we define the residual service time $R^z$ observed at time $z$ by the density $f_{R^z}(x)$:*

$$f_{R^z}(x) := \frac{f_T(x+z)}{\int_z^\infty f_T(x)dx}.$$

Observe that $R^z$ is a well-defined random variable on $\mathbb{R}_0^+$ for any $z \in [0, \sup\{\text{Supp}(f_T)\})$ as

- For any $z \in [0, \sup\{\text{Supp}(f_T)\})$ one has $\int_z^\infty f_T(x)dx > 0$ thus $f_{R^z}(x) \geq 0$ for any $x \in \mathbb{R}_0^+$;
- $$\int_0^\infty f_{R^z}(x)dx = \frac{\int_0^\infty f_T(x+z)dx}{\int_z^\infty f_T(x)dx} = \frac{\int_z^\infty f_T(x)dx}{\int_z^\infty f_T(x)dx} = 1.$$

**Definition 3.1.2.** *We say that distribution of service time $T$ satisfies the **increasing failure rate (IFR)** property if for any $z_1, z_2 \in [0, \sup\{Supp(f_T)\})$ with $z_1 > z_2$ we have $R^{z_1} \overset{D}{\leq} R^{z_2}$. That is, for any $x \in \mathbb{R}^+$:*

$$\mathbb{P}\left(R^{z_1} \geq x\right) \leq \mathbb{P}\left(R^{z_2} \geq x\right).$$

In other words, under the IFR property, the remaining service time observed at time $z_2$ stochastically dominates the remaining service time observed at time $z_1$ if $z_1 > z_2$.

Suppose we are in state $x^0 \in \mathbb{N}^2$ at time $0$ and want to decide which queue to serve.

**Proposition 3.1.2.** *Suppose that*

- *the inter-arrival times for each type of customer have an arbitrary distribution;*
- *the impatience thresholds of class $i$ customers are exponentially distributed with rate $\beta_i$ ($i = 1, 2$) and satisfy $\beta_1 \leq \beta_2$;*
- *the service time distribution is equal for both customer classes, has mean $1/\mu$ and satisfies the IFR property;*
- *the holding costs $c_i$ per time unit per customer of class $i$ are $c_i$ ($i = 1, 2$);*
- *service preemptions are not allowed.*

*Then there exists a constant $M = M(\beta_1, \beta_2, f_T)$ such that for holding costs satisfying $c_1 \geq M \cdot c_2$, for any state $x^0 \in \mathbb{N}^2$, serving the class 1 queue is optimal over serving the class 2 queue.*

*Proof.* We first assume that the holding costs satisfy $c_1 \geq c_2$. Later we show that the same proof technique is also applicable for the case $c_1 < c_2$.

Given an initial state $x^0 \in \mathbb{N}_0^2$, we define two random processes $P_1 = (x^0, \pi_1)$ and $P_2 = (x^0, \pi_2)$ on the same probability space. Suppose that policy $\pi_2$ of process $P_2$ assigns service to a type-2 customer $j_2$ at time $0$ and is optimal afterwards. On the other hand, policy $\pi_1$ of process $P_1$ assigns service to a type-1 customer $j_1$ at time $0$ and its further actions will be explicitly given below.

We use the notation of Definition 2.3.4 and say that policy $\pi_1$ has a lower expected running cost in comparison to policy $\pi_2$ if $\mathcal{C}^{P_2,P_1} \geq 0$. Note that actions of policy $\pi_1$ after customer $j_1$ departs are potentially suboptimal. Therefore, for a random process $P_1' = (x^0, \pi_1')$ that follows a policy $\pi_1'$ which first serves $j_1$ and is later optimal, one has $\mathcal{C}^{P_1,P_1'} \geq 0$. If $\mathcal{C}^{P_2,P_1} \geq 0$ were to hold then we would have $\mathcal{C}^{P_2,P_1'} \geq 0$, which would mean that serving class 1 in state $x^0$ yields a lower expected cost than serving class 2. Using this optimality criterion, we will derive a condition on the input parameters of type $c_1 \geq M \cdot c_2$ under which it holds true.

As both processes $P_1$ and $P_2$ are defined on the same probability space, we assume that they see the same inter-arrival times, impatience thresholds and service requirements for all customers different from $j_1$ and $j_2$. This allows to couple the processes in such a way that customers different from $j_1$ and

$j_2$ do not affect the relative position between the processes.

For customers $j_1$ and $j_2$, use the following notation:

- $I_1 \sim \text{Exp}(\beta_1)$ for the random impatience threshold of customer $j_1$;
- $I_2 \sim \text{Exp}(\beta_2)$ for the random impatience threshold of customer $j_2$;
- $T$ for the random service requirement that we may assume to be the same for both customers $j_1$ and $j_2$.
- $Z := \mathbb{E}\left[I_2 \mathbf{1}_{\{I_2<T\}}\right]$, a certain input-dependent constant.

Because $\beta_1 \leq \beta_2$ and both $I_1$ and $I_2$ are exponentially distributed, we can stochastically order these variables by letting $I_2 := \min\{I_1, I_3\}$, where $I_3 \sim \text{Exp}(\beta_2 - \beta_1)$ independent of $I_1$ and $I_2$. This way we obtain $I_1 \overset{\text{a.s.}}{\geq} I_2$ and $I_1 = I_2$ with probability $\beta_1/\beta_2$.

We now split the space $\Omega$ in 10 subspaces branching on the relative order of $I_1$, $I_2$, $T$ and other random variables that will later become relevant. In each of the branches, we give an explicit definition of the policy $\pi_1$.

1. Suppose that $\omega \in B_1$, where

$$B_1 := \{\omega \in \Omega : I_1(\omega) = I_2(\omega) < T(\omega)\}.$$

   Then at time $I_1(\omega)$ both customers $j_1$ and $j_2$ abandon simultaneously and both processes arrive at the same state $x^1$ with zero elapsed service time.

   Let policy $\pi_1$ copy the actions of policy $\pi_2$ from time $I_1(\omega)$ onwards. Then, as processes $P_1$ and $P_2$ resume from the same state and observe the same events, we get $\Delta C_\omega^{P_2,P_1}[I_1(\omega), \infty) = 0$.

   The only state difference between the processes $P_1$ and $P_2$ over time slot $[0, I_1(\omega))$ is due to customers $j_1$ and $j_2$ both of whom abandom at time $I_1(\omega)$. Therefore, $\Delta C_\omega^{P_2,P_1}[0, I_1(\omega)) = 0$ and thus

$$\Delta C_\omega^{P_2,P_1}[0, \infty) = 0$$

   for all $\omega \in B_1$.

2. Suppose that $\omega \in B_2$, where

$$B_2 := \{\omega \in \Omega : T(\omega) < \min\{I_1(\omega), I_2(\omega)\}\}.$$

   Then at time $T(\omega)$ both processes see service completions: process $P_1$ goes to some state $x^1 - e_1$ with zero elapsed service time and process $P_2$ goes to state $x^1 - e_2$ also with zero elapsed service time.

   For any outcome $\omega \in B_2$, at time $T(\omega)$ the elapsed service time for any customer in the system is zero under both processes $P_1$ and $P_2$. As impatience thresholds have the memoryless property, it means that the branch restrictions do not affect events beyond time $T(\omega)$. This puts us in the setting of Proposition 3.1.1, which states that it is possible to construct policy $\pi_1$ in such a way that the expected running cost of process $P_1$ over time slot $[T(\omega), \infty)$ will not exceed running cost of process $P_2$, with the expectation taken over all outcomes $\omega \in B_2$. That is,

$$\mathbb{E}\left[\Delta C_\omega^{P_2,P_1}[T(\omega), \infty) \cdot \mathbf{1}_{\{\omega \in B_2\}}\right] \geq 0.$$

   Using Lemma 3.1.1, we obtain a sharper lower bound

$$\mathbb{E}\left[\Delta C_\omega^{P_2,P_1}[T(\omega), \infty) \cdot \mathbf{1}_{\{\omega \in B_2\}}\right] \geq \mathbb{P}(B_2) \cdot (c_1 - c_2)Z.$$

   Due to simultaneous service completion of customers $j_1$ and $j_2$ at time $T(\omega)$, there is no difference in state between processes $P_1$ and $P_2$ over the time slot $[0, T(\omega))$. Therefore, $\Delta C_\omega^{P_2,P_1}[0, T(\omega)) = 0$ for any outcome $\omega \in B_2$ and

$$\mathbb{E}\left[\Delta C_\omega^{P_2,P_1}[0, \infty) \cdot \mathbf{1}_{\{\omega \in B_2\}}\right] \geq \mathbb{P}(B_2) \cdot (c_1 - c_2)Z.$$

3. Suppose that outcome $\omega \in B_3$, where

$$B_3 := \{\omega \in \Omega : I_2(\omega) < \min\{I_1(\omega), T(\omega)\}\}.$$

Then at time $I_2(\omega)$ customer $j_2$ abandons simultaneously under both processes, and both processes arrive in some state $x^1$. However, process $P_1$ has elapsed type-1 service time of $I_2(\omega)$ time units and it must proceed serving customer $j_1$ since the service discipline is assumed to be non-preemptive.

Depending on the actions of process $P_2$, we construct process $P_1$ from time $I_2(\omega)$ as follows:

(a) Suppose the optimal process $P_2$ serves a class-1 customer from time $I_2(\omega)$ onwards. Due to impatience thresholds being memoryless and elapsed service time for all class-1 customers being zero, all class-1 customers at time $I_2(\omega)$ in process $P_2$ are mutually equivalent. Therefore, without loss of generality, we assume that the class-1 customer picked by policy $\pi_2$ is $j_1$.

i. If $\omega \in B_{3a1}$, where

$$B_{3a1} := \{\omega \in \Omega : I_2(\omega) < I_1(\omega) \leq T(\omega)\}$$

then at time $I_1(\omega)$ customer $j_1$ will abandon simultaneously in both $P_1$ and $P_2$ so both processes will arrive at the same state $x^2$ with zero elapsed service time.

Let policy $\pi_1$ copy the actions of policy $\pi_2$ from time $I_1(\omega)$ onwards. As both processes resume from the same state and perform the same actions, $\Delta C_\omega^{P_2,P_1}[I_1(\omega), \infty) = 0$.

Due to simultaneous abandonment of customers $j_1$ and $j_2$, there is no difference in state between processes $P_1$ and $P_2$ over the time slot $[0, I_1(\omega))$. This implies $\Delta C_\omega^{P_2,P_1}[0, I_1(\omega)) = 0$ and therefore

$$\Delta C_\omega^{P_2,P_1}[0, \infty) = 0$$

for all $\omega \in B_{3a1}$.

ii. If $\omega \in B_{3a2}$, where

$$B_{3a2} := \{\omega \in \Omega : I_2(\omega) < T(\omega) < I_1(\omega)\}$$

then customer $j_1$ completes his service at time $T(\omega)$ under process $P_1$ and leaves the system at time

$$S(\omega) := \min\{I_1(\omega), I_2(\omega) + T(\omega)\}$$

in process $P_2$.

Let policy $\pi_1$ idle over time slot $[T(\omega), S(\omega))$. Then at time $S(\omega)$ both processes will be in some state $x^2$ with zero elapsed service time.

Let policy $\pi_1$ copy the actions of policy $\pi_2$ from time $S(\omega)$ onwards. As processes $P_1$ and $P_2$ start from the same state, see the same events and perform the same actions, $\Delta C_\omega^{P_2,P_1}[S(\omega), \infty) = 0$.

Note that all difference in state between processes $P_1$ and $P_2$ over time slot $[0, S(\omega))$ is due to customers $j_1$ and $j_2$. We know that $j_2$ abandons simultaneously under both processes and customer $j_1$ spends $T(\omega)$ time units under process $P_1$ and $S(\omega) > T(\omega)$ time units under process $P_2$. Therefore, $\Delta C_\omega^{P_2,P_1}[0, S(\omega)) = c_1(S(\omega) - T(\omega))$ and hence

$$\Delta C_\omega^{P_2,P_1}[0, \infty) \geq c_1(S(\omega) - T(\omega)) \geq 0$$

for any $\omega \in B_{3a2}$.

(b) Suppose the optimal Process 2 serves a type-2 customer $j_2'$ from time $I_2(\omega)$ onwards. Let

- $I_2'$ be the abandonment time of customer $j_2'$;
- $T'$ be the service requirement of customer $j_2$.

The IFR property of the service time distribution allows to assume that service of $j_2'$ under process $P_2$ cannot complete before the service of $j_1$ under process $P_1$, that is

$$T \overset{\text{a.s.}}{<} I_2 + T'.$$

i. Suppose that outcome $\omega \in B_{3b1}$, where

$$B_{3b1} := \{\omega \in B_3 : I_1(\omega) \leq I_2'(\omega), I_1(\omega) < T(\omega), I_2'(\omega) < I_2(\omega) + T'(\omega)\}.$$

Then customer $j_1$ abandons simultaneously in both processes at time $I_1(\omega)$ and customer $j_2'$ abandons simultaneously in both processes at time $I_2'(\omega)$.

Let policy $\pi_1$ idle over the time slot $[I_1(\omega), I_2'(\omega))$. Then at time $I_2'(\omega)$ both processes will be in the same state $x^2$ with zero elapsed service time. If we further let policy $\pi_1$ copy the actions of policy $\pi_2$ then, as both processes start from the same state, see the same events and execute the same actions, $\Delta C_\omega^{P_2,P_1}[I_2'(\omega), \infty) = 0$.

Note that all difference in state between processes $P_1$ and $P_2$ over time slot $[0, I_2'(\omega))$ is due to customers $j_1, j_2$ and $j_2'$. However, as all these three customers abandon simultaneously under both processes, we have $\Delta C_\omega^{P_2,P_1}[0, I_2'(\omega)) = 0$ and thus

$$\Delta C_\omega^{P_2,P_1}[0, \infty) = 0$$

for all outcomes $\omega \in B_{3b1}$.

ii. Suppose that outcome $\omega \in B_{3b2}$, where

$$B_{3b2} := \{\omega \in B_3 : I_1(\omega) < T(\omega) < I_2(\omega) + T'(\omega) < I_2'(\omega)\}.$$

Then customer $j_1$ abandons simultaneously in both processes at time $I_1(\omega)$ and customer $j_2'$ completes its service at time $I_2(\omega) + T'(\omega)$ in process $P_2$.

Let policy $\pi_1$ idle over the time slot $[I_1(\omega), I_2(\omega) + T'(\omega))$. Then at time $I_2(\omega) + T'(\omega)$ process $P_2$ will be in some state $x^2$ and process $P_1$ will be in state $x^2 + e_2$, both with zero elapsed service time.

If process $P_1$ is to follow an optimal policy from time $I_2(\omega) + T'(\omega)$, then by Proposition 3.2.1 its expected cost increase over process $P_2$ cannot be higher than $c_2/\beta_2$, which is achieved by ingnoring the extra type-2 customer and copying the actions of process $P_2$ on the rest. We therefore have

$$\mathbb{E}\left[\Delta C_{\omega'}^{P_2,P_1}[I_2(\omega) + T'(\omega), \infty) \cdot \mathbf{1}_{\{\omega \in B_{3b2}\}}\right] \geq \mathbb{P}(B_{3b2}) \cdot -c_2/\beta_2.$$

Note that all difference in state between processes $P_1$ and $P_2$ over the time slot $[0, I_2(\omega) + T'(\omega))$ is due to customers $j_1, j_2$ and $j_2'$. Customers $j_1$ and $j_2$ abandon simultaneously under both processes and customer $j_2'$ is present for the whole period $[0, I_2(\omega) + T'(\omega))$ under both processes. Therefore,

$$\Delta C_\omega^{P_2,P_1}[0, I_2(\omega) + T'(\omega)) = 0.$$

Combining the above:

$$\mathbb{E}\left[\Delta C_{\omega'}^{P_2,P_1}[0, \infty) \cdot \mathbf{1}_{\{\omega \in B_{3b2}\}}\right] \geq \mathbb{P}(B_{3b2}) \cdot -c_2/\beta_2.$$

iii. Suppose that outcome $\omega \in B_{3b3}$, where

$$B_{3b3} := \{\omega \in B_3 : T(\omega) < I_1(\omega) < I_2'(\omega) < I_2(\omega) + T'(\omega)\}.$$

Then process $P_1$ completes service of customer $j_1$ at time $T(\omega)$ and sees the abandonment of customer $j_2'$ at time $I_2'(\omega)$. Process $P_2$ sees the abandonment of $j_1$ at time $I_1(\omega)$

and the abandonment of $j_2'$ at time $I_2'(\omega)$.

Let process $P_1$ idle over the time slot $(T(\omega), I_2'(\omega))$. Then at time $I_2'(\omega)$ both processes will be in the same state $x^2$. If we further let policy $\pi_1$ copy the actions of policy $\pi_2$ then, as both processes start from the same state, see the the same events and execute the same actions, $\Delta C_\omega^{P_2,P_1}[I_2'(\omega), \infty) = 0$.

All difference in state between processes $P_1$ and $P_2$ over the time slot $[0, I_2'(\omega))$ is induced by customers $j_1, j_2$ and $j_2'$. Cusotmers $j_2$ and $j_2'$ abandon simultaneously under both processes. Customer $j_1$ is present over $[0, T(\omega))$ under process $P_1$ and over $[0, I_1(\omega))$ under process $P_2$. Hence, $\Delta C_\omega^{P_2,P_1}[0, I_2'(\omega)) = c_1(I_1(\omega) - T(\omega))$ and thus

$$\Delta C_\omega^{P_2,P_1}[0, \infty) = c_1(I_1(\omega) - T(\omega)) \geq 0$$

for all outcomes $\omega \in B_{3b3}$.

iv. Suppose outcome $\omega \in B_{3b4}$, where

$$B_{3b4} := \{\omega \in B_3 : T(\omega) < I_1(\omega) < I_2(\omega) + T'(\omega) < I_2'(\omega)\}.$$

Then process $P_1$ sees service completion of customer $j_1$ at time $T(\omega)$. Process $P_2$ sees abandonment of $j_1$ at time $I_1(\omega)$ and service completion of $j_2'$ at a later time $I_2(\omega)+T'(\omega)$.

Let policy $\pi_1$ idle over the time slot $[T(\omega), I_2(\omega) + T'(\omega))$. Then at time $I_2(\omega) + T'(\omega)$ process $P_2$ will be in some state $x^2$ and process $P_1$ will be in state $x^2 + e_2$, both with zero elapsed service time.

If process $P_1$ is to follow an optimal policy from time $I_2(\omega) + T'(\omega)$, then by Proposition 3.2.1 its expected cost increase over process $P_2$ cannot be higher than $c_2/\beta_2$, which is achieved by ingnoring the extra type-2 customer and copying the actions of process $P_2$ on the rest. Hence,

$$\mathbb{E}\left[\Delta C_{\omega'}^{P_2,P_1}[I_2(\omega) + T'(\omega), \infty) \cdot \mathbf{1}_{\{\omega \in B_{3b4}\}}\right] \geq \mathbb{P}(B_{3b4}) \cdot -c_2/\beta_2.$$

All difference in state between processes $P_1$ and $P_2$ over the time slot $[0, I_2(\omega)+T'(\omega))$ is due to customers $j_1, j_2$ and $j_2'$. Customers $j_1$ and $j_2$ abandon simultaneously under both processes and customer $j_2'$ is present for the whole period $[0, I_2(\omega) + T'(\omega))$ under both processes. Therefore, $\Delta C_\omega^{P_2,P_1}[0, I_2(\omega) + T'(\omega)) = 0$ for all $\omega \in B_{3b4}$ and

$$\mathbb{E}\left[\Delta C_{\omega'}^{P_2,P_1}[0, \infty) \cdot \mathbf{1}_{\{\omega \in B_{3b4}\}}\right] \geq \mathbb{P}(B_{3b4}) \cdot -c_2/\beta_2.$$

v. Suppose that $\omega \in B_{3b5}$, where

$$B_{3b5} := \{\omega \in B_3 : T(\omega) < I_2'(\omega) < \min\{I_1(\omega), I_2(\omega) + T'(\omega)\}\}.$$

Then process $P_1$ observes service completion of customer $j_1$ at time $T(\omega)$ and customer $j_2'$ abandons simultaneously under both processes at time $I_2'(\omega)$.

Let policy $\pi_1$ idle over the time slot $[T(\omega), I_2'(\omega))$. Then at time $I_2'(\omega)$ process $P_1$ will be in some state $x^2$ and process $P_2$ will be in state $x^2 + e_1$, both with zero elapsed service time.

By the time $I_2'(\omega)$, all three customers $j_1$, $j_2$ and $j_2'$ will have left the system under both processes. By the memoryless property of impatience thresholds, this means that for any outcome $\omega \in B_{3b5}$, the branch assumptions apply no restrictions to events occurring beyond time $I_2'(\omega)$. Having no restrictions to events that occur beyond time $I_2'(\omega)$ puts us in the setting of Proposition 3.2.2, which states that it is possible to construct policy $\pi_1$ in such a way that

$$\mathbb{E}\left[\Delta C_\omega^{P_2,P_1}[I_2'(\omega), \infty) \cdot \mathbf{1}_{\{\omega \in B_{3b5}\}}\right] \geq \mathbb{P}(B_{3b5}) \cdot c_1 Y,$$

where $Y$ is a constant given by

$$Y := \mathbb{E}\min\{S', I_1'\}$$

for $S' \stackrel{D}{=} T$ and $I_1' \stackrel{D}{=} I_1$.

All difference in state between processes $P_1$ and $P_2$ over the time slot $[0, I_2'(\omega))$ is due to customers $j_1, j_2$ and $j_2'$. Customers $j_2$ and $j_2'$ abandon simultaneously under both processes. Customer $j_1$ is present over time slot $[0, T(\omega))$ under process $P_1$ and over a bigger time slot $[0, I_2'(\omega))$ under process $P_2$. Therefore,

$$\Delta C_\omega^{P_2,P_1}[0, I_2'(\omega)) = c_1(I_2'(\omega) - T(\omega))$$

for any $\omega \in B_{3b5}$.

Combining the results above, the expected cost improvement of process $P_1$ over process $P_2$ over all $\omega \in B_{3b5}$ can be expressed as:

$$\mathbb{E}\left[\Delta C_\omega^{P_2,P_1}[0,\infty) \cdot \mathbf{1}_{\{\omega \in B_{3b5}\}}\right] \geq c_1\left(Y \cdot \mathbb{P}(B_{3b5}) + \mathbb{E}\left[(I_2'(\omega) - T(\omega)) \cdot \mathbf{1}_{\{\omega \in B_{3b5}\}}\right]\right).$$

vi. Suppose that $\omega \in B_{3b6}$, where

$$B_{3b6} := \{\omega \in B_3 : T(\omega) < I_2(\omega) + T'(\omega) < \min\{I_1(\omega), I_2'(\omega)\}\}.$$

Then process $P_1$ sees a service completion of customer $j_1$ at time $T(\omega)$ and process $P_2$ sees service completion of customer $j_2'$ at time $I_2(\omega) + T'(\omega)$.

Let policy $\pi_1$ idle over the time slot $[T(\omega), I_2(\omega) + T'(\omega))$. Then at time $I_2(\omega) + T'(\omega)$ process $P_1$ will be in some state $x^2 - e_1$ and process $P_2$ will be in some state $x^2 - e_2$, both with zero elapsed service time.

Using a similar argumentation to one in branch 2, we can define a policy $\pi_1$ in such a way that

$$\mathbb{E}\left[\Delta C_\omega^{P_2,P_1}[I_2(\omega) + T'(\omega), \infty) \cdot \mathbf{1}_{\{\omega \in B_{3b6}\}}\right] \geq \mathbb{P}(B_{3b6}) \cdot (c_1 - c_2)Z.$$

All difference in state between processes $P_1$ and $P_2$ over time slot $[0, I_2(\omega)+T'(\omega))$ is due to customers $j_1, j_2$ and $j_2'$. Customer $j_2$ abandons simultaneously under both processes, customer $j_1$ is not present over time slot $[T(\omega), I_2(\omega) + T'(\omega))$ under process $P_2$ and customer $j_2'$ is present over $[0, I_2(\omega) + T'(\omega))$ under both processes. Thus,

$$\Delta C_\omega^{P_2,P_1}[0, I_2(\omega) + T'(\omega)) = c_1(I_2(\omega) + T'(\omega) - T(\omega)) \geq 0$$

for any outcome $\omega \in B_{3b6}$.

Taking the expectation with respect to all outcomes $\omega \in B_{3b6}$, we obtain

$$\mathbb{E}\left[\Delta C_\omega^{P_2,P_1}[0,\infty) \cdot \mathbf{1}_{\{\omega \in B_{3b6}\}}\right] \geq \mathbb{P}(B_{3b6}) \cdot (c_1 - c_2)Z + c_1\mathbb{E}\left[(I_2(\omega) + T'(\omega) - T(\omega)) \cdot \mathbf{1}_{\{\omega \in B_{3b6}\}}\right].$$

vii. If outcome $\omega \in B_{3b7}$, where

$$B_{3b7} := \{\omega \in \Omega : I_2'(\omega) \leq \min\{I_1(\omega), T(\omega), I_2(\omega) + T'(\omega)\}\}$$

then customer $j_2'$ abandons simultaneously at time $I_2'(\omega)$ in both $P_1$ and $P_2$. Therefore, at time $I_2'(\omega)$ both processes will be in the same state $x^2$ with $P_1$ still being occupied with serving customer $j_1$.

This situation is similar to the one described in step 3 of the branching. The only difference is the elapsed type-1 service time, which is now increased from $I_2(\omega)$ to $I_2'(\omega)$.

By the IFR assumption, the remaining service time of customer $j_1$ is now stochastically smaller.

Let policy $\pi_1$ emulate the remaining service of customer $j_1$ at the beginning of branch 3. This is always possible by allowing additional idling, whenever necessary. By doing this, we put the relative state of processes $P_1$ and $P_2$ at time $I_2'(\omega)$ identical to the one at time $I_2(\omega)$. Therefore, we can reapply all branching steps 3a1-3a2 and 3b1-3b7, possibly multiple times.

Note that for outcomes $\omega \in B_1 \cup B_2$ one has $\Delta C_\omega^{P_2,P_1}[0,\infty) \geq 0$. Therefore, a sufficient condition for process $P_1$ having a lower expected running cost in comparison to $P_2$ is

$$\mathbb{E}\left[\Delta C_\omega^{P_2,P_1}[0,\infty) \cdot \mathbf{1}_{\{\omega \in B_3\}}\right] \geq 0. \tag{15}$$

For the cases belonging to branch 3a, we always have $\Delta C_\omega^{P_2,P_1}[0,\infty) \geq 0$ so our point of interest are cases 3b1 through 3b6. Observe that in branches 3b3, 3b5 and 3b6 process $P_1$ experiences a lower expected cost over process $P_2$ and in branches 3b2 and 3b4 its running cost is higher. Let $G_{3b3}$, $G_{3b5}$ and $G_{3b6}$ denote the excess cost of process $P_2$ over $P_1$ and $L_{3b2}$ and $L_{3b4}$ denote the excess cost of process $P_1$ over $P_2$ in respective branches.

Using $f_T$, $f_{I_1}$, $f_{I_2}$ to denote the respective densities of service distribution and type-1 and type-2 impatience thresholds, respectively:

$$\begin{aligned}
G_{3b3} &= \textstyle\int_{B_{3b3}} c_1(I_1 - T) f_{I_1} f_{I_2} f_{I_2'} f_T f_{T'} dI_1 dI_2 dI_2' dT dT' \\
G_{3b5} &= \textstyle\int_{B_{3b5}} c_1(Y + (I_2' - T)) f_{I_1} f_{I_2} f_{I_2'} f_T f_{T'} dI_1 dI_2 dI_2' dT dT' \\
G_{3b6} &= \textstyle\int_{B_{3b6}} \left((c_1 - c_2) \cdot Z + c_1(I_2 + T' - T)\right) f_{I_1} f_{I_2} f_{I_2'} f_T f_{T'} dI_1 dI_2 dI_2' dT dT' \\
L_{3b2} &= \mathbb{P}\left[B_{3b2}\right] \cdot \tfrac{c_2}{\beta_2} \\
L_{3b4} &= \mathbb{P}\left[B_{3b4}\right] \cdot \tfrac{c_2}{\beta_2}.
\end{aligned}$$

Thus if the following condition is to hold

$$G_{3b3} + G_{3b5} + G_{3b6} \geq L_{3b2} + L_{3b4} \tag{16}$$

then inequality (15) is satisfied, which would immediately lead to desired $\mathcal{C}^{P_2,P_1} \geq 0$.

Notice that the left-hand side of (16) is increasing in $c_1$ and the right-hand side is increasing in $c_2$. By taking $c_1$ large enough, say $c_1 \geq M_1 \cdot c_2$ for some large constant $M_1 = M_1(\beta_1, \beta_2, f_T)$, the input parameters will satisfy inequality (16). This will lead to the conclusion that serving class-1 queue in state $x^0$ yields a lower expected running cost.

Now suppose that $c_1 \geq c_2$ is no longer valid, i.e. $c_1 < c_2$. In this case, the bound $(c_1 - c_2) \cdot Z$ given by Corollary 3.1.1 in braches 2 and 3b6 changes to a bound ${}^{(c_1 - c_2)}/_{\beta_2}$ given by Corollary 3.1.2. This means that the cost bound obtained in branch 2 no longer indicaes a cost improvement of process $P_1$ over process $P_2$.

Furthermore, as branch 3b can be traversed multiple times (due to argumentation in 3b7), the optimality equation (15) has to be extended to

$$\begin{cases}
\mathbb{E}\left[\Delta C_\omega^{P_2,P_1}[0,\infty) \cdot \mathbf{1}_{\{\omega \in B_2 \cup B_3\}}\right] \geq 0 \\
\mathbb{E}\left[\Delta C_\omega^{P_2,P_1}[0,\infty) \cdot \mathbf{1}_{\{\omega \in B_3\}}\right] \geq 0,
\end{cases}$$

which for

$$\begin{aligned}
L_2 &= \mathbb{P}(B_2) \cdot \tfrac{c_2 - c_1}{\beta_2} \\
G_{3b6}' &= \textstyle\int_{B_{3b6}} \left(\tfrac{c_1 - c_2}{\beta_2} + c_1(I_2 + T' - T)\right) f_{I_1} f_{I_2} f_{I_2'} f_T f_{T'} dI_1 dI_2 dI_2' dT dT'
\end{aligned}$$

is equivalent to

$$\begin{cases}
G_{3b3} + G_{3b5} + G_{3b6}' \geq L_2 + L_{3b2} + L_{3b4} \\
G_{3b3} + G_{3b5} + G_{3b6}' \geq L_{3b2} + L_{3b4}.
\end{cases} \tag{17}$$

Similar to (16), both left-hand sides of (17) are increasing in $c_1$ and the right-hand sides are increasing in $c_2$. Therefore, if $c_1 \geq M_2 \cdot c_2$ for some constant $M_2 = M_2(\beta_1, \beta_2, f_T)$ then the condition (17) is satisfied.

Let $M := \max\{M_1, M_2\}$. If $c_1 \geq M \cdot c_2$ then, irrespective of the order of $c_1$ and $c_2$, both conditions (16) and (17) are satisfied. This implies $\mathcal{C}^{P_1,P_2} \leq 0$ thus it is optimal to serve class 1 customers whenever they are present. □

As one can observe in proof of Proposition 3.1.2, our rather naive coupling approach has severe limitations.

Firstly, to keep the number of events affected by one particular priority change under control, we occasionally allow unforced idling for the new policy $\pi_1$. Secondly, the optimality conditions (16) and (17) given by Proposition 3.1.2 can impose computational difficulties even if all distrubutions within the system are relatively well-behaved. However, at this point we do not see any alternative route to solve this problem.

**Discussion of the preemptive case**

We now briefly focus on the model where service preemptions are allowed and argue that it is not possible to provide even a crude solution with our approach.

As the service time distribution no longer has the memoryless property, keeping track of the elapsed service time is now a must. Doing so adds a number of extra dimensions to the state space. It also increases the number of the available actions, which now are:

1. to proceed with the customer currently in service;
2. (a) to serve a customer of the same class whose elapsed service time is more than zero;
   (b) to serve a customer of a different class whose elapsed service time is more than zero;
3. (a) to serve a customer of the same class whose elapsed service time is zero;
   (b) to serve a customer of a different class whose elapsed service time is zero.

The number of available actions can be reduced if we assume that the service discipline is preemptive-restart: in this case actions 2a and 2b are no longer available. We can also impose the IFR condition on the service time distribution, which will make action 1 always favourable over the action 3a.

This reduces the decision-maker's choice to actions 1 and 3b. In order to have a well-defined service policy, one must be able to make a choice between the two for any combination of queue lengths with any amount of elapsed service time.

At this point, we cannot see any way of solving this question even in a heavily-reduced form, let alone in full generality. Therefore, we believe that solving this problem to optimality will require a completely different approach.

## 3.2 Non-equal service times

We now focus on instances where customer classes have non-equal service time distribution.

### 3.2.1 Exponential case

Suppose that the service times for class $i$ customers are now exponentially distributed with rate $\mu_i$ ($i \in \{1, 2\}$). With the inter-arrival times and impatience thresholds also being exponentially distributed, our problem falls into class of Markov Decision Problems (MDPs). These problems can be approached

by using the Bellman optimality equation, which for our problem takes form

$$\begin{aligned}(\lambda_1+\lambda_2+\beta_1 x_1+\beta_2 x_2+\mu)\,u(x)+g= \quad & c_1x_1+c_2x_2\\ & +\lambda_1 u(x+e_1)+\lambda_2 u(x+e_2)\\ & +\beta_1 x_1 u(x-e_1)+\beta_2 x_2 u(x-e_2)\\ & +\min\{(\mu-\mu_1)u(x)+\mu_1 u(x-e_1),(\mu-\mu_2)u(x)+\mu_2 u(x-e_2)\},\end{aligned} \tag{18}$$

where

- $\mu := \max\{\mu_1,\mu_2\}$;
- $x \in \mathbb{N}^2$ is the current state of the system;
- $u(x)$ is the relative value function. Given $x \in \mathbb{N}_0^2$ and two processes $P_0 = (0,\pi_0^*)$, $P_x = (x,\pi_x^*)$, each of which follows an optimal policy, $u(x) := \mathcal{C}^{P_x,P_0}$;
- $g$ is the optimal expected mean running cost.

To describe the optimal policy in for any state $x \in \mathbb{N}_0^2$, one needs to determine which of the options in (18) attains the minimum. Observe that

$$(\mu-\mu_1)u(x)+\mu_1 u(x-e_1) \le (\mu-\mu_2)u(x)+\mu_2 u(x-e_2) \quad \Leftrightarrow \quad \mu_1(u(x)-u(x-e_1)) \ge \mu_2(u(x)-u(x-e_2)). \tag{19}$$

Thus if inequality (19) holds for all $x \in \mathbb{N}_0^2$ then serving queue 1 is always optimal. We are now going to derive conditions on the input parameters, under which this is always the case. Note that in this subsection we present results in higher generality. This will be useful later on.

Throughout the rest of this subsection, we employ the following notation:

- $I_i$ for the random impatience threshold of a class-$i$ customer ($i = 1,2$);
- $T_i$ for the random service requirement of a class-$i$ customer ($i = 1,2$).

**Proposition 3.2.1.** *Suppose that*

- *the inter-arrival times for each type of customer have an arbitrary distribution;*
- *the impatience thresholds for each type of customer have an arbitrary distribution;*
- *service times for each type of customer have an arbitrary distribution.*

*Then for all $i \in \{1,2\}$ and $x \in \mathbb{N}_0^2$ the following inequality holds:*

$$u(x+e_i)-u(x) \le c_i \mathbb{E} I_i,$$

*where $u(x)$ is the relative value funcion.*

*Proof.* Define two random processes $P_1$ and $P_2$ on the same probability space. The process $P_1 = (x,\pi_1)$ starts in the state $x$ and uses an optimal policy $\pi_1$. Process $P_2 = (x+e_i,\pi_2)$ starts in the state $x+e_i$ and follows a policy $\pi_2$, which will be explicitly given below.

As processes $P_1$ and $P_2$ are defined on the same probability space, we can assume that they see the same arrivals, the same impatience thresholds and service requirements for all customers apart from one extra type-$i$ customer under process $P_2$, whom we will call $j$.

Define the policy $\pi_2$ followed by the process $P_2$ as follows:

- policy $\pi_2$ never serves customer $j$ and lets him eventually abandon;
- for all other remaining customers, policy $\pi_2$ copies the actions of policy $\pi_1$.

Then all customers other than $j$ do not affect the relative position of the two processes.

We use the notation of Definition 2.3.4 to describe the expected cost difference between the processes. As process $P_2$, which runs from state $x+e_i$, follows a potentially suboptimal policy $\pi_2$, the process $P_2' = (x+e_i,\pi_2')$ with $\pi_2'$ an optimal policy, would satisfy $\mathcal{C}^{P_2,P_2'} \ge 0$. As processes $P_1$ and $P_2'$ start from states $x$ and $x+e_i$ and both follow optimal policies,

$$u(x+e_i)-u(x) = \mathcal{C}^{P_2',P_1} = \mathcal{C}^{P_2,P_1} - \mathcal{C}^{P_2,P_2'}.$$

As $\mathcal{C}^{P_2,P_2'} \geq 0$ then showing $\mathcal{C}^{P_2,P_1} \leq c_i \mathbb{E} I_i$ yields the necessary bound.

For an outcome $\omega \in \Omega$ let $S(\omega)$ be the time of the impatience departure of customer $j$ in process $P_2$. As all other customers in processes $P_1$ and $P_2$ enter and leave the system simultaneously, the cost difference between the processes is induced only by customer $j$. Hence

$$\begin{matrix} \Delta C_\omega^{P_2,P_1}[0, S(\omega)) = c_i S(\omega) \\ \Delta C_\omega^{P_2,P_1}[S(\omega), \infty) = 0 \end{matrix} \quad \Rightarrow \quad \Delta C_\omega^{P_2,P_1}[0, \infty) = c_i S(\omega). \tag{20}$$

As $S(\omega)$ stands for the impatience threshold of a type-$i$ customer $j$, which is not restricted by the policy $\pi_2$, we have $\mathbb{E} S(\omega) = \mathbb{E} I_i$. Together with (20) this yields the necessary $\mathcal{C}^{P_2,P_1} = c_i \mathbb{E} I_i$. □

**Proposition 3.2.2.** *Suppose that*

- *the inter-arrival times for each type of customer have an arbitrary distribution;*
- *the impatience thresholds for each type of customer have an arbitrary distribution;*
- *the service times for each type of customer have arbitrary distribution.*

*Then for all $i \in \{1, 2\}$ and $x \in \mathbb{N}_0^2$ the following inequality holds:*

$$u(x + e_i) - u(x) \geq c_i \mathbb{E} \left( \min\{I_i, T_i\} \right), \tag{21}$$

*where $u(x)$ is the relative value function.*

To prove Proposition 3.2.2, we first need to build an extension to our model. Suppose that instead of a single server, we have two and are able to process two customers at the same time. This way, the original model with a single server can be seen as a special case of the dual server model where one of the servers is constantly kept idle. Using this extension, we formulate the following Lemma:

**Lemma 3.2.1.** *For any $x^0 \in \mathbb{N}^2$, any single server process $P_1 = (x^0, \pi_1)$ and any $i \in \{1, 2\}$, there exists a dual server process $P_2 = (x^0, \pi_2)$ such that*

1. *policy $\pi_2$ uses one of the servers for exactly one type-$i$ customer overall;*
2. *$\mathcal{C}^{P_1,P_2} \geq 0$, that is expected overall running cost of process $P_2$ does not exceed that of process $P_1$.*

*Proof.* Let $P_1$ be the given single server process. On the same probability space, let $P_2$ be a dual server process that starts in the same state and follows a policy $\pi_2$, which will be explicitly given below.

As both processes are defined on the same probability space, we assume that both processes see the same arrivals, impatience thresholds and service requirements for all customers.

Suppose that the servers are labeled Server 1 and Server 2 in $P_2$ and define the dual server policy $\pi_2$ as follows:

- At time $0$, pick any customer $j_i$ of type $i$ and assign him to Server 2 until he leaves the system (by either completing service or impatience) at some time $S$. Afterwards, always keep Server 2 idle.
- On Server 1, let policy $\pi_2$ copy actions of policy $\pi_1$. Whenever policy $\pi_1$ serves customer $j_i$, let policy $\pi_2$ idle Server 1.

Observe that, by construction, policy $\pi_2$ satisfies the first condition of the Lemma.

As policy $\pi_2$ starts service of customer $j_i$ immediately and does so without interruptions, $j_i$ cannot spend more time in process $P_2$ than he does in process $P_1$. Moreover, by consturction of policy $\pi_2$, all other customers spend the same amount of time in the system in both processes. This immediately implies that process $P_2$ has the same or lower expected running cost in comparison to process $P_1$, i.e. $\mathcal{C}^{P_1,P_2} \geq 0$. □

**Remark 3.2.1.** *Observe that in the proof of Lemma 3.2.1, a restriction of the dual server process $P_2$ to Server 1 can be seen as single server process $P_2' = (x^0 - e_i, \pi_2')$. As policy $\pi_2'$ allows idling whenever the system is not empty, it is potentially suboptimal. Therefore, for an optimal policy $\pi^*$ and process $P^* = (x^0 - e_i, \pi^*)$ we have $\mathcal{C}^{P_2',P^*} \geq 0$.*

We can now prove Proposition 3.2.2:

*Proof of Proposition 3.2.2.* For fixed $i \in \{1, 2\}$, define three processes on the same probability space. The single server process $P_1 = (x, \pi_1)$ starts in state $x$ and uses an optimal policy $\pi_1$. The single server process $P_2 = (x + e_i, \pi_2)$ starts in state $x + e_i$ and uses an optimal policy $\pi_2$. The dual server process $P_3 = (x + e_i, \pi_3)$ also starts in state $x + e_i$ and uses a policy $\pi_3$, which we will construct later on.

As all processes are defined on the same probability space, we can assume that they see the same arrivals. That is, whenever an arrival occurs, the relative position between processes remains the same. Also assume that the first $(x_1, x_2)$ customers in the respective queues and all new arrivals have the same impatience thresholds and service requirements under all three processes. This leaves Processes 2 and 3 with one extra type-$i$ customer each, which we will call $j_i$.

Define policy $\pi_3$ as follows:

- Policy $\pi_3$ serves customer $j_i$ using Server 2 until $j_i$ either abandons or completes his service at some time $S$. Then $\pi_3$, always keeps Server 2 idle.
- On Server 1, policy $\pi_3$ copies actions of policy $\pi_1$. This is always possible as process $P_3$ restricted to Server 1 is a single-machine process starting in state $x$ and both processes $P_1$ and $P_3$ see the same events.

The process $P_3$ fits the construction described in Lemma 3.2.1 using $P_2$ as a reference process. Therefore, using notation of Definition 2.3.4 we get $\mathcal{C}^{P_2,P_3} \geq 0$. Our aim now is to show that $\mathcal{C}^{P_3,P_1} \geq c_i \mathbb{E}\left(\min\{I_i, T_i\}\right)$, which will immediately imply $\mathcal{C}^{P_2,P_1} \geq c_i \mathbb{E}\left(\min\{I_i, T_i\}\right)$ and thus (21).

For an outcome $\omega \in \Omega$ let $S(\omega)$ be the time of departure of customer $j_i$ in process $P_3$. As, apart from customer $j_i$, processes $P_1$ and $P_3$ see the same arrivals and departures, the only cost difference between the processes is due to customer $j_i$. Thus

$$\begin{array}{c} \Delta C_\omega^{P_3,P_1}[0, S(\omega)) = c_i S(\omega) \\ \Delta C_\omega^{P_3,P_1}[S(\omega), \infty) = 0 \end{array} \quad \Rightarrow \quad \Delta C_\omega^{P_3,P_1}[0, \infty) = c_i S(\omega). \tag{22}$$

As $S(\omega)$ is the minimum of the impatience threshold and service requirement of customer $j_i$, which is not restricted by policy $\pi_3$, we see that $\mathbb{E}\left[S(\omega)\right] = \mathbb{E}\left(\min\{I_i, T_i\}\right)$. Combined with (22) this completes the proof. □

Propositions 3.2.1 and 3.2.2 lead to the following corollary:

**Corollary 3.2.1.** *Suppose that*

- *the inter-arrival times for customers of class $i$ are exponentially distributed with rate $\lambda_i$ ($i = 1, 2$);*
- *the impatience thresholds for customers of class $i$ are exponentially distributed with rate $\beta_i$ ($i = 1, 2$);*
- *the service requirements for customers of class $i$ are exponentially distributed with rate $\mu_i$ ($i = 1, 2$);*
- *the rates satisfy*

$$\frac{c_1 \mu_1}{\beta_1 + \mu_1} \geq \frac{c_2 \mu_2}{\beta_2}. \tag{23}$$

*Then a service policy that prioritises class 1 customers over class 2 customers is optimal.*

*Proof.* By Propositions 3.2.1 and 3.2.2,

$$\mu_1(u(x) - u(x - e_1)) \geq c_1 \mu_1 \mathbb{E}\left(\min\{I_1, T_1\}\right) = \frac{c_1 \mu_1}{\beta_1 + \mu_1} \geq \frac{c_2 \mu_2}{\beta_2} = c_2 \mu_2 \mathbb{E} I_2 \geq \mu_2(u(x) - u(x - e_2)).$$

Using the optimality equation (18), we now see that it is optimal to serve queue 1 for any state $x \in \mathbb{N}^2$. This completes the proof. □

### 3.2.2 Non-exponential case

Now suppose that the service times for type-$i$ customers have a general distribution with mean $1/\mu_i$ ($i = 1, 2$). Because the service times need not have the memoryless property, we are facing a similar problem described in Section 3.1.2. That is, deciding whether it is better to proceed service or preempt will pose severe technical difficulties. Therefore, we assume that service is non-preemptive and all decisions are taken when the elapsed service time is zero.

Throughout the rest of the subsection, we will use the following notation:

- $I_i$ for the impatience threshold of a type-$i$ customer ($i = 1, 2$) with density $f_{I_i}$;
- $T_i$ for the service requirement of a type-$i$ customer ($i = 1, 2$) with density $f_{T_i}$;

Suppose we are in state $x^0$ at time 0 and want to decide which queue to serve.

**Proposition 3.2.3.** *Suppose that*

- *the inter-arrival times for each type of customer have an arbitrary distribution;*
- *the impatience thresholds of class $i$ customers are exponentially distributed with rate $\beta_i$ ($i = 1, 2$) and satisfy $\beta_1 \geq \beta_2$;*
- *the service times satisfy $T_1 \overset{a.s.}{\leq} T_2$;*
- *service preemptions are not allowed.*

*Then there exists a constant $M = M(\beta_1, \beta_2, f_{T_1}, f_{T_2})$ such that for all $c_1 \geq M \cdot c_2$ it is optimal to serve type-1 customers whenever they are present in the system.*

*Proof.* Define two random processes on the same probability space. Let process $P_2 = (x^0, \pi_2)$ be a process that first serves a type-2 customer $j_2$ and then follows an optimal policy $\pi_2$. Let process $P_1 = (x^0, \pi_1)$ follow a policy $\pi_1$ that first serves a type-1 customer $j_1$ and whose further actions will be explicitly given later on.

As both processes $P_1$ and $P_2$ are defined on the same probability space, we can assume that they see the same inter-arrival times, the same impatience thresholds and the same service requirements for all customers over the infinite time horizon.

Moreover, as $I_1$ and $I_2$ are both exponential, they can be stochastically ordered. Suppose that $I_1 := \min\{I_2, I_3\}$, where $I_3 \sim \text{Exp}(\beta_1 - \beta_2)$, independent of $I_1, I_2$. This way, customer $j_2$ can be assumed not to abandon before customer $j_1$, i.e. $I_1 \overset{a.s.}{\leq} I_2$. Similarly, we assume that $T_2 := T_1 + T_3$ where $T_3$ is some positive random variable, which allows to assume that $T_1 \overset{a.s.}{\leq} T_2$.

We refer to Definition 2.3.4 and say that process $P_1$ has a lower expected running cost in comparison to process $P_2$ if $\mathcal{C}^{P_2,P_1} \geq 0$. Note that the policy $\pi_1$ is potentially suboptimal after it is finished serving customer $j_1$. Therefore, for a process $P_1' = (x^0, \pi_1')$ where policy $\pi_1'$ serves $j_1$ and is further optimal, we have $\mathcal{C}^{P_1,P_1'} \geq 0$. As having $\mathcal{C}^{P_2,P_1'} = \mathcal{C}^{P_2,P_1} + \mathcal{C}^{P_1,P_1'} \geq 0$ would imply that serving type 1 in state $x^0$ is optimal, we will therefore seek for combinations of input parameters, under which inequality $\mathcal{C}^{P_2,P_1} \geq 0$ holds true.

We now distinguish five different types of outcomes $\omega$:

1. Suppose outcome $\omega \in B_1$, where

$$B_1 := \{\omega \in \Omega : I_1(\omega) \leq I_2(\omega) < T_2(\omega), I_1(\omega) < T_1(\omega)\}.$$

   Then at time $I_1(\omega)$ customer $j_1$ abandons simultaneously in both processes.

   Let policy $\pi_1$ idle over the time slot $[I_1(\omega), I_2(\omega))$. Then at time $I_2(\omega)$ both processes see abandonment of customer $j_2$ and both arrive at some state $x^1$ with zero elapsed service time. Let policy $\pi_1$ copy the actions of policy $\pi_2$ from time $I_2$ onwards. This is always possible as both

processes start from same state and see the same events. Then $\Delta C_\omega^{P_2,P_1}[I_2(\omega),\infty) = 0$.

The difference in state between processes $P_1$ and $P_2$ over the time slot $[0, I_2(\omega))$ is due to customers $j_1$ and $j_2$, both of whom depart simultaneously via impatience in both processes. This implies $\Delta C_\omega^{P_2,P_1}[0, I_2(\omega)) = 0$ and

$$\Delta C_\omega^{P_2,P_1}[0,\infty) = 0$$

for all outcomes $\omega \in B_1$.

2. Suppose outcome $\omega \in B_2$, where

$$B_2 := \{\omega \in \Omega : I_1(\omega) < T_2(\omega) < I_2(\omega), I_1(\omega) < T_1(\omega)\}.$$

Then at time $I_1(\omega)$ customer $j_1$ abandons simultaneously under both processes.

Let policy $\pi_1$ idle over the time slot $[I_1(\omega), T_2(\omega))$. Then at time $T_2(\omega)$ process $P_1$ will arrive at some state $x^1$ with zero elapsed service time whereas process $P_2$ will see the service completion of customer $j_2$ and arrive in a state $x^1 - e_2$, also with zero elapsed service time.

For any outcome $\omega \in B_2$, at time $T_2(\omega)$ the elapsed service time for any customer in the system is zero under both processes $P_1$ and $P_2$. As impatience thresholds have the memoryless property, it means that the branch restrictions do not affect events beyond time $T_2(\omega)$. This puts us in the setting of Proposition 3.2.1, which states that it is possible to construct policy $\pi_1$ in such a way that the expected running cost of process $P_1$ over time slot $[T_2(\omega), \infty)$ will not exceed the expected running cost of process $P_2$ by more than $c_2/\beta_2$, with the expectation taken over all outcomes $\omega \in B_2$. That is,

$$\mathbb{E}\left[\Delta C_\omega^{P_2,P_1}[T_2(\omega),\infty)) \cdot \mathbf{1}_{\{\omega \in B_2\}}\right] \geq \mathbb{P}(B_2) \cdot -c_2/\beta_2.$$

As customer $j_1$ departs simultaneously and customer $j_2$ is present throughout the whole time slot $[0, T_2(\omega))$ in both processes, $\Delta C_\omega^{P_2,P_1}[0, S_2(\omega)) = 0$ for all $\omega \in B_2$ and thus

$$\mathbb{E}\left[\Delta C_\omega^{P_2,P_1}[0,\infty) \cdot \mathbf{1}_{\{\omega \in B_2\}}\right] \geq \mathbb{P}(B_2) \cdot -c_2/\beta_2.$$

3. Suppose outcome $\omega \in B_3$, where

$$B_3 := \{\omega \in \Omega : T_1(\omega) < I_1(\omega) \leq I_2(\omega) < T_2(\omega)\}.$$

Then at time $T_1(\omega)$ customer $j_1$ completes his service in process $P_1$.

Let policy $\pi_1$ idle over the time slot $[T_1(\omega), I_2(\omega))$. Then at time $I_2(\omega)$ both processes will see customer $j_2$ abandon and thus arrive in the same state $x^1$, both with zero elapsed service time. Let policy $\pi_1$ copy the actions of policy $\pi_2$ from time $I_2(\omega)$ onwards. Because both processes resume from the same state, observe the same events and execute identical actions, $\Delta C_\omega^{P_2,P_1}[I_2(\omega),\infty) = 0$.

As customer $j_2$ spends the same amount of time under both processes but customer $j_1$ spends $I_1(\omega) - T_1(\omega)$ time units less under process $P_1$: $\Delta C_\omega^{P_2,P_1}[0, I_2(\omega)) = c_1(I_1(\omega) - T_1(\omega))$ and thus

$$\Delta C_\omega^{P_2,P_1}[0,\infty) = c_1(I_1(\omega) - T_1(\omega)) \geq 0$$

for any outcome $\omega \in B_3$.

4. Suppose outcome $\omega \in B_4$, where

$$B_4 := \{\omega \in \Omega : T_1(\omega) < I_1(\omega) < T_2(\omega) < I_2(\omega)\}.$$

Then at time $T_1(\omega)$ customer $j_1$ completes his service under process $P_1$.

Let policy $\pi_1$ idle over the time slot $[T_1(\omega), T_2(\omega))$. Then by time $T_2(\omega)$ the process $P_1$ will only

see the service completion of customer $j_1$ and arrive in some state $x$, whereas process $P_2$ will see the abandonment of $j_1$ and the service completion of $j_2$ and thus arrive in a state $x - e_2$.

Using a similar argument to the one in branch 2 and Proposition 3.2.1, we can construct a policy $\pi_1$ in such a way that

$$\mathbb{E}\left[\Delta C_\omega^{P_2,P_1}[T_2(\omega),\infty)) \cdot \mathbf{1}_{\{\omega\in B_4\}}\right] \geq \mathbb{P}(B_4) \cdot -c_2/\beta_2.$$

As customer $j_2$ is present over the entire time slot $[0, T_2(\omega))$ under both processes but customer $j_1$ spends $I_1(\omega) - T_1(\omega)$ time units less under process $P_1$,

$$\Delta C_\omega^{P_2,P_1}[0, T_2(\omega)) = c_1(I_1(\omega) - T_1(\omega))$$

for any outcome $\omega \in B_4$. This leads to

$$\mathbb{E}\left[\Delta C_\omega^{P_2,P_1}[0,\infty)) \cdot \mathbf{1}_{\{\omega\in B_4\}}\right] \geq \mathbb{P}(B_4) \cdot -c_2/\beta_2 + c_1\mathbb{E}\left[(I_1(\omega) - T_1(\omega)) \cdot \mathbf{1}_{\{\omega\in B_4\}}\right].$$

5. Suppose outcome $\omega \in B_5$, where

$$B_5 := \{\omega \in \Omega : T_1(\omega) \leq T_2(\omega) < \min\{I_1(\omega), I_2(\omega)\}\}.$$

Then at time $T_1(\omega)$ customer $j_1$ completes his service in process $P_1$ and at time $T_2(\omega)$ customer $j_2$ completes his service in process $P_2$.

Let policy $\pi_1$ idle over the time slot $[T_1(\omega), T_2(\omega))$. Then at time $T_2(\omega)$ process $P_1$ will arrive at some state $x^1 - e_1$ whereas process $P_2$ will be in state $x^1 - e_2$.

For any outcome $\omega \in B_5$, at time $T_2(\omega)$ the elapsed service time for any customer in the system is zero under both processes $P_1$ and $P_2$. As impatience thresholds have the memoryless property, it means that the branch restrictions do not affect events beyond time $T_2(\omega)$. This allows us to apply Propositions 3.2.1 and 3.2.2 which state that it is possible to construct policy $\pi_1$ in such a way that

$$\mathbb{E}\left[\Delta C_\omega^{P_2,P_1}[T_2(\omega),\infty) \cdot \mathbf{1}_{\{\omega\in B_5\}}\right] \geq \mathbb{P}(B_5) \cdot (c_1 Z - c_2/\beta_2)$$

for constant $Z := \mathbb{E}(\min\{I_1, T_1\})$.

As customer $j_2$ is present for the entire period $[0, T_2(\omega))$ in both processes but customer $j_1$ is present only over $[0, T_1(\omega))$ in process $P_1$,

$$\Delta C_\omega^{P_2,P_1}[0, T_2(\omega)) = c_1(T_2(\omega) - T_1(\omega))$$

for all $\omega \in B_5$. Combining the results above,

$$\mathbb{E}\left[\Delta C_\omega^{P_2,P_1}[T_2(\omega),\infty) \cdot \mathbf{1}_{\{\omega\in B_5\}}\right] \geq \mathbb{P}(B_5) \cdot (c_1 Z - c_2/\beta_2) + c_1\mathbb{E}\left[(T_2(\omega) - T_1(\omega)) \cdot \mathbf{1}_{\{\omega\in B_5\}}\right].$$

Observe that in case 3 the process $P_1$ yields a cost improvement over process $P_2$, in case 2 it yields a cost increase and in cases 4 and 5 there are terms contributing to both cost improvement and cost increase. To show that the expected running cost of $P_1$ is lower than one of $P_2$, we now need to balance these gains and losses. Let $G_3$, $G_4$ and $G_5$ be the excess cost of $P_2$ over $P_1$ and $L_2$, $L_4$ and $L_5$ be the excess cost of $P_1$ over $P_2$ in the respective branches. Then:

$$\begin{aligned}
G_3 &= c_1 \textstyle\int_{B_3}(I_1 - T_1) f_{I_1} f_{I_2} f_{T_1} f_{T_2} dI_1 dI_2 dT_1 dT_2 \\
G_4 &= c_1 \textstyle\int_{B_4}(I_1 - T_1) f_{I_1} f_{I_2} f_{T_1} f_{T_2} dI_1 dI_2 dT_1 dT_2 \\
G_5 &= c_1 \textstyle\int_{B_5}(Z + T_2 - T_1) f_{I_1} f_{I_2} f_{T_1} f_{T_2} dI_1 dI_2 dT_1 dT_2 \\
L_2 &= \tfrac{c_2}{\beta_2}\mathbb{P}[B_2] \\
L_4 &= \tfrac{c_2}{\beta_2}\mathbb{P}[B_4] \\
L_5 &= \tfrac{c_2}{\beta_2}\mathbb{P}[B_5].
\end{aligned}$$

Thus if inequality

$$G_3 + G_4 + G_5 \geq L_2 + L_4 + L_5 \tag{24}$$

is satisfied then $\mathcal{C}^{P_2,P_1} \geq 0$. As the left-hand side of (24) is increasing in $c_1$ and the right-hand side is increasing in $c_2$ then there exists a constant $M$ independent on $c_1, c_2$ such that for $c_1 \geq M \cdot c_2$ the inequality (24) holds true. This concludes the proof. □

**Remark 3.2.2.** *If the impatience thresholds have the same distribution for both customer types, i.e.* $\beta_1 = \beta_2 =: \beta$ *then one can assume that* $I_1 \overset{D}{=} I_2 \overset{D}{=}: I$. *Then cases 2 and 4 are not present in the proof of Proposition 3.2.3. In this case, the optimality condition* (24) *reduces to*

$$G_3 + G_5 \geq L_5$$

*and is immediately satisfied if*

$$c_1 \cdot \mathbb{E}\left(\min\{I, T_1\}\right) = Z \geq \frac{c_2}{\beta}.$$

## 3.3 Summary and conclusions

Despite the complexity of the problem at hand, we have managed to provide some unique solutions.

Firstly, we have shown that in cases where the impatience thresholds are exponentially distributed and the service requirement distributions are relatively well-behaved, there always exists an arrival- and cost-independent constant $M = M(\beta_1, \beta_2, f_{T_1}, f_{T_2})$ such that for holding costs satisfying $c_1 \geq M \cdot c_2$ it is always optimal to serve class 1 customers whenever they are present.

Secondly, for the case where all rates are exponential, we have obtained a relatively simple condition (23) that guarantees optimality of the index policy. While this condition is not symmetric, it can still be relevant for systems where $\mu \ll \beta$ and can be extended to systems with more than two customer classes.

# 4 Approximate solutions for model with abandonments

In the previous section we have identified the special cases, in which it can be explicitly shown that giving full priority to one of the queues is optimal in terms of the expected average running cost. However, it is often the case that the input parameters do not satisfy the aforementioned conditions and thus giving an analytical description of the optimal policy is no longer a simple task.

For the case where all rates are exponential, a possible way forward is to solve the problem numerically by the means of value iteration method described in [8]. While this technique is proven to converge to an optimal solution for any exponential instance, its high computational complexity might make it infeasible to be implemented in practice.

Another route is dropping the optimality requirement and providing a relatively simple heuristic whose performance is *good enough*. Setting this as our main goal for this section, we derive approximate solutions to instances where all rates are exponential and numerically compare their performance with the optimal policy.

## 4.1 Stationary heuristics

### 4.1.1 $c\mu$ rule

We begin with the well-known $c\mu$-rule that has been proven in Proposition 2.4.1 to be optimal for the case without customer abandonments. Even though the problem with abandonments has a different structure, there is still ground to view the $c\mu$ criterion as potentially relevant.

For $i \in \{1, 2\}$ and $t \in \mathbb{R}_+$ let

- $X_i(t) \in \mathbb{N}_0$ be the number of type-$i$ customers in the system at time $t$;
- $S_i(t) \in \{0, 1\}$ be the control variable that indicates whether type-$i$ queue is assigned service attention at time $t$ and satisfies $S_1(t) + S_2(t) \in \{0, 1\}$ for any $t$.

Suppose at time $t_0 \in \mathbb{R}_+$ we have fixed $X_i(t_0) \in \mathbb{N}^2$. Then, for known transition rates $\beta$, $\lambda$ and $\mu$:

$$\frac{d}{dt}\mathbb{E}X_i(t_0) = \lambda_i - \mu_i S_i(t_0) - \beta_i X_i(t_0).$$

Suppose we now want to myopically minimise the expected running costs at time $t_0$. This is equivalent to minimising the expected cost rate, which for time $t \in \mathbb{R}_+$ is given by $\sum_{i=1}^{2} c_i \mathbb{E}X_i(t)$. As $X_i(t_0)$ is fixed for all $i$ then the best we can do is minimising the derivative of the cost rate at time $t_0$. This can be formulated as follows:

$$\min_{S(t_0)} \left\{ \sum_{i=1}^{2} c_i \cdot \frac{d}{dt}\mathbb{E}X_i(t_0) \right\} = \min_{S(t_0)} \left\{ \sum_{i=1}^{2} c_i \left(\lambda_i - \mu_i S_i(t_0) - \beta_i X_i(t_0)\right) \right\}$$

The problem above has solution $S_i(t_0) = 1$ for $i = \arg\max_{k\in\{1,2\}}\{c_k\mu_k\}$ and $S_j(t_0) = 0$ for $j \neq i$, which is exactly the $c\mu$ rule. This shows that the $c\mu$-rule minimises the expected cost rates and hence the expected running costs in a short-term perspective.

### 4.1.2 $\frac{c\mu}{\beta}$ rule

From the optimality equation (18) we know that it is optimal to serve the type-1 queue in state $x \in \mathbb{N}^2$ if and only if the relative value function $u$ satisfies

$$\mu_1 \left(u(x) - u(x - e_1)\right) \geq \mu_2 \left(u(x) - u(x - e_2)\right). \tag{25}$$

**Lemma 4.1.1.** *For any $\epsilon > 0$ there exists a state $x^\epsilon \in \mathbb{N}^2$ such that for all states $x \succeq x^\epsilon$ and $i \in \{1, 2\}$ the following holds:*

$$u(x) - u(x - e_i) > \frac{c_i}{\beta_i} - \epsilon, \tag{26}$$

*where $\succeq$ denotes lexicographical order on $\mathbb{R}^2$.*

*Proof.* We will show that for both $i \in \{1,2\}$ there exists $\bar{x}^i \in \mathbb{N}^2$ such that for all $x \succeq \bar{x}^i$ inequality (26) holds true. By choosing $x_i^\epsilon := \max\{\bar{x}_i^1, \bar{x}_i^2\}$ for $i = 1,2$ we immediately satisfy the desired property.

Due to all customers being mutually equivalent, we assume that they are processed in the first-in-first-out (FIFO) order for any feasible policy $\pi$. Therefore, for a system starting in state $x \in \mathbb{N}_0^2$, the customer at the end of type-$i$ queue (which we will refer to as $j_i^x$) has to wait for the other $x_i$ type-$i$ customers to leave the system before he reaches the head of the queue and is considered for service.

Because of this, for any fixed $\delta > 0$ there exists $x^{\delta,1} \in \mathbb{N}^2$ such that for all starting states $x \succeq x^{\delta,1}$ the probability that customer $j_1^x$ will ever be served is less than $\delta$ under any feasible policy $\pi$. Therefore, choosing $\delta$ small enough and a starting state $x' \succeq x^{\delta,1}$, the expected cost contribution of customer $j_1^{x'}$ can be made arbitrarily close to $c_i/\beta_i$ that is achieved by never serving him. For given $\epsilon > 0$, let $\delta = \delta(\epsilon)$ be such that this expected contribution is $\epsilon$-close to $c_1/\beta_1$.

Let $P_1 = (x', \pi_{x'})$ and $P_2 = (x' - e_1, \pi_{x'-e_1})$ be a pair of random processes that follow optimal policies $\pi_{x'}$ and $\pi_{x'-e_1}$, respectively. Irrespective of how customers other than $j_1^{x'}$ are processed in process $P_1$, their expected long-term cost contribution cannot be lower than one of all customers in process $P_2$. Knowing that the expected long-term cost contribution of the extra customer $j_1^{x'}$ in process $P_1$ is at least $c_1/\beta_1 - \epsilon$ and that both $P_1$ and $P_2$ follow optimal policies:

$$u(x') > u(x' - e_1) + \frac{c_i}{\beta_i} - \epsilon.$$

As the above inequality holds true for all $x' \succeq x^{\delta(\epsilon),1}$ we deduce that $x^{\delta(\epsilon),1}$ is the $\bar{x}^1$ that we have been looking for.

$\bar{x}^2$ can be constructed analogously. □

**Remark 4.1.1.** *By* (26)*, the sequence $x^\epsilon$ is increasing as $\epsilon \to 0$.*

Suppose that $\frac{c_1\mu_1}{\beta_1} > \frac{c_2\mu_2}{\beta_2}$. Then there exists $\Delta > 0$ such that $\frac{c_1\mu_1}{\beta_1} - \Delta > \frac{c_2\mu_2}{\beta_2}$. By using (25), Lemma 4.1.1 and Proposition 3.2.2 for $0 < \epsilon < \Delta/\mu_1$, we establish the existence of a state $x^\epsilon$ such that for all $x \succeq x^\epsilon$:

$$\mu_1 \left(u(x) - u(x - e_1)\right) > \frac{c_1\mu_1}{\beta_1} - \epsilon\mu_1 > \frac{c_1\mu_1}{\beta_1} - \Delta > \frac{c_2\mu_2}{\beta_2} \geq \mu_2 \left(u(x) - u(x - e_2)\right),$$

i.e., for all states *large enough* in both coordinates, serving class 1 is optimal. If $\frac{c_1\mu_1}{\beta_1} < \frac{c_2\mu_2}{\beta_2}$ then we symmetrically obtain that serving class 2 is optimal in states where both coordinates are sufficiently large.

This leads to two conclusions:

1. Condition $\frac{c_1\mu_1}{\beta_1} > \frac{c_2\mu_2}{\beta_2}$ is necessary for it to be optimal to always serve class 1 whenever customers of both classes are present in the system.
2. If the arrival rates $\lambda_1$ and $\lambda_2$ are such that system spends most of the time in *large states* where the optimal policy is close to the $c\mu/\beta$-rule, then the $c\mu/\beta$ rule can also perform relatively well overall.

### 4.1.3 $2U$ rule

In Proposition 2.3.1 we have shown that for a problem without impatience abandonments it is sufficient to only consider the policies that are non-delay. The same is also true when the abandonments are present, as serving or non-serving a customer does not affect his decision to abandon.

In this case, any non-idling policy can be described by to which of the customers $j_1$ and $j_2$ at the heads of their respective queues the server is assigned. Obviously, the decision depends on the current state of the system and the input paramaters. However, suppose that we are only interested in the optimal serving order of $j_1$ and $j_2$ in particular and are willing to ignore all other customers.

Set the arrival rates to zero and pick the starting state $x = e_1 + e_2$. For such a system, the Bellman optimality equation (18) reduces to

$$\begin{aligned}(\beta_1 x_1 + \beta_2 x_2 + \mu)\, u(x) = \quad & c_1 x_1 + c_2 x_2 \\ & + \beta_1 x_1 u(x - e_1)^+ + \beta_2 x_2 u(x - e_2)^+ \\ & + \min\{(\mu - \mu_1)u(x) + \mu_1 u(x - e_1)^+, (\mu - \mu_2)u(x) + \mu_2 u(x - e_2)^+\}.\end{aligned}$$

Using normalisation $u(0,0) = 0$ we immediately see that $u(e_i) = \frac{c_i}{\beta_i + \mu_i}$ for $i = 1, 2$ so

$$(\beta_1+\beta_2+\mu)u(1,1) = c_1+c_2+\frac{c_2\beta_1}{\beta_2+\mu_2}+\frac{c_1\beta_2}{\beta_1+\mu_1}+\min\left\{(\mu-\mu_1)u(1,1)+\frac{c_2\mu_1}{\beta_2+\mu_2}, (\mu-\mu_2)u(1,1)+\frac{c_1\mu_2}{\beta_2+\mu_2}\right\}.$$

Suppose that the minimum in the equation above is achieved at the first term. Then

$$\begin{aligned}(\beta_1+\beta_2+\mu)u(1,1) &= c_1 + c_2 + \tfrac{c_2\beta_1}{\beta_2+\mu_2} + \tfrac{c_1\beta_2}{\beta_1+\mu_1} + (\mu-\mu_1)u(1,1) + \tfrac{c_2\mu_1}{\beta_2+\mu_2} \\ (\beta_1+\beta_2+\mu_1)u(1,1) &= c_1 + c_2 + \tfrac{c_2\beta_1}{\beta_2+\mu_2} + \tfrac{c_1\beta_2}{\beta_1+\mu_1} + \tfrac{c_2\mu_1}{\beta_2+\mu_2} \\ u(1,1) &= \tfrac{1}{\beta_1+\beta_2+\mu_1} \cdot \tfrac{c_1(\beta_1+\mu_1)(\beta_2+\mu_2)+c_2(\beta_1+\mu_1)(\beta_2+\mu_2)+c_2\beta_1(\beta_1+\mu_1)+c_1\beta_2(\beta_2+\mu_2)+c_2\mu_1(\beta_1+\mu_1)}{(\beta_1+\mu_1)(\beta_2+\mu_2)} \\ u(1,1) &= \tfrac{c_1(\beta_1+\beta_2+\mu_1)(\beta_2+\mu_2)+c_2(\beta_1+\mu_1)(\beta_1+\beta_2+\mu_1+\mu_2)}{(\beta_1+\mu_1)(\beta_2+\mu_2)(\beta_1+\beta_2+\mu_1)} \\ u(1,1) &= \tfrac{c_1}{\beta_1+\mu_1} + \tfrac{c_2}{\beta_2+\mu_2} + \tfrac{c_2\mu_2}{(\beta_2+\mu_2)(\beta_1+\beta_2+\mu_1)}\end{aligned}$$

Analogously, if the minimum is achieved at the second term, then

$$u(1,1) = \frac{c_1}{\beta_1+\mu_1} + \frac{c_2}{\beta_2+\mu_2} + \frac{c_1\mu_1}{(\beta_1+\mu_1)(\beta_1+\beta_2+\mu_2)}.$$

Therefore, if

$$\frac{c_2\mu_2}{(\beta_2+\mu_2)(\beta_1+\beta_2+\mu_1)} \le \frac{c_1\mu_1}{(\beta_1+\mu_1)(\beta_1+\beta_2+\mu_2)} \tag{27}$$

then the service order that serves $j_1$ ahead of $j_2$ yields a lower expected running cost compared to the service order that puts $j_2$ ahead of $j_1$.

We will further refer to the service rule that prioritises the classes according to the indices given by (27) as $2U$ (for *two-user*) rule.

## 4.2 Dynamic heuristics. Fluid rule

Another way to approximate the problem is described in [7]. Instead of individual stochastic arrivals, departures and service completions, the authors consider a fluid formulation where all these events are described as continuous flows with deterministic rates.

For $i \in \{1, 2\}$ and $t \in \mathbb{R}_+$ let

- $x_i(t) \in \mathbb{R}_+$ be the amount of type-$i$ customers in the system at time $t$;
- $s_i(t) \in [0, 1]$ be the control parameter that represents the fraction of server's capacity allocated to type-$i$ customer queue at time $t$ and satisfies $s_1(t) + s_2(t) \le 1$ for all $t \in \mathbb{R}_+$.

For $i \in \{1, 2\}$ let $\rho_i := {}^{\lambda_i}/_{\mu_i}$ and $\rho := \rho_1 + \rho_2$. We say that the system is underloaded if $\rho < 1$ and overloaded if $\rho > 1$. It is important that we distinguish between these cases because of the following.

- If $\rho < 1$ then the server's throughput exceeds the total inflow. Aided by the impatience outflow, for any starting state $x^* \in \mathbb{R}_+^2$ and any non-idling control $s$, the system will arrive in an equilibrium state $(0, 0)$ in a finite amount of time $\tau$, where it can be made to remain forever by assigning $s_i(t) := \rho_i$ for all $t \ge \tau$. Thus, any non-idling service policy is optimal in the average sense.

  Therefore, for $\rho < 1$ the problem of minimising the long-term cost reduces to minimising the

cost of draining the system, i.e. the cost of reaching the equilibrium state $(0,0)$, which has the following formulation:

$$\begin{array}{lll} \text{minimise}_s & \int_0^\infty \sum_{i=1}^2 c_i x_i(t) dt & \\ \text{subject to} & \frac{dx_i(t)}{dt} = \lambda_i - \mu_i s_i(t) - \beta_i x_i(t) & \text{for all } i \in \{1,2\}, t \in \mathbb{R}_+ \\ & x_i(t) \geq 0 & \text{for all } i \in \{1,2\}, t \in \mathbb{R}_+ \\ & x_i(0) = x_i^* & \text{for all } i \in \{1,2\} \\ & s_i(t) \geq 0 & \text{for all } i \in \{1,2\}, t \in \mathbb{R}_+ \\ & \sum_{i=1}^2 s_i(t) \leq 1 & \text{for all } t \in \mathbb{R}_+. \end{array} \tag{28}$$

- If $\rho > 1$ then the point $(0,0)$ cannot be an equilibrium as, in it, the total inflow exceeds the total outflow for any control $s$. As the system will always be in some positive state, it only makes sense to seek for control $s$ that is average-optimal. Formally, for $\rho > 1$ the problem has the following formulation:

$$\begin{array}{lll} \text{minimise}_s & \limsup_{T\to\infty} \frac{1}{T} \int_0^T \sum_{i=1}^2 c_i x_i(t) dt & \\ \text{subject to} & \frac{dx_i(t)}{dt} = \lambda_i - \mu_i s_i(t) - \beta_i x_i(t) & \text{for all } i \in \{1,2\}, t \in \mathbb{R}_+ \\ & x_i(t) \geq 0 & \text{for all } i \in \{1,2\}, t \in \mathbb{R}_+ \\ & x_i(0) = x_i^* & \text{for all } i \in \{1,2\} \\ & s_i(t) \geq 0 & \text{for all } i \in \{1,2\}, t \in \mathbb{R}_+ \\ & \sum_{i=1}^2 s_i(t) \leq 1 & \text{for all } t \in \mathbb{R}_+. \end{array} \tag{29}$$

### 4.2.1 Underload case ($\rho < 1$)

We now focus on the case where the system is underloaded. We will show that there exists a so-called switching curve, which splits the state space in two subspaces in such a way that giving full priority to one of the customer classes in a certain subspace is optimal.

For the coordinate $x_1$ as a function $f_1$ of the coordinate $x_2$, we define the switching curve as:

$$f_1(x) := \frac{\lambda_1}{\beta_1} + \frac{a_1 x + a_2 + (a_3 x - a_2) \cdot \left(1 + \frac{\beta_2 x}{\mu_2 - \lambda_2}\right)^{\beta_1/\beta_2}}{a_4 x}, \tag{30}$$

with constants

$$a_1 := \frac{c_1\mu_1(1-\rho)}{\beta_1}, \quad a_2 := \frac{a_1\mu_2(1-\rho_2)}{\beta_2}, \quad a_3 := \left(\frac{c_2\mu_2}{\beta_2} - \frac{c_1\mu_1}{\beta_1}\right)(1-\rho_2), \quad a_4 := \left(\frac{c_1\mu_1}{\beta_1} - \frac{c_2\mu_2}{\beta_2}\right) \cdot \frac{\beta_1}{\mu_1}.$$

Similarly, we can define the switching curve for the coordinate $x_2$ as a function $f_2$ of the coordinate $x_1$ by swapping indices in (30).

**Proposition 4.2.1.** *If $c_1\mu_1/\beta_1 \geq c_2\mu_2/\beta_2$ and $c_1\mu_1 \leq c_2\mu_2$ then an optimal solution $s^*$ to Problem* (28) *is given by*

- $s^*(t) = (1,0)$ *if* $x_1(t) > f_1(x_2(t))$;
- $s^*(t) = (0,1)$ *if* $x_1(t) \leq f_1(x_2(t))$ *and* $x_2(t) > 0$;
- $s^*(t) = (1-\rho_2, \rho_2)$ *if* $x_1(t) \leq f_1(0)$ *and* $x_2(t) = 0$.

*If $c_1\mu_1/\beta_1 \geq c_2\mu_2/\beta_2$ and $c_1\mu_1 \geq c_2\mu_2$ then an optimal solution $s^*$ to Problem* (28) *is given by*

- $s^*(t) = (1,0)$ *if* $x_1(t) > 0$;
- $s^*(t) = (\rho_1, 1-\rho_1)$ *if* $x_1(t) = 0$.

**Remark 4.2.1.** *The queue indices can always be assigned in such a way that at least one of the condition pairs of the Proposition 4.2.1 is satiefied.*

In order to proceed with the proof of Proposition 4.2.1, we require two auxilliary results:

**Lemma 4.2.1.** *There exists an $\epsilon > 0$ such that if $x(t) \preceq (\epsilon, \epsilon)$ then from time $t$ onwards it is optimal to give full priority to the non-empty queue with the highest $c\mu$ index until $(0,0)$ is reached.*

**Lemma 4.2.2.** *Suppose that it is optimal to prioritise class 1 at time 0.*
*If $(c_1\mu_1 - c_2\mu_2) \cdot ({}^{c_1\mu_1}/_{\beta_1} - {}^{c_2\mu_2}/_{\beta_2}) \leq 0$ then the following control $s^*$ is optimal:*

- $s^*(t) = (1,0)$ *if* $x_1(t) > f_1(x_2(t))$;
- $s^*(t) = (0,1)$ *if* $x_1(t) \leq f_1(x_2(t))$ *and* $x_2(t) > 0$;
- $s^*(t) = (1-\rho_2, \rho_2)$ *if* $x_1(t) \leq f_1(0)$ *and* $x_2(t) = 0$.

*If $(c_1\mu_1 - c_2\mu_2) \cdot ({}^{c_1\mu_1}/_{\beta_1} - {}^{c_2\mu_2}/_{\beta_2}) \geq 0$ then the following control $s^*$ is optimal:*

- $s^*(t) = (1,0)$ *if* $x_1(t) > 0$;
- $s^*(t) = (\rho_1, 1-\rho_1)$ *if* $x_1(t) = 0$.

**Remark 4.2.2.** *Note that by swapping the queue labels and using the function $f_2$ instead of function $f_1$ given by* (30)*, Lemma 4.2.2 gives description of the optimal policy also when it is optimal to prioritise class 2 at time 0.*

The proof of Lemmas 4.2.1 and 4.2.2 is to be found in [7], where they are formulated as Lemma 1 and Lemma 2, respectively. Using these results, we can now proceed with the proof of Proposition 4.2.1.

*Proof.* Consider the case ${}^{c_1\mu_1}/_{\beta_1} \geq {}^{c_2\mu_2}/_{\beta_2}$ and $c_1\mu_1 \leq c_2\mu_2$.
Suppose there exists a time $t_0$ such that $x_2(t_0) > f_2(x_1(t_0))$ and it is optimal to serve class 2 at time $t_0$. Using Remark 4.2.2 and the first part of Lemma 4.2.2, we deduce that it is optimal to prioritise class 2 until some time $t_1 \geq t_0$ where $x_2(t_1) \leq f_2(x_1(t_1))$, i.e. until $x_2$ drops below the switching curve $f_2(x_1)$. The first part of Lemma 4.2.2 also says that it is further optimal to assign full priority to class 1 until the class 1 queue is empty and then to assign it with partial priority until the origin $(0,0)$ is reached. This, however, is in contradiction with the Lemma 4.2.1 which states that class 1 with the lower $c\mu$-index cannot be given any priority while approaching the origin.
Therefore, it cannot be optimal to serve class 2 whenever $x_2 > f_2(x_1)$, i.e. in states separated from the origin by the switching curve (30) or its symmetric analogue $f_2$. This means that in such states it must be optimal to serve class 1 instead. Using the first part of Lemma 4.2.2, we obtain the necessary result.

Now consider the case ${}^{c_1\mu_1}/_{\beta_1} \geq {}^{c_2\mu_2}/_{\beta_2}$ and $c_1\mu_1 \geq c_2\mu_2$.
Suppose there exists a time $t_0$ and state $x(t_0)$ such that it is optimal to serve class 2 at time $t_0$. Then by Remark 4.2.2 and the second part of Lemma 4.2.2, it is optimal to fully prioritise class 2 until class 2 queue is empty. The second part of Lemma 4.2.2 states that it is further optimal to assign class 2 with partial priority to maintain $x_2 = 0$ and allocate the remaining service to class 1 until the origin is reached. This, however, is in contradiction with the Lemma 4.2.1 which states that class 1 with the higher $c\mu$-index should be given full and not partial priority while approaching the origin.
Therefore, it cannot be optimal to serve class 2 at time $t_0$ in state $x(t_0)$ so it must be optimal to serve class 1 instead. Applying the second part of Lemma 4.2.2 again, we obtain the necessary result. □

### 4.2.2 Overload case ($\rho > 1$)

We now consider Problem (29). Suppose that the customer classes are labelled such that ${}^{c_1\mu_1}/_{\beta_1} \geq {}^{c_2\mu_2}/_{\beta_2}$.

**Proposition 4.2.2.** *If $\rho > 1$ then an optimal solution $s^*$ to Problem* (29) *is given by*

$$s^*(t) = (\min\{\rho_1, 1\}, 1 - \min\{\rho_1, 1\}), \quad \textit{for all } t \in \mathbb{R}_+.$$

*Proof.* The goal of Problem (29) is to minimise the average cost over an infinite time horizon. Hence, to solve it to optimality it is sufficient to find a policy whose equilibrium point has minimum cost per unit time and show that this policy converges to this equilibrium from any starting state $x_0 \in \mathbb{R}^2_+$.

For a feasible policy $s$ with equilibrium point $x^s \in \mathbb{R}^2_+$ we have

$$\lambda_i = \mu_i s_i + \beta_i x_i^s \quad \text{for both } i \in \{1,2\} \tag{31}$$

Using (31), the minimisation objective in Problem (29) transforms to

$$\arg\min_s \left\{ \sum_{i=1}^{2} c_i x_i^s \right\} = \arg\min_s \left\{ \sum_{i=1}^{2} c_i \frac{\lambda_i - \mu_i s_i}{\beta_i} \right\} = \arg\max_s \left\{ \sum_{i=1}^{2} \frac{c_i \mu_i}{\beta_i} \cdot s_i \right\}, \tag{32}$$

which implies that an optimal policy must assign as much service attention to the queue with the highest $c\mu/\beta$-index (which in our case is queue 1) as it possibly can.
Observe that it cannot be feasible to assign $s_1 > \lambda_1/\mu_1 = \rho_1$ as equation (31) would lead to $x_1^s < 0$, which contradicts formulation (29). Thus for $\rho_1 < 1$, by equation (32), a policy $s = (\rho_1, 1 - \rho_1)$ is optimal as it maximises the amount of service attention given to queue 1 and does not contradict equation (31). Similarly, for $\rho_1 \geq 1$ a policy $s = (1, 0)$ maximises the service attention given to queue 1 and does not contradict equation (31).

We now show that each of the policies above converges to an equilibrium. For a starting state $x^* \in \mathbb{R}_+^2$ and policy $s = s(t)$, the dynamics of type-$i$ queue ($i = 1, 2$) for any $t \in \mathbb{R}_+$ is described by

$$\begin{cases} \frac{dx_i(t)}{dt} = \lambda_i - \mu_i s_i(t) - \beta_i x_i(t) \\ x_i(0) = x_i^*. \end{cases} \tag{33}$$

If $\rho_1 < 1$ then for the policy $s(t) = (\rho_1, 1 - \rho_1)$ the first equation of (33) for $i = 1$ reduces to $\frac{dx_1(t)}{dt} = -\beta_1 x_1(t)$. This implies that $x_1(t) = x_1^* e^{-\beta_1 t}$, which converges to 0 as $t \to \infty$. For the same policy $s$, the first equation of (33) for $i = 2$ is

$$\frac{dx_2(t)}{dt} = \lambda_2 - \mu_2(1 - \rho_1) - \beta_2 x_2(t).$$

This has the explicit solution

$$x_2(t) = \frac{\lambda_2 - \mu_2(1 - \rho_1)}{\beta_2} + \left( x_2^* - \frac{\lambda_2 - \mu_2(1 - \rho_1)}{\beta_2} \right) e^{-\beta_2 t},$$

which converges to $\frac{\lambda_2 - \mu_2(1-\rho_1)}{\beta_2}$ as $t \to \infty$ for any $x_2^*$.
We see that for policy $s = (\rho_1, 1 - \rho_1)$ we have $\lim_{t\to\infty} x(t) = \left(0, \frac{\lambda_2 - \mu_2(1-\rho_1)}{\beta_2}\right)$, which matches the equilibrium as prescribed by equation (31).

Similarly, for $\rho_1 > 1$ and policy $s = (1, 0)$, the first equation of (33) for $i = 1$ reduces to $\frac{dx_1(t)}{dt} = \lambda_1 - \mu_1 - \beta_1 x_1(t)$ with solution $x_1(t) \to \frac{\lambda_1 - \mu_1}{\beta_1}$ as $t \to \infty$ for any starting state $x_1^*$. For the same policy $s$, the first equation of (33) for $i = 2$ reads $\frac{dx_2(t)}{dt} = \lambda_2 - \beta_2 x_2(t)$ with solution $x_2(t) \to \frac{\lambda_2}{\beta_2}$ as $t \to \infty$ for any starting state $x_2^*$. These results also match the equilibrium point given by (31). □

### 4.2.3 Analysis of the fluid rule

To sum up the content of Subsections 4.2.1 and 4.2.2, the fluid heuristic can be described as follows.

- Suppose $\rho > 1$, i.e. system is overloaded. Then the heuristic follows the $c\mu/\beta$ rule.
- Suppose $\rho < 1$, i.e. system is underloaded. If the customer classes are labelled such that $c_1\mu_1 \geq c_2\mu_2$ and $c_1\mu_1/\beta_1 \geq c_2\mu_2/\beta_1$ then class 1 gets full priority under the fluid heuristic.
- Suppose $\rho < 1$. If the customer classes are labelled such that $c_1\mu_1 \leq c_2\mu_2$ and $c_1\mu_1/\beta_1 \geq c_2\mu_2/\beta_1$ then a switching curve given by (30) determines the optimal policy. Below this switching curve, fluid heuristic gives priority to class 2 and above this switching curve priority is given to class 1.

We now study the properties of the switching curve, which will be useful doing numerical experiments.

**Lemma 4.2.3.** *If $c_1\mu_1 \leq c_2\mu_2$ and $c_1\mu_1/\beta_1 \geq c_2\mu_2/\beta_1$ then the function $f_1$ given by* (30) *is decreasing in $x$ for $x \geq 0$.*

*Proof.* The proof of Lemma 4.2.3 is a matter of taking the derivative in (30) and taking care of the terms using the known inequalities. □

**Lemma 4.2.4.** *The switching curve* (30) *appears in the first quadrant if and only if* $(c_1\mu_1 - c_2\mu_2)\cdot\left(\frac{c_2\mu_2}{\beta_2} - \frac{c_1\mu_1}{\beta_1}\right) \geq 0$.

*Proof.* By Lemma 4.2.3, the switching curve is decreasing in $x$ so it appears in first quadrant if and only if $f_1(0) \geq 0$. As

$$f_1(0) = (1-\rho)\frac{\mu_1}{\beta_1\beta_2} \cdot \frac{c_2\mu_2 - c_1\mu_1}{c_1\mu_1/\beta_1 - c_2\mu_2/\beta_2}$$

then having $(1-\rho) > 0$ and $\mu_1/\beta_1\beta_2 > 0$ completes the proof. □

Finally, we numerically analyse the dependency of the switching curve (30) on the system load $\rho$. Fix $\beta = (0.1, 2.0)$, $c = (1, 9)$, $\mu = (15, 15)$ and adjust $\lambda = \lambda_1 = \lambda_2$. Results for $\lambda \in \{1, 3, 5, 7\}$ ($\rho \in \{\frac{2}{15}, \frac{6}{15}, \frac{10}{15}, \frac{14}{15}\}$) are shown in Figure 1.

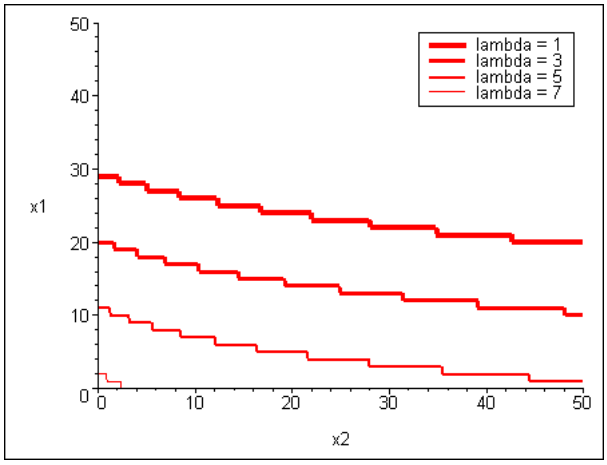


Figure 1: Switching curve (30) for $\beta = (0.1, 2.0)$, $c = (1, 9)$, $\mu = (15, 15)$ with different values of $\lambda$.

As $\rho$ increases to 1, the region where class 2 with lower $c\mu/\beta$ index is given priority shrinks in favor of class 1 with lower $c\mu$ index. Eventually, when $\rho \geq 1$ this region becomes empty thus the fluid rule fully follows the $c\mu/\beta$ rule and assigns full priority to class 1 with higher $c\mu/\beta$ index.

## 4.3 Numerical approximation

Our goal in this subsection is to compare the performance of the four heuristics above with the optimal policy. To do this, we give an explicit construction of an approximation scheme that we will later employ for numerical experiments.

### 4.3.1 Approximation scheme

As known approximation schemes work only for systems with a finite number of states, a state space truncation must be performed. Given desired precision $\epsilon \ll 1$, we are going to truncate a part of the state space, where the system will spend no more than an $\epsilon$-fraction of the time even if the server is kept constantly idle.

In such a system, each of the queues $i \in \{1, 2\}$ turns into an independent $M/M/\infty$ queue with arrival rate $\lambda_i$ and departure rate $\beta_i$. It is shown in [1] that the stationary distribution of an $M/M/\infty$

queue is Poisson with parameter $\lambda_i/\beta_i$. We therefore define the upper bounds of our state space by

$$x_i^{\max} := \max\left\{ n \in \mathbb{N} \quad \text{s.t.} \quad \sum_{j=0}^{n+1} \frac{\left(\frac{\lambda_i}{\beta_i}\right)^j}{j!} e^{-\lambda_i/\beta_i} < 1-\epsilon \right\} \quad \text{for } i \in \{1,2\}.$$

Given the finite state space

$$\mathcal{X} := \left\{ x \in \mathbb{N}_0^2 \quad \text{s.t.} \quad 0 \le x_i \le x_i^{\max} \quad \text{for all } i \in \{1,2\} \right\}$$

we compute the expected average running cost by the means of value iteration.

For all $x \in \mathcal{X}$ set $V_0(x) = 0$ and for $\mu := \max\{\mu_1, \mu_2\}$ define uniformisation parameter

$$R := \lambda_1 + \lambda_2 + \mu + \sum_{i=1}^{2} \beta_i x_i^{\max}.$$

We now begin the value iteration to determine the optimal running cost.

$$\begin{aligned}
V_{n+1}(0,0) = \quad & \tfrac{\lambda_1}{R} V_n(1,0) + \tfrac{\lambda_2}{R} V_n(0,1) + \tfrac{R-\lambda_1-\lambda_2}{R} V_n(0,0) \\
V_{n+1}(x_1^{\max},0) = \quad & c_1 x_1^{\max} + \tfrac{\lambda_2}{R} V_n(x_1^{\max},1) + \tfrac{\beta_1 x_1^{\max}}{R} V_n(x_1^{\max}-1,0) + \tfrac{\lambda_1+\mu+\beta_2 x_2^{\max}}{R} V_n(x_1^{\max},0) \\
V_{n+1}(0,x_2^{\max}) = \quad & c_2 x_2^{\max} + \tfrac{\lambda_1}{R} V_n(1,x_2^{\max}) + \tfrac{\beta_2 x_2^{\max}}{R} V_n(0,x_2^{\max}-1) + \tfrac{\lambda_2+\mu+\beta_1 x_1^{\max}}{R} V_n(0,x_2^{\max}) \\
V_{n+1}(x_1^{\max},x_2^{\max}) = \quad & c_1 x_1^{\max} + c_2 x_2^{\max} \\
& + \tfrac{\beta_1 x_1^{\max}}{R} V_n(x_1^{\max}-1,x_2^{\max}) + \tfrac{\beta_2 x_2^{\max}}{R} V_n(x_1^{\max},x_2^{\max}-1) \\
& + \tfrac{1}{R} \min \begin{Bmatrix} \mu_1 V_n(x_1^{\max}-1,x_2^{\max}) & \mu_2 V_n(x_1^{\max},x_2^{\max}-1) \\ +(\mu-\mu_1) V_n(x_1^{\max},x_2^{\max}), & +(\mu-\mu_2) V_n(x_1^{\max},x_2^{\max}) \end{Bmatrix} \\
& + \tfrac{\lambda_1+\lambda_2}{R} V_n(x_1^{\max},x_2^{\max})
\end{aligned}$$

for $i \in \{2,..,n_2^{\max}-1\}$ :

$$\begin{aligned}
V_{n+1}(i,0) = \quad & c_1 i + \tfrac{\lambda_1}{R} V_n(i+1,0) + \tfrac{\lambda_2}{R} V_n(i,1) + \tfrac{i\beta_1+\mu_1}{R} V_n(i-1,0) \\
& + \tfrac{\beta_1(x_1^{\max}-i)+\beta_2 x_2^{\max}+(\mu-\mu_1}{R} V_n(i,0) \\
V_{n+1}(i,x_2^{\max}) = \quad & c_1 i + c_2 x_2^{\max} \\
& + \tfrac{\lambda_1}{R} V_n(i+1,x_2^{\max}) + \tfrac{\beta_1 i}{R} V_n(i-1,x_2^{\max}) + \tfrac{\beta_2 x_2^{\max}}{R} V_n(i,x_2^{\max}-1) \\
& + \tfrac{1}{R} \min \begin{Bmatrix} \mu_1 V_n(i-1,x_2^{\max}) & \mu_2 V_n(i,x_2^{\max}-1) \\ +(\mu-\mu_1) V_n(i,x_2^{\max}), & +(\mu-\mu_2) V_n(i,x_2^{\max}) \end{Bmatrix} \\
& + \tfrac{\lambda_2+\beta_1(x_1^{\max}-i)}{R} V_n(i,x_2^{\max})
\end{aligned}$$

for $j \in \{2,..,n_2^{\max}-1\}$ :

$$\begin{aligned}
V_{n+1}(0,j) = \quad & c_2 j + \tfrac{\lambda_1}{R} V_n(1,j) + \tfrac{\lambda_2}{R} V_n(0,j+1) + \tfrac{j\beta_2+\mu_2}{R} V_n(0,j-1) \\
& + \tfrac{\beta_1 x_1^{\max}+\beta_2(x_2^{\max}-j)+(\mu-\mu_2)}{R} V_n(0,j) \\
V_{n+1}(x_1^{\max},j) = \quad & c_1 x_1^{\max} + c_2 j \\
& + \tfrac{\lambda_2}{R} V_n(x_1^{\max},j+1) + \tfrac{\beta_1 x_1^{\max}}{R} V_n(x_1^{\max}-1,j) + \tfrac{\beta_2 j}{R} V_n(x_1^{\max},j-1) \\
& + \tfrac{1}{R} \min \begin{Bmatrix} \mu_1 V_n(x_1^{\max}-1,j) & \mu_2 V_n(x_1^{\max},j-1) \\ +(\mu-\mu_1) V_n(x_1^{\max},j), & +(\mu-\mu_2) V_n(x_1^{\max},j) \end{Bmatrix} \\
& + \tfrac{\lambda_1+\beta_2(x_2^{\max}-j)}{R} V_n(x_1^{\max},j)
\end{aligned} \tag{34}$$

and, finally, for $i \in \{2,..,n_1^{\max}-1\}$ and $j \in \{2,..,n_1^{\max}-1\}$

$$\begin{aligned} V_{n+1}(i,j) = \quad & c_1 i + c_2 j \\ & + \tfrac{\lambda_1}{R} V_n(i+1,j) + \tfrac{\lambda_2}{R} V_n(i,j+1) + \tfrac{\beta_1 i}{R} V_n(i-1,j) + \tfrac{\beta_2 j}{R} V_n(i,j-1) \\ & + \tfrac{1}{R} \min \begin{Bmatrix} \mu_1 V_n(i-1,j) & \mu_2 V_n(i,j-1) \\ +(\mu-\mu_1) V_n(i,j)' & +(\mu-\mu_2) V_n(i,j) \end{Bmatrix} \\ & + \tfrac{\beta_1 (x_1^{\max}-i) + \beta_2 (x_2^{\max}-j)}{R} V_n(i,j). \end{aligned} \tag{35}$$

Using that for any $x \in \mathcal{X}$ and any $i \in \{1,2\}$

$$\lim_{n\to\infty} \left( V_n(x) - V_n(x-e_i)^+ \right) = V(x) - V(x-e_i)^+$$

$$\lim_{n\to\infty} \left( V_n(x) - V_{n-1}(x) \right) = g,$$

where $g$ is the unknown optimal long term running cost, for $D_n(x) := V_n(x) - V_{n-1}(x)$ one possible stopping criterion would be

$$\max_{x\in\mathcal{X}} |D_n(x) - D_{n-1}(x)| = \max_{x\in\mathcal{X}} |V_n(x) - 2V_{n-1}(x) + V_{n-2}(x)| < \epsilon.$$

However, as we have seen in practice, even if above criterion is filled, values of $\max_{x\in\mathcal{X}} D_n(x)$ and $\min_{x\in\mathcal{X}} D_n(x)$ can still be far apart meaning that not for all $x \in \mathcal{X}$ value of $D_n(x)$ has yet converged to $g$. Therefore, we use an additional stopping criterion

$$\left| \max_{x\in\mathcal{X}} D_n(x) - \min_{x\in\mathcal{X}} D_n(x) \right| = \left| \max_{x\in\mathcal{X}} \{V_n(x) - V_{n-1}(x)\} - \min_{x\in\mathcal{X}} \{V_n(x) - V_{n-1}(x)\} \right| < \epsilon.$$

To simulate each of the heuristics, we add additional binary control variables that will influence the minima picked in equations (34) and (35) to represent a certain type of action being always selected in a given state.

### 4.3.2 Numerical experiments

**Response to load**

We first see how each of the heuristics responds to varying level of load. To do this, we fix $\beta = (1.0, 0.4)$, $c = (2.5, 1.0)$, $\mu = (8, 12)$ and vary the arrival rates $\lambda_1 = \lambda_2$ between $0$ and $15$. We have intentionally selected $\beta$, $c$ and $\mu$ such that neither class has both $c\mu$ and $c\mu/\beta$ indices higher than the other class. In this case, a switching curve appears for the fluid heuristic and it does not trivially follow $c\mu$ and $c\mu/\beta$ rules. Numerical results for this experiment can be observed on Figure 2.

As the only variable here is $\lambda$, none of the stationary $c\mu$, $c\mu/\beta$ or $2U$ rules change the prioritised customer class; for the $c\mu$ and $2U$ rules this is class 1 and for the $c\mu/\beta$ rule this is class 2.
The $c\mu$ and $2U$ rules that do local minimisation, perform rather well in underload scenarios ($\lambda < 4.8$), but as the load increases they are far from optimal. On the contrary, the $c\mu/\beta$ is the worst of considered heuristics for this specific case in underload. However, as $\rho$ goes past 1, the system tends to spend more time in the states far from the origin, where $c\mu/\beta$-rule was argued in Subsection 4.1.2 to be approaching optimality.
Because the mean number of customers in the system is relatively low in underload, the switching curve only starts to play a role for the fluid heuristic for $\rho$ values close to 1. As $\rho$ approaches $1$ (or $\lambda$ approachies $4.8$), the switching curve in the fluid heuristic approaches the origin and class 2 is given priority in an increasing number of states. Thus the cost of the fluid heuristic moves from the cost of $c\mu$ that prioritises class 1 towards the cost of $c\mu/\beta$ that prioritises class 2. Eventually, for $\rho > 1$ the fluid heuristic copies the $c\mu/\beta$-rule completely and serves class 2 only.

**Response to cost**

Looking at the stationary heuristics, one can observe that the criteria used for priority assignment are linear in the running costs. Thus, it could be useful to pick a scenario and adjust the holding costs to

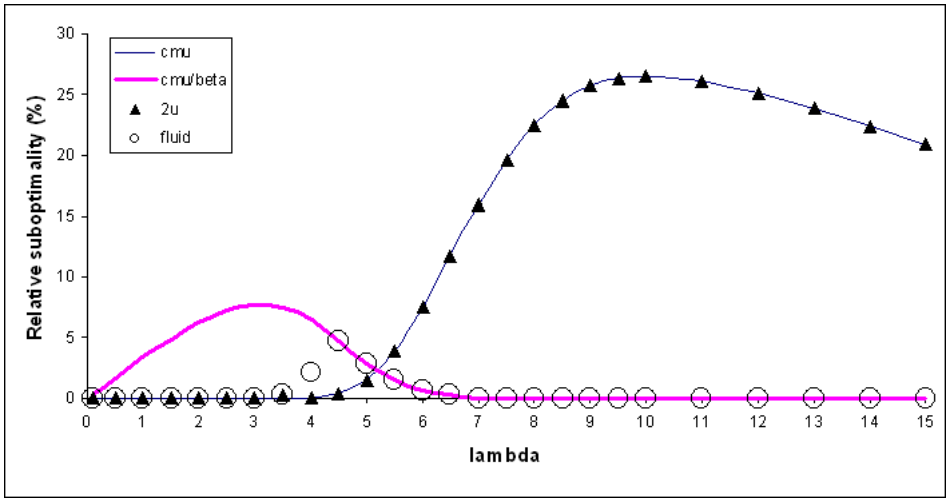


Figure 2: Relative suboptimality gap for $\beta = (1.0, 0.4)$, $c = (2.5, 1.0)$, $\mu = (8, 12)$ with varying $\lambda_1 = \lambda_2$.

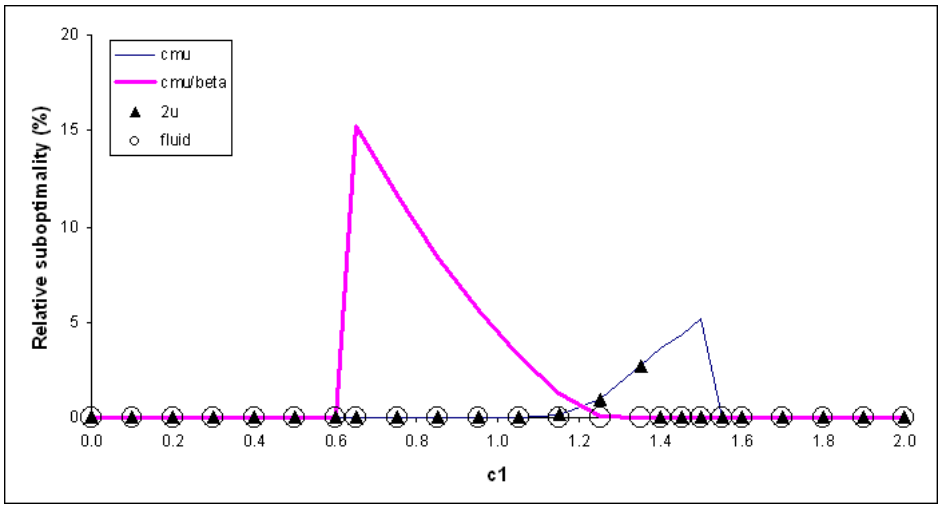


Figure 3: Relative suboptimality gap for $\beta = (0.4, 1.0)$, $c_2 = 1$, $\lambda = (2, 3)$ and $\mu = (8, 12)$ with varying $c_1$.

see at which point each of the policies makes a priority change and what consequences for the expected running cost are. We fix $\beta = (0.4, 1.0)$, $c_2 = 1$, $\lambda = (2, 3)$ and $\mu = (8, 12)$ and vary $c_1$ between 0 and 2. Numerical results for this experiment are shown in Figure 3.

With this combination of fixed parameters, the $c\mu$ rule will prioritise class 2 for $c_1 \in [0, 1.5]$, the $c\mu/\beta$ rule will prioritise class 2 for $c_1 \in [0, 0.6]$ and the $2U$ rule will proritise class 2 for $c_1 \in [0, {}^{837}/_{667} \sim 1.25]$. As shown above, prioritising the wrong class can make the policy deviate from the optimum and with certain combinations of the input parameters the suboptimality gap can be relatively high.
For $c_1 \in [0.6, 1.5]$, neither of the classes has $c\mu$ and $c\mu/\beta$ indices higher than the other class, thus a switching curve appears in the fluid heuristic, which greatly improves its performance in comparison to all static rules.

To have a better view on how much serving the wrong class of customer affects the suboptimality gap, we modify our previous settings by increasing the arrival rate $\lambda_1$ from 2 to 16. This way, the expected number of unserved type-1 customers in the system will increase so the $c\mu$ and $2U$ heuristics that prioritise class 2 will perform worse.
We pursue a similar goal for the $c\mu/\beta$ heuristic that wrongly prioritises class 1 for certain values of $c_1$.

However, we cannot pick an arbitrarily high value for $\lambda_2$ as this will put the system in overloaded regime, where we know the $c\mu/\beta$-rule to perform well. We therefore pick $\lambda_2 = 8$ such that $\rho = {}^{11}/_{12} < 1$. Experiment results on modified instances with $c_1$ range limited to $[0.55, 1.55]$ can be seen in Figure 3.

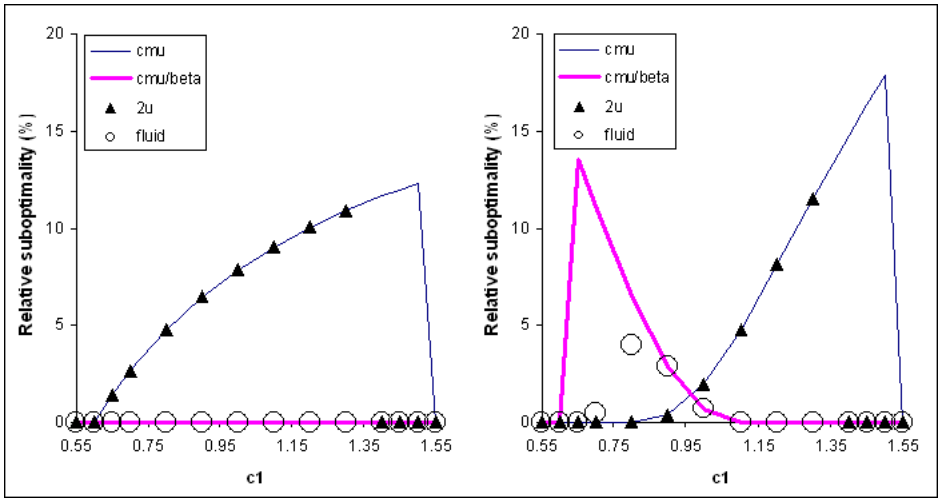


Figure 4: Relative suboptimality gap for $\beta = (0.4, 1.0)$, $c_2 = 1$, $\mu = (8, 12)$, $\lambda = (16, 3)$ (left) or $\lambda = (2, 8)$ (right) with varying $c_1$.

The case with increased $\lambda_1$ in Figure 4a corresponds to $\rho = {}^{9}/_{4} > 1$ so the performance of the $c\mu/\beta$ and fluid rules that prioritise class 1 throughout most of the range is either optimal or near-optimal. The same cannot be said about the $c\mu$ and $2U$ rules that prioritise class 2. Along with $\lambda_1$, the number of new class 1 arrivals has gone up. But as the $c\mu$ and $2U$ rules do no adjustments for this increased inflow of type-1 customers, we see that their suboptimality gaps increase from $3 - 4\%$ to $11 - 12\%$.
This also hints that suboptimality gap for the $c\mu$ rule can potentially be unbounded. Suppose we have two queues such that $c_1\mu_1 < c_2\mu_2$. Suppose that $\beta_1 = 0$, $\beta_2 \gg \mu_2$ and $\lambda_2$ is such that class 2 will occupy most of server's time when given full priority under the $c\mu$ rule. As $\beta_1 = 0$ and class 1 will get very little service time under the $c\mu$ rule then the length of queue 1 will explode inducing potentially unbounded holding costs. However, if a policy was to switch its focus to prioritising class 1 then due to $\beta_2 \gg \mu_2$ its loss due to disprioritising class 2 would be much smaller than cost gained by prioritising class 1.

In the case with increased $\lambda_2$ in Figure 4b, the suboptimality gap and range of the $c\mu/\beta$ heuristic did not increase, contrary to what we have expected to see. This can be explained by increased $\rho$, which generally improves the performance of the $c\mu/\beta$ heuristic, compensating for the loss associated with the increase in arrival rate of unprioritised type-2 arrivals.
What is also unexpected is that an increase in type-2 arrival rate has increased the suboptimality gap of the $c\mu$ and $2U$ heuristics, both of which prioritise class 2 throughout most of the $c_1$ range.

**Response to impatience rate adjustment**

Finally, we evaluate the performance of each heuristic for a series of instances with varying abandonment rates. We fix $\beta_2 = 0.5$, $c = (1.25, 1.0)$, $\lambda = (2, 3)$, $\mu = (8, 8)$ and vary $\beta_1$ between $0.1$ and $4.0$.

As the $c\mu$ rule does not respond to the changes in abandonments, it will prioritise class 1 throughout the whole testing range. The $c\mu$ and $2U$ rules will prioritise class 1 only for $\beta_1 \in [0.1, {}^{5}/_{8}]$ and $\beta_1 \in [0.1, {}^{21}/_{8}]$ respectively, outside which they prioritise class 2.
The numerical results of this experiment can be seen on Figure 5.

The $c\mu$ heuristic that is known to be optimal for the case with no abandonments also performs well when abandonment rates are low. However, when the abandonment rate of the prioritised class increases, the $c\mu$ rule becomes increasingly suboptimal.

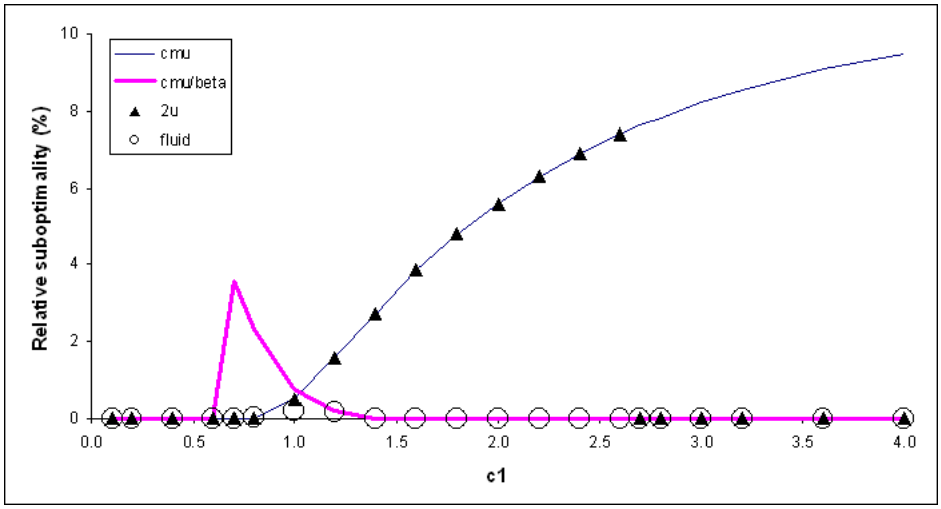


Figure 5: Relative suboptimality gap for $\beta_2 = 0.5$, $c = (1.25, 1.0)$, $\lambda = (2, 3)$, $\mu = (8, 8)$ with varying $\beta_1$.

Because the $2U$ heuristic is heavily dependent on $c\mu$ ratio, it might also prioritise the wrong class of customer. However, unlike the $c\mu$ rule that is not able to react to the changes in abandonment rates, $\beta$ terms in the denominators in (27) also play a role. For $\beta_1$ high enough the $2U$ rule eventually switches priority to class 2 thus improving performance over the $c\mu$ rule that stays with class 1.
For $\beta_1 \in [5/8, 4]$ we have $c_1\mu_1 \geq c_2\mu_2$ and $c_1\mu_1/\beta_1 \leq c_2\mu_2/\beta_2$ so a switching curve appears in the fluid heuristic, which gradually shifts its priority from customer class 1 to class 2.

## 4.4 Summary and conclusions

From our numerical experiments, we are able to draw a number of conclusions.

Firstly, the $c\mu$ heuristic generally performs well on instances where the abandonment rates are low. However, as this heuristic ignores the abandonment rates, its suboptimality gap is potentially unbounded.

Secondly, the $c\mu/\beta$ heuristic generally performs well on instances with high system load. This is due to the fact that the optimal solution approaches the $c\mu/\beta$ rule in system states, where both queue lengths are relatively high.

Thirdly, the $2U$ heuristic is closer to optimality than the $c\mu$ heuristic as it takes abandonment rates into consideration. However, there still exist instances where its performance is far from optimal.

Finally, the fluid heuristic is the best of all heuristics reviewed in this thesis. Its good performance in underloaded systems can be explained by taking all factors, including the current system state, into account and in systems with overload its performance is due to copying the $c\mu/\beta$ rule.

# A Appendix

## Maple code for computing optimal and heuristic costs in Section 4.3

```
> restart:
  with(linalg):
  with(LinearAlgebra):
> beta1:=0.4:
  beta2:=1.0:
  c1:=1:
  c2:=1.0:
  lambda1:=16:
  lambda2:=3:
  mu1:=8:
  mu2:=12:
  mu:=max(mu1,mu2):
  prec:=0.0001:
> n1max:=0:
  n2max:=0:
  sumprob1:=0:
  sumprob2:=0:
  while evalf(sumprob1)<1-prec do
     sumprob1:=sumprob1+((lambda1/beta1)^n1max*exp(-lambda1/beta1))/n1max!:
     n1max:=n1max+1
  end:
  while evalf(sumprob2)<1-prec do
     sumprob2:=sumprob2+((lambda2/beta2)^n2max*exp(-lambda2/beta2))/n2max!:
     n2max:=n2max+1:
  end:
  sumrates:=lambda1+lambda2+(n1max-1)*beta1+(n2max-1)*beta2+mu:
  M0:=Matrix(n1max,n2max):
  M1:=Matrix(n1max,n2max):
  DM1:=Matrix(n1max,n2max):
  DM2:=Matrix(n1max,n2max):
  DDM:=Matrix(n1max,n2max):
> for i from 1 to n1max do
     for j from 1 to n2max do
        M0[i,j]:=0:
        DM1[i,j]:=0:
        DM2[i,j]:=0:
     end:
  end:
  ind:=1:
  DDM[1,1]:=-sumrates:
  DM1[1,1]:=-sumrates:
  DM2[1,1]:=-sumrates:
  while (max(DM1)-min(DM1)>prec/sumrates and max(DM2)-min(DM2)>prec/sumrates) do
     M1[1,1]:=(lambda1*M0[2,1]+lambda2*M0[1,2]
              +((n1max-1)*beta1+(n2max-1)*beta2+mu)*M0[1,1])
             /sumrates:
     M1[n1max,1]:=(c1*(n1max-1)
                  +lambda2*M0[n1max,2]+(n1max-1)*beta1*M0[n1max-1,1]+mu1*M0[n1max-1,1]
                  +(lambda1+(n2max-1)*beta2+(mu-mu1))*M0[n1max,1])
                  /sumrates:
     M1[1,n2max]:=(c2*(n2max-2)
                  +lambda1*M0[2,n2max]+(n2max-1)*beta2*M0[1,n2max-1]+mu2*M0[1,n2max-1]
                  +(lambda2+(n1max-1)*beta1+(mu-mu2))*M0[1,n2max])
                  /sumrates:
     M1[n1max,n2max]:=(c1*(n1max-1)+c2*(n2max-1)
                      +(n1max-1)*beta1*M0[n1max-1,n2max]+(n2max-1)*beta2*M0[n1max,n2max-1]
                      +min(mu1*M0[n1max-1,n2max]+(mu-mu1)*M0[n1max,n2max],
                           mu2*M0[n1max,n2max-1]+(mu-mu2)*M0[n1max,n2max])
                      +(lambda1+lambda2)*M0[n1max,n2max])
                      /sumrates:
     for i from 2 to n1max-1 do
        M1[i,1]:=(c1*(i-1)
                 +lambda1*M0[i+1,1]+lambda2*M0[i,2]+(i-1)*beta1*M0[i-1,1]+mu1*M0[i-1,1]
                 +((n1max-i)*beta1+(n2max-1)*beta2+(mu-mu1))*M0[i,1])
                 /sumrates:
        M1[i,n2max]:=(c1*(i-1)+c2*(n2max-1)
```

```
                    +lambda1*M0[i+1,n2max]+(i-1)*beta1*M0[i-1,n2max]+(n2max-1)*beta2*M0[i,n2max-1]
                    +min(mu1*M0[i-1,n2max]+(mu-mu1)*M0[i,n2max],mu2*M0[i,n2max-1]+(mu-mu2)*M0[i,n2max])
                    +(lambda2+(n1max-i)*beta1)*M0[i,n2max])
                    /sumrates:
    end:
    for j from 2 to n2max-1 do
       M1[1,j]:=(c2*(j-1)
                +lambda1*M0[2,j]+lambda2*M0[1,j+1]+(j-1)*beta2*M0[1,j-1]+mu2*M0[1,j-1]
                +((n1max-1)*beta1+(n2max-j)*beta2+(mu-mu2))*M0[1,j])
                /sumrates:
       M1[n1max,j]:=(c1*(n1max-1)+c2*(j-1)
                    +lambda2*M0[n1max,j+1]+(n1max-1)*beta1*M0[n1max-1,j]+(j-1)*beta2*M0[n1max,j-1]
                    +min(mu1*M0[n1max-1,j]+(mu-mu1)*M0[n1max,j],mu2*M0[n1max,j-1]+(mu-mu2)*M0[n1max,j])
                    +(lambda1+(n2max-j)*beta2)*M0[n1max,j])
                    /sumrates:
    end:
    for i from 2 to n1max-1 do
       for j from 2 to n2max-1 do
          M1[i,j]:=(c1*(i-1)+c2*(j-1)
                   +lambda1*M0[i+1,j]+lambda2*M0[i,j+1]+(i-1)*beta1*M0[i-1,j]+(j-1)*beta2*M0[i,j-1]
                   +min(mu1*M0[i-1,j]+(mu-mu1)*M0[i,j],mu2*M0[i,j-1]+(mu-mu2)*M0[i,j])
                   +((n1max-i)*beta1+(n2max-j)*beta2)*M0[i,j])
                   /sumrates:
          end:
    end:
    if ind=1 then
                  for i from 1 to n1max do
                     for j from 1 to n2max do
                        DM1[i,j]:=M1[i,j]-M0[i,j]
                     end:
                  end:
             else
                  for i from 1 to n1max do
                     for j from 1 to n2max do
                        DM2[i,j]:=M1[i,j]-M0[i,j]
                     end:
                  end:
    end if:
    for i from 1 to n1max do
       for j from 1 to n2max do
          DDM[i,j]:=DM1[i,j]-DM2[i,j]:
          M0[i,j]:=M1[i,j]
       end:
    end:
    ind:=-ind:
  end:
  opt:=0:
  for i from 1 to n1max do
     for j from 1 to n2max do
        opt:=opt+sumrates*(DM1[i,j]+DM2[i,j])/(n1max*n2max*2)
     end:
  end:
> cost:=proc(CTRL::Matrix)::real:
    M0:=Matrix(n1max,n2max):
    M1:=Matrix(n1max,n2max):
    DM1:=Matrix(n1max,n2max):
    DM2:=Matrix(n1max,n2max):
    DDM:=Matrix(n1max,n2max):
    for i from 1 to n1max do
       for j from 1 to n2max do
          M0[i,j]:=0:
          DM1[i,j]:=0:
          DM2[i,j]:=0:
          DDM[i,j]:=0:
       end:
    end:
    ind:=1:
    DDM[1,1]:=-sumrates:
    DM1[1,1]:=-sumrates:
    DM2[1,1]:=-sumrates:
```

```
while (max(DM1)-min(DM1)>prec/sumrates and max(DM2)-min(DM2)>prec/sumrates) do
   M1[1,1]:=(lambda1*M0[2,1]+lambda2*M0[1,2]
            +((n1max-1)*beta1+(n2max-1)*beta2+mu)*M0[1,1])
            /sumrates:
   M1[n1max,1]:=(c1*(n1max-1)
                 +lambda2*M0[n1max,2]+(n1max-1)*beta1*M0[n1max-1,1]+mu1*M0[n1max-1,1]
                 +(lambda1+(n2max-1)*beta2+(mu-mu1))*M0[n1max,1])
                 /sumrates:
   M1[1,n2max]:=(c2*(n2max-2)
                 +lambda1*M0[2,n2max]+(n2max-1)*beta2*M0[1,n2max-1]+mu2*M0[1,n2max-1]
                 +(lambda2+(n1max-1)*beta1+(mu-mu2))*M0[1,n2max])
                 /sumrates:
   M1[n1max,n2max]:=(c1*(n1max-1)+c2*(n2max-1)
                     +(n1max-1)*beta1*M0[n1max-1,n2max]+(n2max-1)*beta2*M0[n1max,n2max-1]
                     +CTRL[n1max,n2max]*(mu1*M0[n1max-1,n2max]+(mu-mu1)*M0[n1max,n2max])
                     +(1-CTRL[n1max,n2max])*(mu2*M0[n1max,n2max-1]+(mu-mu2)*M0[n1max,n2max])
                     +(lambda1+lambda2)*M0[n1max,n2max])
                     /sumrates:
   for i from 2 to n1max-1 do
      M1[i,1]:=(c1*(i-1)
               +lambda1*M0[i+1,1]+lambda2*M0[i,2]+(i-1)*beta1*M0[i-1,1]+mu1*M0[i-1,1]
               +((n1max-i)*beta1+(n2max-1)*beta2+(mu-mu1))*M0[i,1])
               /sumrates:
      M1[i,n2max]:=(c1*(i-1)+c2*(n2max-1)
                   +lambda1*M0[i+1,n2max]+(i-1)*beta1*M0[i-1,n2max]+(n2max-1)*beta2*M0[i,n2max-1]
                   +CTRL[i,n2max]*(mu1*M0[i-1,n2max]+(mu-mu1)*M0[i,n2max])
                   +(1-CTRL[i,n2max])*(mu2*M0[i,n2max-1]+(mu-mu2)*M0[i,n2max])
                   +(lambda2+(n1max-i)*beta1)*M0[i,n2max])
                   /sumrates:
   end:
   for j from 2 to n2max-1 do
      M1[1,j]:=(c2*(j-1)
               +lambda1*M0[2,j]+lambda2*M0[1,j+1]+(j-1)*beta2*M0[1,j-1]+mu2*M0[1,j-1]
               +((n1max-1)*beta1+(n2max-j)*beta2+(mu-mu2))*M0[1,j])
               /sumrates:
      M1[n1max,j]:=(c1*(n1max-1)+c2*(j-1)
                   +lambda2*M0[n1max,j+1]+(n1max-1)*beta1*M0[n1max-1,j]+(j-1)*beta2*M0[n1max,j-1]
                   +CTRL[n1max,j]*(mu1*M0[n1max-1,j]+(mu-mu1)*M0[n1max,j])
                   +(1-CTRL[n1max,j])*(mu2*M0[n1max,j-1]+(mu-mu2)*M0[n1max,j])
                   +(lambda1+(n2max-j)*beta2)*M0[n1max,j])
                   /sumrates:
   end:
   for i from 2 to n1max-1 do
      for j from 2 to n2max-1 do
         M1[i,j]:=(c1*(i-1)+c2*(j-1)
                  +lambda1*M0[i+1,j]+lambda2*M0[i,j+1]+(i-1)*beta1*M0[i-1,j]+(j-1)*beta2*M0[i,j-1]
                  +CTRL[i,j]*(mu1*M0[i-1,j]+(mu-mu1)*M0[i,j])
                  +(1-CTRL[i,j])*(mu2*M0[i,j-1]+(mu-mu2)*M0[i,j])
                  +((n1max-i)*beta1+(n2max-j)*beta2)*M0[i,j])
                  /sumrates:
      end:
   end:
   if ind=1 then
                  for i from 1 to n1max do
                     for j from 1 to n2max do
                        DM1[i,j]:=M1[i,j]-M0[i,j]
                     end:
                  end:
            else
                  for i from 1 to n1max do
                     for j from 1 to n2max do
                        DM2[i,j]:=M1[i,j]-M0[i,j]
                     end:
                  end:
   end if:
   for i from 1 to n1max do
      for j from 1 to n2max do
         DDM[i,j]:=DM1[i,j]-DM2[i,j]:
         M0[i,j]:=M1[i,j]
      end:
```

```
      end:
      ind:=-ind:
   end:
   a:=0:
   for i from 1 to n1max do
      for j from 1 to n2max do
         a:=a+sumrates*(DM1[i,j]+DM2[i,j])/(n1max*n2max*2)
      end:
   end:
   cost:=a
end proc:
> CM:=Matrix(n1max,n2max):
if c1*mu1>c2*mu2 then
   for i from 1 to n1max do
      for j from 1 to n2max do
         CM[i,j]:=1
      end:
   end:
                else
   for i from 1 to n1max do
      for j from 1 to n2max do
         CM[i,j]:=0
      end:
   end:
end if:
h1cost:=cost(CM):
> CMB:=Matrix(n1max,n2max):
if c1*mu1/beta1>c2*mu2/beta2 then
  for i from 1 to n1max do
      for j from 1 to n2max do
         CMB[i,j]:=1
      end:
   end:
                else
   for i from 1 to n1max do
      for j from 1 to n2max do
         CMB[i,j]:=0
      end:
   end:
end if:
> h2cost:=cost(CMB):
R2U:=Matrix(n1max,n2max):
if c1*mu1/((beta1+mu1)*(beta1+beta2+mu2))>c2*mu2/((beta2+mu2)*(beta1+beta2+mu1)) then
   for i from 1 to n1max do
      for j from 1 to n2max do
         R2U[i,j]:=1
      end:
   end:
                else
   for i from 1 to n1max do
      for j from 1 to n2max do
         R2U[i,j]:=0
      end:
   end:
end if:
h3cost:=cost(R2U):
> FLD:=Matrix(n1max,n2max):
b1:=c1*mu1*(1-lambda1/mu1-lambda2/mu2)/beta1:
b2:=b1*mu2*(1-lambda2/mu2)/beta2:
b3:=(c2*mu2/beta2-c1*mu1/beta1)*(1-lambda2/mu2):
b4:=(c1*mu1/beta1-c2*mu2/beta2)*beta1/mu1:
f1:=x->lambda1/beta1+(b1*x+b2+(b3*x-b2)*(1+beta2*x/(mu2-lambda2))^(beta1/beta2))/(b4*x):
a1:=c2*mu2*(1-lambda1/mu1-lambda2/mu2)/beta2:
a2:=a1*mu1*(1-lambda1/mu1)/beta1:
a3:=(c1*mu1/beta1-c2*mu2/beta2)*(1-lambda1/mu1):
a4:=(c2*mu2/beta2-c1*mu1/beta1)*beta2/mu2:
f2:=x->lambda2/beta2+(a1*x+a2+(a3*x-a2)*(1+beta1*x/(mu1-lambda1))^(beta2/beta1))/(a4*x):
if c1*mu1/beta1>c2*mu2/beta2 then
                                  if c1*mu1>c2*mu2 then
                                                       for i from 1 to n1max do
```

```
                                                  for j from 1 to n2max do
                                                     FLD[i,j]:=1
                                                  end:
                                               end:
                                          else
                                               for i from 2 to n1max do
                                                  for j from 2 to n2max do
                                                     if i-1>f1(j-1) then FLD[i,j]:=1:
                                                                    else FLD[i,j]:=0:
                                                     end if:
                                                  end:
                                               end:
                                     end if:
                                else
                                     if c1*mu1>c2*mu2 then
                                               for i from 2 to n1max do
                                                  for j from 2 to n2max do
                                                     if j-1>f2(i-1) then FLD[i,j]:=0:
                                                                    else FLD[i,j]:=1:
                                                     end if:
                                                  end:
                                               end:
                                          else
                                               for i from 1 to n1max do
                                                  for j from 1 to n2max do
                                                     FLD[i,j]:=0:
                                                  end:
                                               end:
                                     end if:
  end if:
  h4cost:=cost(FLD):
> evalf(opt,6);
  evalf(h1cost,6);
  evalf(h2cost,6);
  evalf(h3cost,6);
  evalf(h4cost,6);
```

# References


[1] I. Adan and J.Resing, "Queueing Theory," 2002.

[2] U. Ayesta, P. Jacko and V. Novak, "A Nearly-Optimal Index Rule for Scheduling of Users with Abandonment," 2011.

[3] J.S. Baras, D.-J. Ma and A.M. Makowski, "K competing queues with geometric service requirements and linear costs: The $\mu c$-rule is always optimal," *Systems & Control Letters*, Vol. 6, No. 3, pp. 173-180, August 1985.

[4] S. Bhulai, H. Blok and F.M. Spieksma, "K competing queues with customer abandonment: optimality of a generalized $c\mu$-rule by the Smoothed Rate Truncation method", 2013.

[5] C. Buyukkoc, P. Varaiya and J. Walrand, "The $c\mu$ Rule Revisited," *Advances in Applied Probability*, Vol. 17, No. 1, pp. 237-238, March 1985.

[6] D.G. Down, G. Koole and M.E. Lewis, "Dynamic control of a single-server system with abandonments," *Queueing systems*, Vol. 67, pp. 63-90, 2010.

[7] M. Larranaga, U. Ayesta and I.M. Verloop, "Dynamic fluid-based scheduling in a multi-cass abandonment queue," 2013.

[8] M.L. Puterman, "Markov Decision Processes." Wiley, 1994.

[9] M. Sidi, "Two competing discrete-time queues with priority," *Queueing Systems*, Vol 3, pp. 347-362, 1988.

[10] J.A. van Mieghem, "Dynamic scheduling with convex delay costs: The generalized $c\mu$ rule", *The Annals of Applied Probability*, Vol. 5, Issue 3, pp. 809-833, August 1995.