A Simple yet Exact Analysis of the MultiQueue

Stefan Walzer
Karlsruhe Institute of Technology, Germany

Marvin Williams
Karlsruhe Institute of Technology, Germany

Abstract

The MultiQueue is a relaxed concurrent priority queue consisting of n internal priority queues, where an insertion uses a random queue and a deletion considers two random queues and deletes the minimum from the one with the smaller minimum. The *rank error* of the deletion is the number of smaller elements in the MultiQueue.

Alistarh et al. [2] have demonstrated in a sophisticated potential argument that the expected rank error remains bounded by $\mathcal{O}(n)$ over long sequences of deletions.

In this paper we present a simpler analysis by identifying the stable distribution of an underlying Markov chain and with it the long-term distribution of the rank error exactly. Simple calculations then reveal the expected long-term rank error to be $\frac{5}{6}n - 1 + \frac{1}{6n}$. Our arguments generalize to deletion schemes where the probability to delete from a given queue depends only on the rank of the queue. Specifically, this includes deleting from the best of c randomly selected queues for any $c > 1$.

2012 ACM Subject Classification Mathematics of computing → Stochastic processes; Theory of computation → Data structures design and analysis

Keywords and phrases MultiQueue, concurrent data structure, stochastic process, Markov chain

Digital Object Identifier 10.4230/LIPIcs.ESA.2025.85

Related Version *Full Version*: `https://arxiv.org/abs/2410.08714` [20]

Funding This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No. 882 500).

1 Introduction

Priority queues maintain a set of elements from a totally ordered domain, with an *insert* operation adding an element and a *deleteMin* operation extracting the smallest element. They are a fundamental building block for a wide range of applications such as task scheduling, graph algorithms, and discrete event simulation. The parallel nature of modern computing hardware motivates the design of concurrent priority queues that allow multiple processing elements to insert and delete elements concurrently. *Strict*[1] concurrent priority queues (e.g., [19, 11, 7, 17]) suffer from poor scalability due to inherent contention on the smallest element [4, 8]. To alleviate this contention, in *relaxed* priority queues deletions should still preferentially extract small elements but need not always extract the minimum. In other words, the correctness requirement is turned into a quality measure: We speak of a *rank error* of $r - 1$ if the extracted element has rank r among all elements currently in the priority queue. In many scenarios, relaxed priority queues outperform strict priority queues, as the higher scalability outweighs the additional work caused by the relaxation. They are an active field of research and a vast range of designs has been proposed [10, 16, 22, 3, 18, 21, 15, 23].

[1] in the sense of *linearizability*

The MultiQueue. The MultiQueue, initially proposed by Rihani et al. [16] and improved upon by Williams et al. [21], emerged as the state-of-the art relaxed priority queue. Due to its high scalability and robust quality, the MultiQueue inspired a number of follow-up works [15, 23]. Its design uses the *power-of-two-choices* paradigm and is delightfully simple: We use n (sequential) priority queues for some fixed $n \in \mathbb{N}$. Each insertion adds its element to a queue chosen uniformly at random and each deletion picks *two* queues uniformly at random and deletes from the one with the smaller minimum. Figure 1 illustrates the MultiQueue

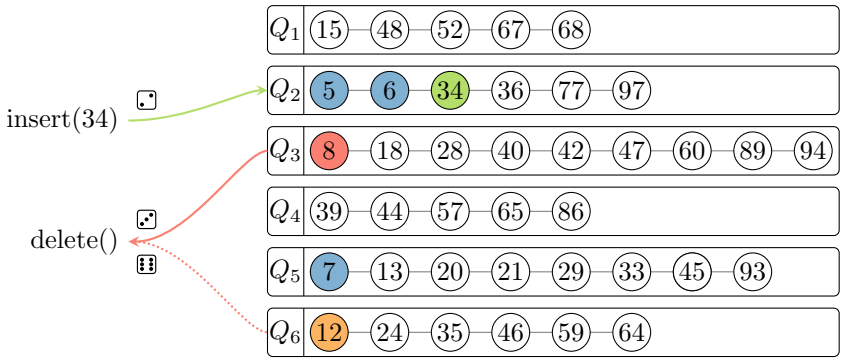

Figure 1 The MultiQueue with $n = 6$ queues and some elements already inserted. The shown insertion picks queue 2 to insert the element 34 (green). The shown deletion picks queues 3 and 6 with minima 8 (red) and 12 (orange), and deletes the 8 since it is smaller. The deletion exhibits a rank error of 3 due to the 3 smaller elements highlighted in blue.

with $n = 6$ queues. We can generalize this design to pick any number $c > 1$ of queues for deletions where even non-integer choices of c make sense: We would then pick $\lfloor c \rfloor + 1$ queues with probability $c - \lfloor c \rfloor$ and $\lfloor c \rfloor$ queues otherwise. In practice, the individual queues are protected by mutual exclusion locks, and n is proportional to the number of processing elements to find unlocked queues in (expected) constant time.

Existing theory on the MultiQueue. Our theoretical understanding of the MultiQueue is still incomplete. One obstacle is that the order of operations matters: Intuitively, deletions can cause the distribution of elements to drift apart and increase the expected rank error, while insertions of small elements can mask accumulated differences. This suggests that the worst-case setting is when following the insertion of sufficiently many elements, only deletions occur. Like Alistarh et al. [2], we exclusively consider this setting. At first glance, this process seems closely related to the classical balls-into-bins process, where balls are placed one after the other into the least loaded of two randomly chosen bins. Famously, the difference between the highest load and the average load for n bins is in $\mathcal{O}(\log \log n)$ with high probability for any number of balls. Numerous variants of the balls-into-bins process have been proposed and studied [5, 12, 6, 14]. However, reducing the process to a balls-into-bins process imposes multiple difficulties. The state of the MultiQueue is not fully described by the number of elements in each queue but also involves information about the ranks of the elements. Moreover, deleting an element from a queue can affect the ranks of elements in other queues. Despite these challenges, Alistarh et al. [2] managed to transfer the potential argument from the balls-into-bins analysis by Peres et al. [14] to a MultiQueue analysis via an intermediate "exponential process" that avoids correlations between the elements in the queues. They prove that the expected rank error is in $\mathcal{O}(n)$ and the expected worst-case rank error is in $\mathcal{O}(n \log n)$ for any number of deletions and any $c \in (1, 2]$, while the rank errors diverge

with the number of deletions for $c = 1$. In follow-up work, this technique is generalized to other relaxed concurrent data structures [1] and a process where each processing element has its own priority queue and "steals" elements from other processing elements with some probability [15].

Contribution. In this paper, we present an analysis of the MultiQueue that, compared to the potential argument by Alistarh et al. [2], is simultaneously simpler and more precise. We characterise the exact long-term distribution of the rank error for any $c > 1$ and any $n \in \mathbb{N}$. From this distribution we can derive, for instance, that the expected rank error for $c = 2$ is $\frac{5}{6}n - 1 + \frac{1}{6n}$. In addition to covering any $c > 1$, our analysis generalizes to all deletion schemes where the probability for a queue to be selected solely depends on the rank of the queue (when ordering the queues according to their smallest element).

We achieve this by modelling the deletion process as a Markov chain and observing that its stationary distribution can be described using a sequence of $n - 1$ independent geometrically distributed random variables. In the full version of this paper [20] we also apply our techniques to the elegant exponential process by Alistarh et al. [2], which we believe to be of independent interest.

Outline. First, in Section 2, we introduce the formal model of a generalized MultiQueue and the setting in which we analyze it. We then state our main results, including Theorem 1 that characterizes the long-term distribution of the ranks of the top-elements in the MultiQueue. In Section 3, we present our analysis of the generalized MultiQueue, leading to the proof of Theorem 1. We apply Theorem 1 to compute the expected rank error for any $c > 1$ and any $n \in \mathbb{N}$ in Section 4. Finally, in Section 5, we discuss implications and limitations of our results and outline possible future work.

2 Formal Model and Results

Analogous to Alistarh et al. [2], we analyze the MultiQueue in a simplified setting where only deletions occur.

The σ-MultiQueue. A σ-MultiQueue consists of n priority queues and a choice distribution σ on $[n]$ that captures a generalised deletion strategy (as explained below). The n queues are initially populated by randomly partitioning an infinite set $\{x_1 < x_2 < x_3 < \ldots\}$ of elements. [2] The queues, identified with the sets of elements they contain, are denoted by $Q_1, \ldots, Q_n$, indexed such that $\min Q_1 < \min Q_2 < \ldots < \min Q_n$. The minima are also called *top-elements*. We then perform a sequence of deletions. Each time, we select an index $i \in [n]$ according to σ and delete the top-element of Q_i. [3] Then, we relabel the queues such that their top-elements appear in ascending order again. Let $(Q_1^{(s)}, \ldots, Q_n^{(s)})$ be the sequence of queues after s deletions and $r_i^{(s)}$ the rank of the top-element $\min Q_i^{(s)}$ among $Q_1^{(s)} \cup \ldots \cup Q_n^{(s)}$, for any $s \in \mathbb{N}$ and $i \in [n]$.

Intuitively, σ should be biased towards smaller values of i, i.e., towards selecting queues with smaller top-elements, to ensure that the rank error does not diverge over time. Using the notation $\sigma_i := \Pr_{i^* \sim \sigma}[i^* = i]$ and $\sigma_{\leq i} := \Pr_{i^* \sim \sigma}[i^* \leq i]$, the formal requirement for σ turns out as:

[2] Note that every queue receives an infinite number of elements with probability 1.
[3] We write $[n]$ as a shorthand for $\{1, \ldots, n\}$.

$$\forall i \in [n-1] : \sigma_{\leq i} > \tfrac{i}{n} \qquad (\star)$$

If σ does not satisfy $(\star)$, the MultiQueue deletes too frequently from "bad" queues (i.e., with large top-elements) and the gap between "good" and "bad" queues increases over time. We prove a corresponding formal claim in the full version of this paper [20].

Surprisingly independent random variables. One might think that the ranks $r_1^{(s)} < \ldots < r_n^{(s)}$ are correlated in complex ways. While they *are* correlated, they effectively arise as the prefix sum of $n-1$ *independent* random variables. More precisely, our main theorem draws attention to the differences between the ranks of consecutive top-elements.

▶ **Theorem 1** (Proof in Section 3). *Let $(r_1^{(s)}, \ldots, r_n^{(s)})$ denote the ranks of the top-elements of a σ-MultiQueue with σ satisfying $(\star)$ after s deletions. Then $(r_1^{(s)}, \ldots, r_n^{(s)})$ converges in distribution to a sequence $(r_1, \ldots, r_n)$ of random variables where*

$$r_i = 1 + \sum_{j=1}^{i-1} \delta_j \text{ for } i \in [n] \text{ with } \delta_i \sim \mathrm{Geom}_1\Big(1 - \frac{i}{n\sigma_{\leq i}}\Big) \text{ for } i \in [n-1],$$

where $\mathrm{Geom}_1(p)$ denotes the geometric distribution of the number of Bernoulli trials with success probability p until (and including) the first success.

From the distribution of the ranks of the top-elements given by Theorem 1 it is straightforward to derive the rank error distribution.

▶ **Corollary 2** (Proof in Section 3). *Let $R^{(s)}$ denote the rank error exhibited by the deletion in step s in the σ-MultiQueue with σ satisfying $(\star)$. Then $R^{(s)}$ converges in distribution to the random variable R where*

$$R = r_{i^*} - 1 \text{ with } i^* \sim \sigma \text{ and } r_i \text{ as in Theorem 1.}$$

The expected rank error is (in the long run)

$$\mathbb{E}[R] = \sum_{i=1}^{n-1} \frac{\sigma_{\leq i}(1-\sigma_{\leq i})}{\sigma_{\leq i} - \frac{i}{n}}.$$

Application to the c-MultiQueue. For $c > 1$ we define the c-MultiQueue to be the σ-MultiQueue where σ is the distribution that corresponds to picking the best out of c randomly selected queues as described in Section 1. For instance, the probability to select one of the first i queues with $c = 2$ is $\sigma_{\leq i} = 1 - (1 - \frac{i}{n})^2$. Note that $(\star)$ is satisfied (see Lemma 8). We can then derive several useful quantities from Theorem 1, including the expected rank error.

▶ **Theorem 3.** *Consider the expectation $\mathbb{E}[R]$ of the rank error $R = r_{i^*} - 1$ of the c-MultiQueue in the long run (i.e., after convergence).*

(i) *For $c = 2$ we have $\mathbb{E}[R] = \frac{5}{6}n - 1 + \frac{1}{6n} = \frac{5}{6}n - \Theta(1)$.*

(ii) *For $c = 1 + \varepsilon$ with $\varepsilon \in (0,1)$, we have $\mathbb{E}[R] = (\frac{1}{\varepsilon} - \frac{\varepsilon}{6})n - \frac{1}{\varepsilon} + \frac{\varepsilon}{6n} = (\frac{1}{\varepsilon} - \frac{\varepsilon}{6})n - \mathcal{O}(1/\varepsilon)$.*

(iii) *For any $c > 1$ we have $n \cdot \int_0^1 f_c(x)\,dx - \frac{c}{c-1} \leq \mathbb{E}[R] \leq n \cdot \int_0^1 f_c(x)\,dx$, where $f_c : (0,1) \to \mathbb{R}$ is defined as follows using $\varepsilon := c - \lfloor c \rfloor \in [0,1)$*

$$f_c(x) = x^{\lfloor c \rfloor - 1}(1 - \varepsilon + \varepsilon x) \cdot \frac{1 - x^{\lfloor c \rfloor}(1 - \varepsilon + \varepsilon x)}{1 - x^{\lfloor c \rfloor - 1}(1 - \varepsilon + \varepsilon x)}.$$

(iv) *Cruder but simpler bounds for any $c \geq 2$ are: $\frac{n}{\lceil c \rceil} - \frac{c}{c-1} \leq \mathbb{E}[R] \leq \frac{n}{\lfloor c \rfloor - 1}$.*

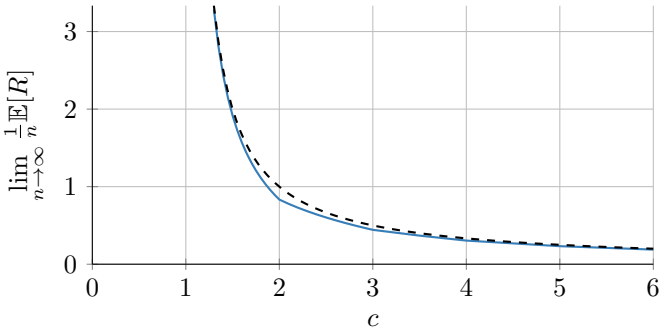

Figure 2 For large n, the c-MultiQueue has an expected rank error of $\mathbb{E}[R] = n \cdot \int_0^1 f_c(x)\,dx + o(n)$ where $f_c(x)$ is defined in Theorem 3. The solid line shows $c \mapsto \int_0^1 f_c(x)\,dx$ and hence the constant in front of the leading term. The dashed line $c \mapsto \frac{1}{c-1}$ is an asymptote for both $c \to 1$ and $c \to \infty$.

Figure 2 plots the expected asymptotic rank error per queue depending on c, and an approximation $\frac{1}{c-1}$. We also give concentration bounds for the rank error in Theorem 9.

The Exponential-Jump Process. As an intermediate step in their analysis, Alistarh et al. [2] introduce the "exponential process", where new top-elements are not given by the current state but generated by adding an exponential random variable to the current top-element. We reformulate this process as the equivalent *exponential-jump process* (EJP) as follows. The EJP involves n tokens on the real number line and a distribution σ on $[n]$. In every step, we sample $i^* \sim \sigma$ and $X \sim \mathrm{Exp}(1)$. We then identify the i^*th token from the left and move it a distance of X to the right. More formally, the state of the process is given by the sequence $t_1 < \ldots < t_n$ of positions of the tokens and the state transition can be described as

$$(t_1, \ldots, t_n) \rightsquigarrow \mathrm{sort}(t_1, \ldots, t_{i^*} + X, \ldots, t_n) \text{ where } i^* \sim \sigma \text{ and } X \sim \mathrm{Exp}(1). \tag{EJP}$$

We provide an exact analysis, which we believe to be of independent interest, and explain the connection between the EJP and the MultiQueue in the full version of this paper [20]. With the same methods as before, we analyse the differences $(d_1, \ldots, d_{n-1})$ with $d_i = t_{i+1} - t_i$.

▶ **Theorem 4.** *Let $n \in \mathbb{N}$ and let σ be a distribution on $[n]$ that satisfies $(\star)$. The EJP admits a stationary distribution π for the differences $(d_1, \ldots, d_{n-1})$ with*

$$\pi = \bigtimes_{i=1}^{n-1} \mathrm{Exp}(n \cdot \sigma_{\le i} - i).$$

In other words, the distances between neighbouring tokens are, in the long-run, mutually independent and exponentially distributed with parameters as given.

3 A Direct Analysis of the σ-MultiQueue

When analysing random processes, it is often a good idea to reveal information only when needed, keeping the rest hidden behind a veil of probability. In our case, the idea is to conceal the queue an element is in until the element becomes a top-element.

We discuss this idea using the example in Figure 3 where $n = 4$ queues are initially populated with the set $\mathbb{N}$.

In (a) we see an explicit representation of a possible state, where queues are labeled in increasing order of their top-elements. We can tell, for instance, that when removing the top-element 7 of queue 2 then the new top-element would be 9. In (b) we keep track of the

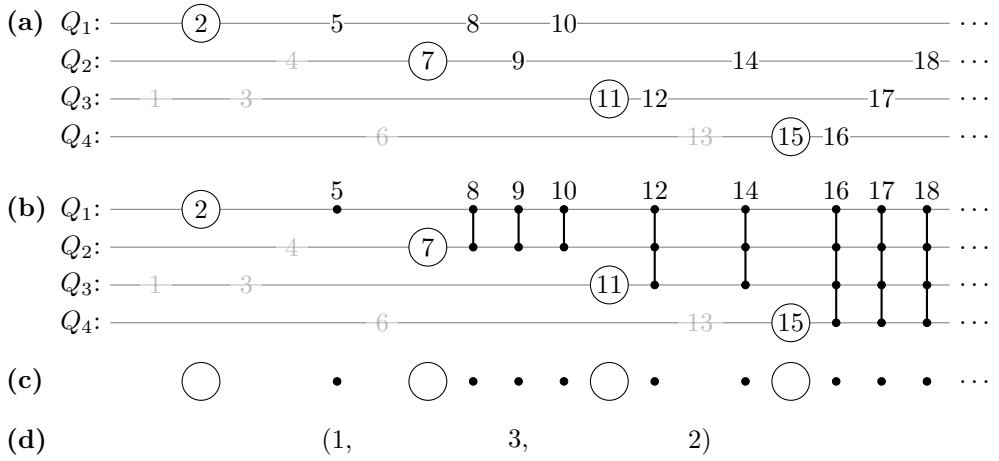

Figure 3 Increasingly abstract ways for modeling the state of a MultiQueue system with 4 queues. **(a)** contains all information. **(b)** assumes that the queues in which the elements reside has only been partially revealed. **(c)** abstracts away from concrete elements. **(d)** represents the information in (c) using numbers.

current and past top-elements of all queues but do not reveal ahead of time which queue each element of $\mathbb{N}$ is assigned to. As far as we know, the elements $16, 17, 18, \ldots$ are assigned to each of the four queues with equal probability. It is unavoidable that we obtain partial information, however: If an element is smaller than the top-element of some queue, it cannot possibly be contained in that queue. The elements 8, 9 and 10 are in queue 1 and 2 with probability 1/2 each, and the elements 12 and 14 are in queues 1, 2 and 3 with probability 1/3 each. Element 5 is surely contained in queue 1, but we can treat this as a degenerate probability distribution rather than as a special case. Note what happens when element 7 is deleted: First, 8 has a chance of 1/2 of being the new top-element of queue 2. If it turns out that 8 is not the new top-element, then 9 gets the same chance, then 10. If all three elements are rejected, then element 12 is considered, getting a chance of 1/3 (because it could still be in three queues) and so on.

Since we are only interested in the ranks of top-elements over time, we can forget the removed elements and the concrete element values and arrive at representation (c), showing n *balls* "◯" representing top-elements and *dots* "•" representing other elements. Equivalently, in (d) we list the sequence of dot-counts in between the balls, omitting the infinite number of dots to the right of the last ball.

We now represent the σ-MultiQueue as a Markov chain with states in $\mathbb{N}_0^{n-1}$ as in (d), borrowing language from (b) and (c) when useful. Since the state space is countably infinite, the role of the transition matrix is filled by an infinite family P of transition probabilities where $P(\vec{d}_{\text{start}}, \vec{d}_{\text{end}})$ denotes the probability to transition from state $\vec{d}_{\text{start}} \in \mathbb{N}_0^{n-1}$ to state $\vec{d}_{\text{end}} \in \mathbb{N}_0^{n-1}$. These probabilities are implicitly described below. We write $\vec{d}$ for a state $(d_1, \ldots, d_{n-1}) \in \mathbb{N}_0^{n-1}$ and $\vec{e}_i = 0^{i-1}10^{n-i-1} \in \{0,1\}^{n-1}$ denotes the ith unit vector for $i \in [n-1]$. We avoid special cases related to the last ball by defining $d_n = \infty$ and $\vec{e}_n = \vec{0}$.

A state $\vec{d}_{\text{start}}$ transitions to another state $\vec{d}_{\text{end}}$ via a sequence of *transitional states* $(\vec{d}, i) \in \mathbb{N}_0^{n-1} \times [n]$ in which one ball $i \in [n]$ (numbered from left to right) is marked as the *active ball*.

1. Given state $\vec{d}_{\text{start}}$, we sample $i^* \sim \sigma$ and obtain the transitional state $(\vec{d}_{\text{start}}, i^*)$.
   Interpretation: In terms of (c) we activate ball i^* and in terms of (b) we delete the top-element from queue Q_{i^*} and look for a new one.

2. As long as we are in a transitional state $(\vec{d}, i)$, there are two cases:

2.1 If $d_i > 0$, then with probability $(i-1)/i$ we continue with transitional state $(\vec{d} - \vec{e}_i + \vec{e}_{i-1}, i)$ and with probability $1/i$ the transition ends with state $\vec{d}_{\text{end}} = \vec{d} - \vec{e}_i$.
Interpretation: In terms of (c) the active ball decides to skip past the dot to the right of it, or consumes the dot and stops. In terms of (b), we reveal whether the next top-element candidate for Q_i is contained in Q_i; if it is, the candidate becomes the new top-element and we stop, otherwise we continue the search.

2.2 If $d_i = 0$, then we continue with transitional state $(\vec{d}, i+1)$.
Interpretation: In terms of (c) the active ball overtakes another ball, thereby becoming ball $i+1$. In terms of (b), we update the ordering of the queues since the new top-element of Q_i is now known to be larger than the top-element of queue Q_{i+1}.

For clarity, we illustrate the possible transitions for a concrete state $\vec{d}_{\text{start}} \in \mathbb{N}_0^{n-1}$ and resulting transition probabilities in Figure 4. We remark that the following analysis does not refer to the transition probabilities directly, but uses their implicit characterisation.

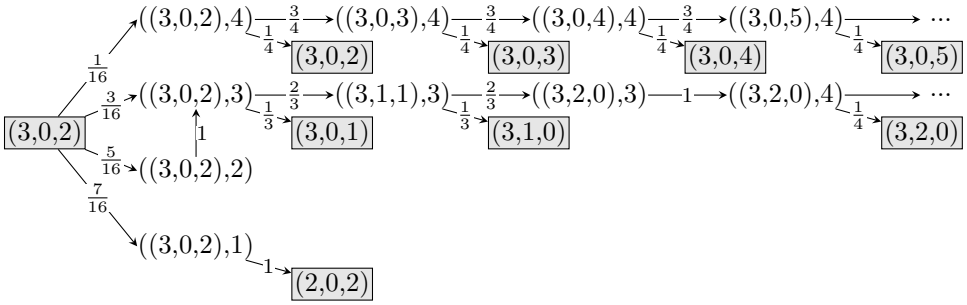

Figure 4 Possible states and transitional states reachable from $\vec{d}_{\text{start}} = (3,0,2)$ in a single state transition. The probabilities $\frac{1}{16}, \frac{3}{16}, \frac{5}{16}, \frac{7}{16}$ on the outgoing edges of $(3,0,2)$ assume the special case of the 2-MultiQueue. Taking the example $\vec{d}_{\text{end}} = (3,1,0)$, the probability to transition from $\vec{d}_{\text{start}}$ to $\vec{d}_{\text{end}}$ arises from the two corresponding paths as $P(\vec{d}_{\text{start}}, \vec{d}_{\text{end}}) = \frac{3}{16} \cdot \frac{2}{3} \cdot \frac{1}{3} + \frac{5}{16} \cdot 1 \cdot \frac{2}{3} \cdot \frac{1}{3}$.

We now state the main result of this section, which characterizes a stationary distribution of P. With this lemma, we can finally prove Theorem 1 and Corollary 2.

▶ **Theorem 5.** *Let $n \in \mathbb{N}$ and let σ be a distribution on $[n]$ that satisfies $(\star)$. The transition probabilities P admits the stationary distribution π given by*

$$\pi = \bigtimes_{i=1}^{n-1} \text{Geom}\Big(1 - \frac{i}{n \cdot \sigma_{\le i}}\Big),$$

where $\text{Geom}(p)$ denotes the geometric distribution of the number of failed Bernoulli trials with success probability p before the first success, and $\bigtimes$ denotes the direct product of distributions.

In particular, the $n-1$ components of $\vec{d} \sim \pi$ are *independent* random variables. We further make the following useful observation.

▶ **Observation 6.** *For any $\vec{d} \in \mathbb{N}_0^{n-1}$ and any $i \in [n]$ we have $\pi(\vec{d} + \vec{e}_i) = \pi(\vec{d}) \cdot \frac{i}{n \cdot \sigma_{\le i}}$.*

Proof of Observation 6. If $i = n$ then $\vec{e}_i = \vec{0}$ and $\sigma_{\le i} = 1$ so the claim is trivial. Now assume $i < n$. For any $\vec{d} \in \mathbb{N}_0^{n-1}$ we have

$$\pi(\vec{d}) = \prod_{i=1}^{n-1} (1-p_i)^{d_i} p_i \text{ where } p_i = 1 - \frac{i}{n \cdot \sigma_{\le i}}.$$

Hence, $\pi(\vec{d}+\vec{e}_i) = \pi(\vec{d}) \cdot (1-p_i)$ and the claim follows. ◀

The following auxiliary lemma captures the main insight required in the proof of Theorem 5.

▶ **Lemma 7.** *Let $\nu(\vec{d}, i) \in \mathbb{R}_{\geq 0}$ be the probability that a transitional state $(\vec{d}, i) \in \mathbb{N}_0^{n-1} \times [n]$ occurs when transitioning from state $\vec{d}_{\text{start}} \sim \pi$ according to P with σ satisfying $(\star)$. Then,*

$$\nu(\vec{d}, i) = \pi(\vec{d}) \cdot \sigma_{\leq i}.$$

Proof of Lemma 7. We prove $\nu(\vec{d}, i) = \pi(\vec{d}) \cdot \sigma_{\leq i}$ by induction. In the induction step we may assume that $\nu(\vec{d}', i') = \pi(\vec{d}') \cdot \sigma_{\leq i'}$ holds whenever $i' < i$ or when $i = i'$ and $d'_1 + \ldots + d'_{i-1} < d_1 + \ldots + d_{i-1}$. In general, there are three ways in which a transitional state $(\vec{d}, i)$ might be reached:

- **(i)** The transition started with $\vec{d} = \vec{d}_{\text{start}}$ and ball i was activated.
- **(ii)** Ball i has skipped a dot and we thus reached $(\vec{d}, i)$ from $(\vec{d} - \vec{e}_{i-1} + \vec{e}_i, i)$.
- **(iii)** Ball i has just overtaken another ball and we thus reached $(\vec{d}, i)$ from $(\vec{d}, i-1)$.

For $i = 1$, consider a transitional state $(\vec{d}, 1)$ where the first ball is active. Here, only (i) is possible since the first ball never skips a dot (transition ends with probability 1 in rule 2.1) and there is no ball to its left. The probability for $(\vec{d}, 1)$ to occur is thus

$$\nu(\vec{d}, 1) = \pi(\vec{d}) \cdot \sigma_1 = \pi(\vec{d}) \cdot \sigma_{\leq 1} \qquad \text{(recall that } \sigma_i := \Pr_{i^* \sim \sigma}[i^* = i] \text{ and } \sigma_{\leq i} := \Pr_{i^* \sim \sigma}[i^* \leq i]\text{)}.$$

For $i > 1$, there are two cases depending on whether $d_{i-1} > 0$, i.e., whether there is a dot in between ball $i-1$ and ball i. If $d_{i-1} > 0$, then only (i) and (ii) are possible, so

$$\begin{aligned}
\nu(\vec{d}, i) &= \pi(\vec{d}) \cdot \sigma_i + \nu(\vec{d} - \vec{e}_{i-1} + \vec{e}_i, i) \cdot (1 - \tfrac{1}{i}) \\
&= \pi(\vec{d}) \cdot \sigma_i + \pi(\vec{d} - \vec{e}_{i-1} + \vec{e}_i) \cdot \sigma_{\leq i} \cdot (1 - \tfrac{1}{i}) && \text{(Induction)} \\
&= \pi(\vec{d}) \cdot \sigma_i + \pi(\vec{d}) \cdot \frac{n \cdot \sigma_{\leq i-1}}{i-1} \cdot \frac{i}{n \cdot \sigma_{\leq i}} \cdot \sigma_{\leq i} \cdot (1 - \tfrac{1}{i}) && \text{(Observation 6)} \\
&= \pi(\vec{d})(\sigma_i + \sigma_{\leq i-1}) = \pi(\vec{d}) \cdot \sigma_{\leq i}.
\end{aligned}$$

If $d_{i-1} = 0$, then only (i) and (iii) are possible, so

$$\nu(\vec{d}, i) = \pi(\vec{d}) \cdot \sigma_i + \nu(\vec{d}, i-1) \overset{\text{ind.}}{=} \pi(\vec{d}) \cdot \sigma_i + \pi(\vec{d}) \cdot \sigma_{\leq i-1} = \pi(\vec{d}) \cdot \sigma_{\leq i}.$$ ◀

With Lemma 7 in place, we can now prove Theorem 5.

Proof of Theorem 5. Given a state $\vec{d}_{\text{start}} \sim \pi$, we transition to a new state $\vec{d}_{\text{end}}$ according to the transition probabilities P. To end in $\vec{d}_{\text{end}}$, we first need to reach the transitional state $(\vec{d}_{\text{end}} + \vec{e}_i, i)$ for some $i \in [n]$ and then decide to end the transition there. Note that we need $(\vec{d}_{\text{end}} + \vec{e}_i, i)$ rather than $(\vec{d}_{\text{end}}, i)$, since ending the transition (rule 2.1) reduces d_i by one. The probability to end in $\vec{d}_{\text{end}}$ is therefore

$$\begin{aligned}
\Pr[\vec{d}_{\text{end}} = \vec{d}] &= \sum_{i=1}^{n} \nu(\vec{d} + \vec{e}_i, i) \cdot \tfrac{1}{i} \overset{\text{Lem. 7}}{=} \sum_{i=1}^{n} \pi(\vec{d} + \vec{e}_i) \cdot \sigma_{\leq i} \cdot \tfrac{1}{i} \\
&\overset{\text{Obs. 6}}{=} \sum_{i=1}^{n} \pi(\vec{d}) \cdot \frac{i}{n \cdot \sigma_{\leq i}} \cdot \sigma_{\leq i} \cdot \tfrac{1}{i} = \pi(\vec{d}) \sum_{i=1}^{n} \tfrac{1}{n} = \pi(\vec{d}).
\end{aligned}$$

It follows that $\vec{d}_{\text{end}}$ is again distributed according to π and π is a stationary distribution. ◀

Finally, we prove Theorem 1 and Corollary 2.

Proof of Theorem 1. Let $\vec{d}^{(s)} = (d_1^{(s)}, \dots, d_{n-1}^{(s)})$ with $d_i^{(s)} = r_{i+1}^{(s)} - r_i^{(s)} - 1$. Then, $(\vec{d}^{(s)})_{s\in\mathbb{N}}$ is a Markov chain with transition probabilities P. Since we can reach $(0, \dots, 0)$ from any state and vice versa, the Markov chain is irreducible. The Markov chain is aperiodic since $(0, \dots, 0)$ can transition into itself (if ball n is activated and immediately stops). This implies that the stationary distribution π that we found is unique and that $\vec{d}^{(s)}$ converges in distribution to $d \sim \pi$ (see [13, Theorem 1.8.3]). Let $\vec{\delta}^{(s)} = (\delta_1^{(s)}, \dots, \delta_{n-1}^{(s)})$ with $\delta_i^{(s)} = d_i^{(s)} + 1$. Clearly, $(\vec{\delta}^{(s)})_{s\in\mathbb{N}}$ converges in the same way, except that the geometric random variables are shifted. By definition, we have $r_i^{(s)} = 1 + \sum_{j=1}^{i-1} \delta_j^{(s)}$ so the claimed distributional limit $(r_1, \dots, r_n)$ of $(r_1^{(s)}, \dots, r_n^{(s)})$ follows. ◀

Proof of Corollary 2. Theorem 1 states the ranks of the top-elements converge in distribution to $(r_1, \dots, r_n)$. The distribution for R follows from the fact that we select the queue to delete from according to σ and deleting an element with rank r yields a rank error of $r - 1$. Using the fact that $\mathbb{E}_{X\sim\mathrm{Geom}_1(p)}[X] = 1/p$, we have for the expected rank error

$$\begin{aligned}\mathbb{E}[R] = \mathbb{E}\left[r_{i^*} - 1\right] &= \sum_{i=1}^{n} \sigma_i \sum_{j=1}^{i-1} \mathbb{E}[\delta_j] = \sum_{j=1}^{n-1} \mathbb{E}[\delta_j] \sum_{i=j+1}^{n} \sigma_i = \sum_{j=1}^{n-1} \mathbb{E}[\delta_j] \cdot (1 - \sigma_{\le j}) \\ &= \sum_{j=1}^{n-1} \frac{n\sigma_{\le j}}{n\sigma_{\le j} - j} \cdot (1 - \sigma_{\le j}) = \sum_{j=1}^{n-1} \frac{\sigma_{\le j}(1 - \sigma_{\le j})}{\sigma_{\le j} - \frac{j}{n}}.\end{aligned}$$ ◀

4 Application to the c-MultiQueue

In this section we apply our results on the general σ-MultiQueue to the c-MultiQueue with $c > 1$. After checking in Lemma 8 that the corresponding σ satisfies $(\star)$, we proceed to compute expected rank errors (Theorem 3) and derive a concentration bound (Theorem 9). This involves straightforward (though mildly tedious) calculations.

▶ **Lemma 8.** *In the c-MultiQueue with $c = \lfloor c \rfloor + \varepsilon > 1$ we have*

$$\sigma_{\le i} = 1 - (1 - \tfrac{i}{n})^{\lfloor c \rfloor}(1 - \varepsilon\tfrac{i}{n}) \text{ for all } i \in [n], \text{ and } \sigma \text{ satisfies } (\star).$$

Proof. Recall that we sample $\lfloor c \rfloor$ queues with probability $1 - \varepsilon$ and $\lfloor c \rfloor + 1$ queues otherwise. We fail to select one of the first i queues only if none of them were sampled. Hence for $i < n$:

$$\begin{aligned}\sigma_{\le i} &= 1 - (1 - \varepsilon)(1 - \tfrac{i}{n})^{\lfloor c \rfloor} - \varepsilon(1 - \tfrac{i}{n})^{\lfloor c \rfloor + 1} = 1 - (1 - \tfrac{i}{n})^{\lfloor c \rfloor}(1 - \varepsilon + \varepsilon(1 - \tfrac{i}{n})) \\ &= 1 - (1 - \tfrac{i}{n})^{\lfloor c \rfloor}(1 - \varepsilon\tfrac{i}{n}) > 1 - (1 - \tfrac{i}{n}) = \tfrac{i}{n}.\end{aligned}$$

The inequality uses that we have $\lfloor c \rfloor \ge 2$, or $\varepsilon > 0$, or both. ◀

We now prove Theorem 3, restated here for easier reference.

▶ **Theorem 3.** *Consider the expectation $\mathbb{E}[R]$ of the rank error $R = r_{i^*} - 1$ of the c-MultiQueue in the long run (i.e., after convergence).*

(i) *For $c = 2$ we have $\mathbb{E}[R] = \frac{5}{6}n - 1 + \frac{1}{6n} = \frac{5}{6}n - \Theta(1)$.*

(ii) *For $c = 1 + \varepsilon$ with $\varepsilon \in (0, 1)$, we have $\mathbb{E}[R] = (\frac{1}{\varepsilon} - \frac{\varepsilon}{6})n - \frac{1}{\varepsilon} + \frac{\varepsilon}{6n} = (\frac{1}{\varepsilon} - \frac{\varepsilon}{6})n - \mathcal{O}(1/\varepsilon)$.*

(iii) *For any $c > 1$ we have* $n \cdot \int_0^1 f_c(x)\,dx - \frac{c}{c-1} \le \mathbb{E}[R] \le n \cdot \int_0^1 f_c(x)\,dx$, *where* $f_c : (0,1) \to \mathbb{R}$ *is defined as follows using* $\varepsilon := c - \lfloor c \rfloor \in [0,1)$

$$f_c(x) = x^{\lfloor c \rfloor - 1}(1 - \varepsilon + \varepsilon x) \cdot \frac{1 - x^{\lfloor c \rfloor}(1 - \varepsilon + \varepsilon x)}{1 - x^{\lfloor c \rfloor - 1}(1 - \varepsilon + \varepsilon x)}.$$

(iv) *Cruder but simpler bounds for any* $c \ge 2$ *are:* $\frac{n}{\lceil c \rceil} - \frac{c}{c-1} \le \mathbb{E}[R] \le \frac{n}{\lfloor c \rfloor - 1}$.

Proof of Theorem 3. We have just checked in Lemma 8 that σ satisfies $(\star)$ and computed $\sigma_{\le i} = 1 - (1 - \frac{i}{n})^{\lfloor c \rfloor}(1 - \varepsilon\frac{i}{n})$. We can therefore specialise the formula for the expected rank error from Corollary 2 by plugging in σ and simplifying, which yields

$$\begin{aligned}
\mathbb{E}[R] &= \sum_{i=1}^{n-1}(1 - \sigma_{\le i})\frac{\sigma_{\le i}}{\sigma_{\le i} - \frac{i}{n}} = \sum_{i=1}^{n-1}(1 - \tfrac{i}{n})^{\lfloor c \rfloor}(1 - \varepsilon\tfrac{i}{n})\frac{1 - (1 - \frac{i}{n})^{\lfloor c \rfloor}(1 - \varepsilon\frac{i}{n})}{1 - (1 - \frac{i}{n})^{\lfloor c \rfloor}(1 - \varepsilon\frac{i}{n}) - \frac{i}{n}} \\
&= \sum_{i=1}^{n-1}(1 - \tfrac{i}{n})^{\lfloor c \rfloor - 1}(1 - \varepsilon\tfrac{i}{n})\frac{1 - (1 - \frac{i}{n})^{\lfloor c \rfloor}(1 - \varepsilon\frac{i}{n})}{1 - (1 - \frac{i}{n})^{\lfloor c \rfloor - 1}(1 - \varepsilon\frac{i}{n})} && \text{(cancel } 1 - \tfrac{i}{n}\text{)} \\
&= \sum_{i=1}^{n-1}(\tfrac{i}{n})^{\lfloor c \rfloor - 1}(1 - \varepsilon + \varepsilon\tfrac{i}{n})\frac{1 - (\frac{i}{n})^{\lfloor c \rfloor}(1 - \varepsilon + \varepsilon\frac{i}{n})}{1 - (\frac{i}{n})^{\lfloor c \rfloor - 1}(1 - \varepsilon + \varepsilon\frac{i}{n})} && \text{(substitute } i \to n - i.\text{)} \\
&= \sum_{i=1}^{n-1} f_c(\tfrac{i}{n}). && \text{(using the definition of } f_c(x) \text{ given above.)}
\end{aligned}$$

We are now ready to prove claim (i). Using that $f_2(x) = x \cdot \frac{1-x^2}{1-x} = x \cdot (1 + x) = x + x^2$, we have

$$\begin{aligned}
\mathbb{E}[R] &= \sum_{i=1}^{n-1} f_2(\tfrac{i}{n}) = \sum_{i=1}^{n-1}\left(\tfrac{i}{n} + (\tfrac{i}{n})^2\right) = \frac{1}{n}\sum_{i=1}^{n-1} i + \frac{1}{n^2}\sum_{i=1}^{n-1} i^2 \\
&= \frac{n-1}{2} + \frac{(n-1)(2n-1)}{6n} = \tfrac{5}{6}n - 1 + \tfrac{1}{6n}.
\end{aligned}$$

Similarly, we can prove (ii). First, we simplify $f_c(x)$ for $c = 1 + \varepsilon$ with $\varepsilon \in (0,1)$:

$$\begin{aligned}
f_{1+\varepsilon}(x) &= (1 - \varepsilon + \varepsilon x) \cdot \frac{1 - x(1 - \varepsilon + \varepsilon x)}{\varepsilon - \varepsilon x} = (1 - \varepsilon + \varepsilon x) \cdot \frac{(1-x)(1 + \varepsilon x)}{\varepsilon(1-x)} \\
&= (1 - \varepsilon + \varepsilon x) \cdot \frac{1 + \varepsilon x}{\varepsilon} = \tfrac{1-\varepsilon}{\varepsilon} + (2 - \varepsilon)x + \varepsilon x^2.
\end{aligned}$$

We then get (omitting a simple calculation):

$$\begin{aligned}
\mathbb{E}[R] &= \sum_{i=1}^{n-1} f_{1+\varepsilon}(\tfrac{i}{n}) = \sum_{i=1}^{n-1}(\tfrac{1-\varepsilon}{\varepsilon} + (2 - \varepsilon)\tfrac{i}{n} + \varepsilon(\tfrac{i}{n})^2) \\
&= \tfrac{1-\varepsilon}{\varepsilon}(n-1) + (2 - \varepsilon)\tfrac{n(n-1)}{2n} + \varepsilon\tfrac{(n-1)n(2n-1)}{6n^2} = \ldots = (\tfrac{1}{\varepsilon} - \tfrac{\varepsilon}{6})n - \tfrac{1}{\varepsilon} + \tfrac{\varepsilon}{6n}.
\end{aligned}$$

We now turn our attention to general $c > 1$ again. Our goal is to approximate the sum by an integral. To bound the approximation error effectively, we will first show that f_c is monotonic. For this let us examine the fractional term $g_c(x)$ occurring in $f_c(x)$:

$$g_c(x) := \frac{1-x^{\lfloor c\rfloor}(1-\varepsilon+\varepsilon x)}{1-x^{\lfloor c\rfloor-1}(1-\varepsilon+\varepsilon x)} = 1+\frac{(x^{\lfloor c\rfloor-1}-x^{\lfloor c\rfloor})(1-\varepsilon+\varepsilon x)}{1-x^{\lfloor c\rfloor-1}(1-\varepsilon+\varepsilon x)}$$
$$= 1+\frac{x^{\lfloor c\rfloor-1}(1-x)(1-\varepsilon+\varepsilon x)}{1-x^{\lfloor c\rfloor-1}+\varepsilon x^{\lfloor c\rfloor-1}(1-x)} = 1+\frac{x^{\lfloor c\rfloor-1}(1-x)(1-\varepsilon+\varepsilon x)}{(1-x)\sum_{i=0}^{\lfloor c\rfloor-2}x^i+\varepsilon x^{\lfloor c\rfloor-1}(1-x)}$$
$$= 1+\frac{1-\varepsilon+\varepsilon x}{\sum_{i=1}^{\lfloor c\rfloor-1}x^{-i}+\varepsilon}. \tag{1}$$

From (1) we can see that $g_c(x)$ does not have a singularity at $x=1$ and by setting $g_c(1)=f_c(1)=1+\frac{1}{c-1}=\frac{c}{c-1}$ both functions are continuous on $[0,1]$. We also see from (1) that $g_c(x)$ is increasing in x and decreasing in c (recall that c determines $\varepsilon=c-\lfloor c\rfloor$). Because the other factor $x^{\lfloor c\rfloor-1}(1-\varepsilon+\varepsilon x)$ of $f_c(x)$ is also increasing in x and decreasing in c, we conclude that $f_c(x)$ as a whole is increasing in x and decreasing in c. This makes the upper and lower sums bounding the integral of f_c particularly simple:

$$\sum_{i=1}^{n-1}f_c(\tfrac{i}{n})=\sum_{i=0}^{n-1}f_c(\tfrac{i}{n})\le n\int_0^1 f_c(x)\,dx\le\sum_{i=1}^{n}f_c(\tfrac{i}{n})=f_c(1)+\sum_{i=1}^{n-1}f_c(\tfrac{i}{n})=\tfrac{c}{c-1}+\sum_{i=1}^{n-1}f_c(\tfrac{i}{n}).$$

By rearranging we obtain (iii). We now turn to the cruder bound (iv). We begin with $c\in\mathbb{N}\setminus\{1\}$. In this case we have $f_c(x)=x^{c-1}\cdot g_c(x)$ where $1\le g_c(x)\le\frac{c}{c-1}$ for all $x\in[0,1]$. This gives:

$$\tfrac{1}{c}=\int_0^1 x^{c-1}\,dx\le\int_0^1 f_c(x)\,dx\le\int_0^1 x^{c-1}\tfrac{c}{c-1}\,dx=\tfrac{c}{c-1}\int_0^1 x^{c-1}\,dx=\tfrac{1}{c-1}.$$

Combining this with (iii) gives $\frac{n}{c}-\frac{c}{c-1}\le\mathbb{E}[R]\le\frac{n}{c-1}$ when $c\in\mathbb{N}\setminus\{1\}$. When $c\notin\mathbb{N}$ we can use that $f_c(x)$ is decreasing in c and round conservatively to obtain the claimed result. ◀

▶ **Theorem 9.** *In the 2-MultiQueue[4], the highest rank error observed over a polynomial number of deletions is in $\mathcal{O}(n\log n)$ with high probability.*

Proof. We will use a tail bound by Janson [9, Theorem 2.1] on the sum of geometrically distributed random variables from. It reads

$$\Pr[X\ge k\cdot\mu]\le e^{-p_*\mu(k-1-\log k)}\text{ for any }k\ge 1,$$

where X is a sum of geometrically distributed random variables (with possibly differing success probabilities), p_* is the smallest success probability and $\mu=\mathbb{E}[X]$.

We apply this bound to $X=r_n-1$, which has the required form by Theorem 1. It is easy to check that $p_*=\delta_{n-1}=\frac{1}{n+1}=\Theta(\frac{1}{n})$ and

$$\mu=\mathbb{E}[X]=\sum_{i=1}^{n-1}\frac{1-(1-\frac{i}{n})^2}{1-(1-\frac{i}{n})^2-\frac{i}{n}}=\sum_{i=1}^{n-1}\frac{2-\frac{i}{n}}{1-\frac{i}{n}}=n-1+n\sum_{i=1}^{n-1}\tfrac{1}{i}=\Theta(n\log n).$$

Since the rank error R clearly satisfies $R\le r_n-1=X$ we have for k large enough

$$\Pr[R>k\cdot n\log n]\le\Pr[X>k\cdot n\log n]=e^{-\Theta(\frac{1}{n}n\log(n)k)}=n^{-\Theta(k)}.$$

By a union bound, the probability that we observe a rank error exceeding $k\cdot n\log n$ at some point during n^c deletions is at most $n^c\cdot n^{-\Theta(k)}$. For large enough k, this probability is still small. ◀

[4] A similar analysis for $c=1+\varepsilon$ is also possible.

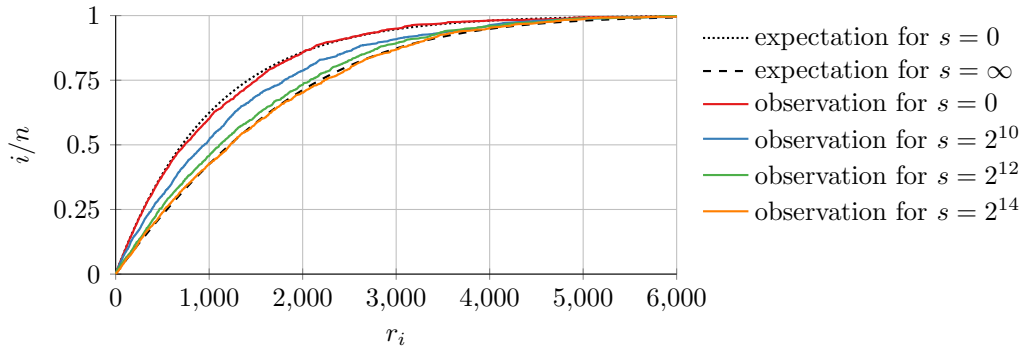

Figure 5 The convergence of the 2-MultiQueue with $n = 2^{10}$ to its stable state. After s deletions, let $r_i^{(s)}$ be the rank of queue i. The plot shows the observed ranks $i \mapsto r_i^{(s)}$ for some values of s, as well as the expected ranks $i \mapsto \mathbb{E}[r_i^{(0)}]$ and $i \mapsto \mathbb{E}[r_i^{(\infty)}]$ in the initial state and the converged state.

To give some intuition for how quickly the 2-MultiQueue converges to its stable state and for how close the observed ranks of top-elements are to their expectation we provide experimental data in Figure 5.

5 Future Work

While we have fully analyzed the long-term behavior of the MultiQueue in the deletion-only setting, there are still open questions in the general setting. Specifically, our methods are not directly applicable when insertions of arbitrary elements can happen after deletions. We conjecture that the deletion-only setting represents the worst-case in the sense that the expected rank error cannot be made worse even by adversarial insertions. On the flip side, we conjecture that the expected rank error is never better than before the first deletion. Our reasoning is as follows: Inserting sufficiently large elements does not affect deletions and is equivalent to inserting before deleting. When inserting many small elements, the state drifts towards the state where no deletions happened. In summary, we expect the state of the MultiQueue always to be "between" the insertion-only and the deletion-only setting, and that our analysis yields accurate predictions after sufficiently many deletions regardless of previous operations.

Our analysis applies directly to applications that do insert new elements after the first deletion, such as heap sort and simple batch job scheduling. A much broader range of applications, including Dijkstra's algorithm and discrete-event simulations, process elements in *monotonous* fashion, meaning that the elements deleted from the priority queue are monotonically increasing. When using a MultiQueue, newly inserted elements are unlikely to become top-elements, and we expect the state of the MultiQueue to stay "close" to the deletion-only setting.

Williams et al. [21] proposed the *delay* as an additional quality metric and *stickiness* as a way to increase throughput. The delay of an element e measures how many elements worse than e are deleted while e resides in the queue. Stickiness lets threads reuse the same queue for multiple consecutive operations. We believe that the delay can be analyzed directly with our approach and that the Markov chain can be adapted to handle stickiness as well.

In practice, it is relevant how fast the system stabilizes and converges to the postulated distributions of ranks or what rank errors are to be expected until then. Thus, analyzing the convergence speed is a natural next step.

Alistarh et al. [1] analyze the MultiQueue in concurrent settings where comparisons can become *stale*, meaning that after deciding which queue to delete from but before actually deleting from it, its top-element might change. We find it interesting whether our analysis can be adapted to this scenario as well.

References

1 Dan Alistarh, Trevor Brown, Justin Kopinsky, Jerry Z. Li, and Giorgi Nadiradze. Distributionally Linearizable Data Structures. In *SPAA*, 2018. doi:10.1145/3210377.3210411.

2 Dan Alistarh, Justin Kopinsky, Jerry Li, and Giorgi Nadiradze. The Power of Choice in Priority Scheduling. In *PODC*, 2017. doi:10.1145/3087801.3087810.

3 Dan Alistarh, Justin Kopinsky, Jerry Li, and Nir Shavit. The SprayList: A scalable relaxed priority queue. In *PPoPP*, 2015. doi:10.1145/2688500.2688523.

4 Hagit Attiya, Rachid Guerraoui, Danny Hendler, Petr Kuznetsov, Maged M. Michael, and Martin Vechev. Laws of order: Expensive synchronization in concurrent algorithms cannot be eliminated. In *POPL*, 2011. doi:10.1145/1926385.1926442.

5 Yossi Azar, Andrei Z. Broder, Anna R. Karlin, and Eli Upfal. Balanced Allocations. *SICOMP*, 29(1), 1999. doi:10.1137/S0097539795288490.

6 Petra Berenbrink, Artur Czumaj, Angelika Steger, and Berthold Vöcking. Balanced Allocations: The Heavily Loaded Case. *SICOMP*, 35(6), 2006. doi:10.1137/S009753970444435X.

7 Irina Calciu, Hammurabi Mendes, and Maurice Herlihy. The Adaptive Priority Queue with Elimination and Combining. In *DC*, 2014. doi:10.1007/978-3-662-45174-8_28.

8 Faith Ellen, Danny Hendler, and Nir Shavit. On the Inherent Sequentiality of Concurrent Objects. *SICOMP*, 41(3), 2012. doi:10.1137/08072646X.

9 Svante Janson. Tail bounds for sums of geometric and exponential variables. *S&P L*, 135, 2018. doi:10.1016/j.spl.2017.11.017.

10 Richard M. Karp and Yanjun Zhang. Randomized parallel algorithms for backtrack search and branch-and-bound computation. *JACM*, 40(3), 1993. doi:10.1145/174130.174145.

11 Jonatan Lindén and Bengt Jonsson. A Skiplist-Based Concurrent Priority Queue with Minimal Memory Contention. In *OPODIS*, 2013. doi:10.1007/978-3-319-03850-6_15.

12 Michael Mitzenmacher, Andréa W. Richa, and Ramesh Sitaraman. The Power of Two Random Choices: A Survey of Techniques and Results, 2001. URL: http://www.eecs.harvard.edu/~michaelm/postscripts/handbook2001.pdf.

13 J. R. Norris. *Markov Chains*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge university press, 1997. doi:10.1017/CBO9780511810633.

14 Yuval Peres, Kunal Talwar, and Udi Wieder. Graphical balanced allocations and the $(1+\beta)$-choice process. *RS & A*, 47(4), 2015. doi:10.1002/rsa.20558.

15 Anastasiia Postnikova, Nikita Koval, Giorgi Nadiradze, and Dan Alistarh. Multi-queues can be state-of-the-art priority schedulers. In *PPoPP*, 2022. doi:10.1145/3503221.3508432.

16 Hamza Rihani, Peter Sanders, and Roman Dementiev. MultiQueues: Simple Relaxed Concurrent Priority Queues. In *SPAA*, 2015. doi:10.1145/2755573.2755616.

17 Adones Rukundo and Philippas Tsigas. TSLQueue: An Efficient Lock-Free Design for Priority Queues. In *Euro-Par*, 2021. doi:10.1007/978-3-030-85665-6_24.

18 Konstantinos Sagonas and Kjell Winblad. The Contention Avoiding Concurrent Priority Queue. In *LCPC*, 2017. doi:10.1007/978-3-319-52709-3_23.

19 N. Shavit and I. Lotan. Skiplist-based concurrent priority queues. In *IPDPS*, 2000. doi:10.1109/IPDPS.2000.845994.

20 Stefan Walzer and Marvin Williams. A Simple yet Exact Analysis of the MultiQueue, 2025. doi:10.48550/arXiv.2410.08714.

21 Marvin Williams, Peter Sanders, and Roman Dementiev. Engineering MultiQueues: Fast Relaxed Concurrent Priority Queues. In *ESA*, 2021. doi:10.4230/LIPIcs.ESA.2021.81.

22 Martin Wimmer, Jakob Gruber, Jesper Larsson Träff, and Philippas Tsigas. The lock-free k-LSM relaxed priority queue. In *PPoPP*, 2015. doi:10.1145/2688500.2688547.

23 Guozheng Zhang, Gilead Posluns, and Mark C. Jeffrey. Multi Bucket Queues: Efficient Concurrent Priority Scheduling. In *SPAA*, 2024. doi:10.1145/3626183.3659962.