Upper Tail Analysis of Bucket Sort and Random Tries[*]

Ioana O. Bercea[†]   Guy Even[*]

Abstract

Bucket Sort is known to run in expected linear time when the input keys are distributed independently and uniformly at random in the interval $[0, 1)$. The analysis holds even when a quadratic time algorithm is used to sort the keys in each bucket. We show how to obtain linear time guarantees on the running time of Bucket Sort that hold with *very high probability*. Specifically, we investigate the asymptotic behavior of the exponent in the upper tail probability of the running time of Bucket Sort. We consider large additive deviations from the expectation, of the form cn for large enough (constant) c, where n is the number of keys that are sorted.

Our analysis shows a profound difference between variants of Bucket Sort that use a quadratic time algorithm within each bucket and variants that use a $\Theta(b \log b)$ time algorithm for sorting b keys in a bucket. When a quadratic time algorithm is used to sort the keys in a bucket, the probability that Bucket Sort takes cn more time than expected is exponential in $\Theta(\sqrt{n} \log n)$. When a $\Theta(b \log b)$ algorithm is used to sort the keys in a bucket, the exponent becomes $\Theta(n)$. We prove this latter theorem by showing an upper bound on the tail of a random variable defined on tries, a result which we believe is of independent interest. This result also enables us to analyze the upper tail probability of a well-studied trie parameter, the external path length, and show that the probability that it deviates from its expected value by an additive factor of cn is exponential in $\Theta(n)$.

1   Introduction

The Bucket Sort algorithm sorts n keys in the interval $[0, 1)$ as follows: (i) Distribute the keys among n buckets, where the jth bucket consists of all the keys in the interval $[j/n, (j + 1)/n)$. (ii) Sort the keys in each bucket. (iii) Scan the buckets and output the keys in each bucket in their sorted order. We consider two natural classes of Bucket Sort algorithms that differ in how the keys inside each bucket are sorted. The first class of BucketSort algorithms that we consider sorts the keys inside a bucket using a quadratic time algorithm (such as Insertion Sort). We refer to algorithms in this class as b^2-Bucket Sort. The second class of algorithms sorts the keys in a bucket using a $\Theta(b \log b)$ algorithm for sorting b keys (such as Merge Sort). We refer to this variant as $b \log b$-Bucket Sort.

When the n keys are distributed independently and uniformly at random, the expected running time of Bucket Sort is $\Theta(n)$, even when a quadratic time algorithm is used to sort the keys in each bucket [CLRS09, MU17, SMDD19]. A natural question is whether such linear time guarantees hold with high probability. For Quick Sort, analyses of this sort have a long and rich history [Jan15, FJ02, MH96].

In this paper, we focus on analyzing the running time of Bucket Sort with respect to large deviations, e.g., running times that exceed the expectation by $10n$. In particular, we study the asymptotic behavior of the exponent in the upper tail of the running time.

Rate of the upper tail. We analyze the upper tail probability of a random variable using the notion of rate, defined as follows.[1]

Definition 1. *Given a random variable Y with expected value μ, we define the* rate *of the upper tail of Y to be the function defined on $t > 0$ as follows:*

$$R_Y(t) \triangleq -\ln\left(\Pr\left[Y \geq \mu + t\right]\right).$$

Note that we consider an additive deviation from the expectation, i.e., we bound the probability that the random variable deviates from its expected value by an additive term of t, for sufficiently large values of t.[2] In particular, we consider values of $t = cn$, where n is the size of the input and c is a constant greater than some threshold. Finally, we abbreviate and refer to $R_Y(t)$ as the *rate of Y*.

We study the rates of the running times of deterministic Bucket Sort algorithms in which the input is sampled from a uniform probability distribution. We also consider parameters of tries induced by infinite prefix-free binary strings chosen independently and uniformly at random.

1.1   Our Contributions

Our first two results derive the rates of the two classes of Bucket Sort algorithms and show that they are different. Specifically, we prove the following:

Theorem 2. *There exists a constant $C > 0$ such that, for all $c > C$, the rate $R_{b^2}(\cdot)$ of the b^2-Bucket Sort algorithm on n keys chosen independently and uniformly at random in $[0, 1)$ satisfies $R_{b^2}(cn) = \Theta(\sqrt{n} \log n)$.*

Since the expected running time of b^2-Bucket Sort is $\Theta(n)$, Theorem 2 states that the probability that b^2-Bucket Sort on random keys takes more than dn time is $e^{-\Theta(\sqrt{n} \log n)}$ (for a sufficiently large constant d).[3] Theorem 2 proves both a lower bound and an upper bound on the asymptotic rate $R_{b^2}(cn)$. In particular, Theorem 2 rules out the possibility that the probability that the running time of b^2-Bucket Sort is greater than $100n$ is bounded by $e^{-\Theta(n)}$.

We prove the lower bound on $R_{b^2}(cn)$ by applying multiplicative Chernoff bounds in different regimes of large (superconstant, in fact) deviations from the mean. In such settings, the dependency of the exponent of the Chernoff bound on the deviation from the mean can have a significant impact on the quality of the bounds we obtain. Indeed, we employ a rarely used form of the Chernoff bound that exhibits a $\delta \log \delta$ dependency in the exponent when the deviation from the mean is δ (see Eq. 17 in Appendix C and Chapter 10.1.1 in [Doe18]). Although the proof of this bound is straightforward, the proof of Theorem 2 crucially relies on this additional (superconstant) $\log \delta$ factor (see Claim 23).

For $b \log b$-Bucket Sort on random keys, we show that the rate is linear in the size of the input:

Theorem 3. *There exists a constant $C > 0$ such that, for all $c > C$, the rate $R_{b \log b}(\cdot)$ of the $b \log b$-Bucket Sort algorithm on n keys chosen independently and uniformly at random in $[0, 1)$ satisfies $R_{b \log b}(cn) = \Theta(n)$.*

---

[1]Throughout the paper, $\ln x$ denotes the natural logarithm of x and $\log x$ denotes the logarithm of base 2 of x.

[2]One should not confuse this analysis with concentration bounds that address small deviations from the expectation.

[3]The threshold C depends on: (1) the constant that appears in the sorting algorithm used within each bucket, and (2) the constant that appears in the expected running time of b^2-Bucket Sort.

We prove the lower bound on $R_{b \log b}(cn)$ by analyzing a random variable arising in random tries. Specifically, we consider tries on infinite binary strings in which each bit is chosen independently and uniformly at random. The parameter we study is called the *excess path length* and is defined formally in Section 2. We show that the time it takes to sort the buckets in $b \log b$-Bucket Sort can be upper bounded by the excess path length in a random trie (Lemma 12). We then bound the upper tail of the excess path length (Theorem 15) and use it to lower bound $R_{b \log b}(cn)$.

We also use the upper tail of the excess path length to derive the rate of a well-studied trie parameter, the sum of root to leaf paths in a minimal trie, called the *nonvoid external path length* [Knu98, SF13]. It is known that the expected value of the nonvoid external path length in a random trie is $n \log n + \Theta(n)$ [Knu98, Szp11, SF13]. We show the following:

Theorem 4. *There exists a constant $C > 0$ such that, for all $c > C$, the rate $R_0(\cdot)$ of the nonvoid external path length of a minimal trie on n infinite binary strings chosen independently and uniformly at random satisfies $R_0(cn) = \Theta(n)$.*

Note that Theorem 4 implies that the probability that the nonvoid external path length is more than $n \log n + dn$ is $e^{-\Theta(n)}$ (for a sufficiently large constant d).

1.2 Related Work

Showing that Bucket Sort runs in linear expected time when the keys are distributed independently and uniformly at random in $[0, 1)$ is a classic textbook result [CLRS09, MU17, SMDD19]. Bounds on the expectation as well as limiting distributions for the running time have also been studied for different versions of Bucket Sort [MFJR00, Dev86]. We are not aware of any work that directly addresses the rate of the running time of Bucket Sort. The upper and lower tails of the running time of Quick Sort have been studied in depth [Jan15, FJ02], including in the regime of large deviations [MH96].

The expected value of the nonvoid external path length of a trie is a classic result in applying the methods of analytic combinatorics to the analysis of algorithms [Knu98, Szp11, ML92, SF13, CFV01]. We consider the case in which the binary strings are independent and random (i.e., the bits are independent and unbiased). In [Knu98, Szp11, ML92, SF13] it is shown that for random strings, the expected value of the nonvoid external path length is $n \log n + \Theta(n)$. The variance of the nonvoid external path length and limiting distributions for it have also been studied extensively for different string distributions [JR88, KPS89, VF90].

In Knuth [Knu98, Section 5.2.2], the nonvoid external path length is shown to be proportional to the number of bit comparisons of radix exchange sort. The bound in Thm. 4 therefore applies to the rate of the number of bit comparisons of radix exchange sort when the strings are distributed independently and uniformly at random.

The connection between the running time of sorting algorithms and various trie parameters (including external path length) has also been studied by Seidel [Sei10], albeit in a significantly different model than ours. Specifically, [Sei10] analyzes the expected number of bit comparisons of Quick Sort and Merge Sort when the input is a randomly permuted set of strings sampled from a given distribution. In Seidel's model, the cost of comparing two strings is proportional to the length of their longest common prefix. Seidel shows that the running time of these algorithms can be naturally expressed in terms of parameters of the trie induced by the input strings. We emphasize that our analysis connects the running time of Bucket Sort to the excess path length in the comparison model (in which the cost of comparing two keys does not depend on their binary representation).

1.3 Paper Organization

Preliminaries and definitions are in Sec. 2. In Section 3, we present reductions from the running time of $b \log b$-Bucket Sort and the nonvoid external path length to the excess path length. The bound on the upper tail of the excess path length is proved in Sec. 4. Section 5 proves a lower bound on the rate of b^2-Bucket Sort. Upper bounds on the rates are proved in Appendix A. Theorems 2, 3 and 4 are completed in Sec. 6. Finally, in Sec. 7, we include a discussion on the difference between the rate of Bucket Sort and that of Quick Sort.

2 Preliminaries and Definitions

Bucket Sort. The input to Bucket Sort consists of n keys $X \triangleq \{x_1, \ldots, x_n\}$ in the interval $[0, 1)$. We define bucket j to be the set of keys in the interval $[j/n, (j+1)/n)$. Let $\mathbf{b}(X) \triangleq (B_0, \ldots B_{n-1})$ be the *occupancy vector* for input X, where B_j denotes the number of keys in X that fall in bucket j.

The buckets are separately sorted and the final output is computed by scanning the sorted buckets in increasing order. The initial assignment of keys to buckets and the final scanning of the sorted buckets takes $\Theta(n)$ time. We henceforth focus only on the time spent on sorting the keys in each bucket.

We consider the two natural options for sorting buckets: (i) Sort b keys in time $\Theta(b^2)$, using a sorting algorithm such as Insertion Sort or Bubble Sort. We refer to this option as b^2-Bucket Sort. (ii) Sort b keys in time $\Theta(b \log b)$ using a sorting algorithm such as Merge Sort or Heap Sort. We refer to this option as $b \log b$-Bucket Sort. Let $[n]$ denote the set $\{0, \ldots, n-1\}$ and let $\mathbf{b} = (B_0, \ldots, B_{n-1})$ denote an arbitrary occupancy vector. We define the functions

$$f(\mathbf{b}) \triangleq \sum_{j \in [n]} B_j^2 \qquad g(\mathbf{b}) \triangleq \sum_{j \in [n], B_j > 0} B_j \log B_j.$$

We let $T_{b^2}(X)$ and $T_{b \log b}(X)$ denote the running time on input X of b^2-Bucket Sort and $b \log b$-Bucket Sort, respectively. Then, $T_{b^2}(X) = \Theta(n + f(\mathbf{b}(X)))$ and $T_{b \log b}(X) = \Theta(n + g(\mathbf{b}(X)))$. [4]

Excess Path Length and Tries. We let $|\alpha|$ denote the length of a binary string $\alpha \in \{0, 1\}^*$. For a set L, let $|L|$ denote the cardinality of L.

Definition 5. *A set of strings $\{\alpha_1, \ldots, \alpha_s\}$ is* prefix-free *if, for every $i \neq j$, the string α_i is not a prefix of α_j.*

A *trie* is a rooted binary tree with edges labeled $\{0, 1\}$ such that two edges emanating from the same trie node are labeled differently. For a binary string α, let $\pi(\alpha)$ denote the trie node v, where the path from the root to v is labeled α. We say that a trie node u is a *predecessor* of v if u is in the path from the root to v. For a set U of trie nodes, the reduced trie that contains U is denoted by $T(U)$, namely, $T(U)$ consists of U and all the predecessors of nodes in U. Given a set of binary strings L, let $T(L)$ denote the trie $T(\pi(L))$. If the set L is prefix-free and contains only finite-length strings, then every node in $\pi(L)$ is a leaf of $T(L)$.

For a set L of prefix-free binary strings, let $\varphi_0(L)$ denote the set of minimal prefixes of strings in L subject to the constraint that $\varphi_0(L)$ is prefix-free. The trie $T(\varphi_0(L))$ is called the *minimal trie on L*. Note that the structure of $\varphi_0(L)$ (or of $T(\varphi_0(L))$) does not change if we append more bits to the strings in L.

The following definition extends the definition of $\varphi_0(L)$ by requiring that the prefixes have length at least k.

---

[4]Interestingly, the sum of squares of bin occupancies, i.e., $f(\mathbf{b})$, also appears in the FKS perfect hashing construction [FKS82].

Definition 6 (minimal k-prefixes)**.** *Let $L = \{\alpha_0, \ldots, \alpha_{n-1}\}$ denote a set of n distinct infinite binary strings. Given a parameter $k \geq 0$, the set of minimal k-prefixes of L, denoted by $\varphi_k(L) \triangleq \{\beta_0, \ldots, \beta_{n-1}\}$, is the set that satisfies the following properties:*

1. *for all $i \in [n]$, the string β_i is a prefix of α_i,*

2. *for all $i \in [n]$, $|\beta_i| \geq k$,*

3. *The set $\varphi_k(L)$ is prefix-free,*

4. $\sum_{i=0}^{n-1} |\beta_i|$ *is minimal among all sets that satisfy the first 3 conditions.*

The embedding of $\varphi_k(L)$ in a trie maps every string in $\varphi_k(L)$ to a distinct leaf of depth at least k. The definition can be modified to handle prefix-free sets of finite strings by appending an arbitrary infinite string (say, zeros) to each finite string. In this paper, we are interested in the following trie parameter defined on $\varphi_k(L)$:

Definition 7. *The k-excess path length $p_k(L)$ of a set L of n distinct infinite binary strings is defined as:*

$$p_k(L) \triangleq \sum_{\alpha \in \varphi_k(L)} (|\alpha| - k).$$

In [SF13], $p_0(L)$ is called the *nonvoid external path length* of the minimal trie on L. When $k = \lceil \log |L| \rceil$, we simply refer to $p_k(L)$ as the *excess path length* of L.

Distributions. Let $\mathcal{X}_n$ denote the uniform distribution over $[0,1)^n$. Note that if the set $X = \{x_0, \ldots, x_{n-1}\}$ is chosen according to $\mathcal{X}_n$, then $x_0, \ldots, x_{n-1}$ are chosen independently and uniformly at random from the interval $[0, 1)$. Let μ_{b^2} (res. $\mu_{b \log b}$) denote the expected ruuning time $T_{b^2}(X)$ (resp., $T_{b \log b}(X)$) when $X \sim \mathcal{X}_n$. Similarly, let μ_f (res. μ_g) denote the expected values of $f(\mathbf{b}(X))$ (resp., $g(\mathbf{b}(X))$) when $X \sim \mathcal{X}_n$. It is known that $\mu_f = 2n - 1$ (see [CLRS09, MU17]), and consequently, we have that $\mu_{b^2} = \Theta(n)$. Since $g \leq f$, we also have $\mu_g = \Theta(n)$ as well as $\mu_{b \log b} = \Theta(n)$.

Let $\mathcal{L}_n$ denote the uniform distribution over n infinite binary strings. Note that if $L = \{\alpha_0, \ldots, \alpha_{n-1}\}$ is chosen according to $\mathcal{L}_n$, then all the bits of the strings are independent and unbiased. We let μ_0 denote the expected value of the external nonvoid path $p_0(L)$ when $L \sim \mathcal{L}_n$. It is know that $\mu_0 = n \log n + \Theta(n)$ (see [Knu98, Szp11, SF13]).

Rates. Let $R_{b^2}(\cdot)$ (resp., $R_{b \log b}(\cdot)$) denote the rate of $T_{b^2}(X)$ (resp., $T_{b \log b}(X)$) when $X \sim \mathcal{X}_n$. Similarly, let $R_f(\cdot)$ (resp., $R_g(\cdot)$) denote the rate of $f(\mathbf{b}(X))$ (resp., $g(\mathbf{b}(X))$) when $X \sim \mathcal{X}_n$.

We first note that, to study the asymptotic behavior of R_{b^2} (for sufficiently large deviations) it suffices to study the asymptotic behavior of R_f. The proof of the following appears in Appendix B.

Observation 8. *For every $c > 0$, there exist constants $\delta_1 = \Theta(c)$ and $\delta_2 = \Theta(c)$ such that:*

$$R_f(\delta_1 \cdot n) \leq R_{b^2}(c \cdot n) \leq R_f(\delta_2 \cdot n).$$

An analogous statement holds for the rates $R_{b \log b}$ and R_g. The rate of the nonvoid external path length $p_0(L)$ is denoted by $R_0(\cdot)$.

3   Reductions

3.1   Balls-into-Bins Abstraction

We interpret the assignment of keys to buckets using a balls-into-bins abstraction. The keys correspond to balls, and the buckets correspond to bins. The assumption that $X \sim \mathcal{X}_n$ implies that the balls choose the bins independently and uniformly at random. The value B_j then equals the occupancy of bin j.

A similar balls-into-bins abstraction holds for the embedding of the minimal $(\log n)$-prefixes of $L \sim \mathcal{L}_n$ in a trie (assuming n is a power of 2). Indeed, let $\{v_0, \ldots, v_{n-1}\}$ denote the n nodes of the trie $T(L)$ at depth $\log n$. For a node v_j, we say that a string α *chooses* v_j, if the path labeled α contains v_j. Since the strings are random, each string chooses a node of depth $\log n$ independently and uniformly at random. Let C_j denote the number of strings in L who choose node v_j.[5] We refer to C_j as the *occupancy* of v_j with respect to L and define the vector $\mathbf{c}(L) \triangleq (C_0, \ldots, C_{n-1})$.

Observation 9. *When $X \sim \mathcal{X}_n$ and $L \sim \mathcal{L}_n$, the occupancy vector $\mathbf{b}(X)$ has the same joint probability distribution as $\mathbf{c}(L)$.*

3.2   Lower Bounding the Rate of $b \log b$-Bucket Sort

By Obs. 8, to prove a lower bound in $R_{b \log b}$ it suffices to prove a lower bound on R_g. In this section we show how to lower bound R_g by bounding the upper tail probability of the excess path length $p_{\log n}(L)$. We begin with the following observation about the nonvoid external path length $p_0(L)$:

Observation 10. *For every set L of n infinite prefix-free binary strings, $p_0(L) \geq n \log n$.*

Now consider an arbitrary vector $\mathbf{c}(L)$ and apply Observation 10 to each node of depth $\log n$ separately. We obtain the following corollary.

Corollary 11. *For every set L of n infinite prefix-free binary strings, $p_{\log n}(L) \geq g(\mathbf{c}(L))$.*

We lower bound the rate of $g(\mathbf{b}(X))$ as follows.

Lemma 12. *For every $c > 0$,*

$$\Pr_{X \leftarrow \mathcal{X}_n} \left[g(\mathbf{b}(X)) \geq \mu_g + cn \right] \leq \Pr_{L \leftarrow \mathcal{L}_n} \left[p_{\log n}(L) \geq cn \right] . \tag{1}$$

Proof. Recall that μ_g denotes the expected value of of $g(\mathbf{b}(X))$. Since $\mu_g > 0$, we have that $\Pr\left[g(\mathbf{b}(X)) \geq \mu_g + cn\right] \leq \Pr\left[g(\mathbf{b}(X)) \geq cn\right]$. Observation 9 implies that

$$\Pr_{X \leftarrow \mathcal{X}_n} \left[g(\mathbf{b}(X)) \geq cn \right] = \Pr_{L \leftarrow \mathcal{L}_n} \left[g(\mathbf{c}(L)) \geq cn \right] .$$

The claim then follows by Corollary 11. $\qquad\square$

Hence, a lower bound on the rate of $g(\mathbf{b}(X))$ follows by bounding the RHS of Eq. 1.

---

[5]Formally, $T(L)$ may contain a subset of these n nodes. If a node v_j at depth $\log n$ is not chosen by any string, then define $C_j = 0$.

3.3 Lower Bounding the Rate of the Nonvoid External Path Length

In this section, we show how to use the upper tail of $p_{\log n}(L)$ to lower bound the rate of the nonvoid external path length $p_0(L)$.

Observation 13. *For every set L of n infinite prefix-free binary strings, we have that:*

$$p_0(L) \leq n \log n + p_{\log n}(L).$$

Proof. The strings in $\varphi_0(L)$ are themselves prefixes of strings in $\varphi_{\log n}(L)$. We therefore get that $\sum_{\alpha \in \varphi_0(L)} |\alpha| \leq \sum_{\beta \in \varphi_{\log n}(L)} |\beta|$, and the claim follows. $\square$

Observation 10 implies that $\mu_0 \geq n \log n$. Together with Obs. 13 this implies that:

Corollary 14. *For every $L \sim \mathcal{L}_n$ and every $c > 0$.*

$$\Pr\left[p_0(L) \geq \mu_0 + cn\right] \leq \Pr\left[p_{\log n}(L) \geq cn\right].$$

4 The Upper Tail of the Excess Path Length

We bound the upper tail of $p_{\log n}(L)$ as follows:

Theorem 15. *Let $L \sim \mathcal{L}_n$. For every $c > 0$:*

$$\Pr\left[p_{\log n}(L) \geq (8c + 16) \cdot n\right] \leq \exp\left(-\frac{c - 1 - \ln c}{4} \cdot n\right).$$

Proof. Let $L = \{\alpha_1, \ldots, \alpha_n\}$ be a set of infinite random binary strings. We consider the evolution of the set $\varphi_{\log n}(L)$ of minimal $\log n$-prefixes as we process the strings α_i one by one. Specifically, let $L^{(i)} \triangleq \{\alpha_1, \ldots, \alpha_i\}$, for $1 \leq i \leq n$, and $L_0 = \emptyset$.

Let $\varphi(L^{(i)}) \triangleq \left\{ s_j \circ \delta_j^{(i)} \mid s_j \circ \delta_j^{(i)} \text{is a prefix of } \alpha_j \text{ and } |s_i| = \lceil \log n \rceil \text{ for } 1 \leq j \leq i \right\}$. Note that

$$p_{\log n}(L^{(i)}) = \sum_{j \in [i]} \left| \delta_j^{(i)} \right|.$$

We bound $p_{\log n}(L)$ by considering the increase $\Delta_i \triangleq p_{\log n}(L^{(i)}) - p_{\log n}(L^{(i-1)})$. Since $p_{\log n}(L^{(0)}) = 0$ and $p_{\log n}(L^{(n)}) = p_{\log n}(L)$, then $p_{\log n}(L) = \sum_{i=1}^{n} \Delta_i$.

The addition of the string α_i has two types of contributions to Δ_i. The first contribution is $\delta_i^{(i)}$. The second contribution is due to the need to extend colliding strings. Indeed, since the set $L^{(i-1)}$ is prefix-free, there exists at most one $j < i$ such that $s_j \circ \delta_j^{(i-1)}$ is a prefix of α_i. If $s_j \circ \delta_j^{(i-1)}$ is a prefix of α_i, then $\Delta_i = \left| \delta_j^{(i)} \right| - \left| \delta_j^{(i-1)} \right| + \left| \delta_i^{(i)} \right|$. Because $\delta_j^{(i)}$ and $\delta_i^{(i)}$ are minimal subject to being prefix-free, we also have that $\left| \delta_j^{(i)} \right| = \left| \delta_i^{(i)} \right|$. Hence, $\Delta_i \leq 2 \cdot \left| \delta_i^{(i)} \right|$. This implies that, for every τ:

$$\Pr\left[\Delta_i \geq 2\tau\right] \leq \Pr\left[\left| \delta_i^{(i)} \right| \geq \tau\right].$$

We now proceed to bound $\Pr\left[\left| \delta_i^{(i)} \right| \geq \tau\right]$. Fix $i \geq 1$ and let $\delta_i(\ell)$ denote the prefix of length ℓ of $\delta_i^{(i)}$. We denote by n_ℓ the number of leaves in the subtree rooted at $s_i \circ \gamma_i(\ell)$ in the trie $T(L^{(i-1)})$ (i.e., right before the string α_i is processed). Formally,

$$n_\ell \triangleq \left| \left\{ j < i \mid s_i \circ \delta_i(\ell) \text{ is a prefix of } s_j \circ \delta_j^{(i-1)} \right\} \right|.$$

7

Clearly, $n_0 = \left|\{j < i \mid s_i = s_j\}\right|$ and $n_{\left|\gamma_i^{(i)}\right|} = 0$. We bound $\left|\gamma_i^{(i)}\right|$ by bounding the minimum ℓ for which n_ℓ becomes zero as follows: define the binary random variable $Z_{\ell+1}$ to be 1 if $n_{\ell+1} \leq \frac{1}{2} \cdot n_\ell$, and 0 otherwise. Note that $\Pr[Z_\ell = 1] \geq 1/2$ and that $\{Z_\ell\}_\ell$ are independent. By definition,

$$\left|\delta_i^{(i)}\right| \geq \tau \implies \sum_{s=1}^{\tau} Z_s \leq \log(1 + n_0) \ . \tag{2}$$

By the law of total probability,

$$\Pr\left[\left|\delta_i^{(i)}\right| \geq \tau\right] \leq \Pr\left[\log(1 + n_0) \geq \tau/8\right] + \Pr\left[\left(\left|\delta_i^{(i)}\right| \geq \tau\right) \mid \log(1 + n_0) \leq \tau/8\right] \ . \tag{3}$$

We now bound the two terms in the RHS of Eq. 3. Note that $\mathbb{E}[n_0] \leq 1$. In fact $\mathbb{E}\left[n_0 \mid \bigwedge_{j<i} \Delta_j = \xi_j\right] \leq 1$ for every realization $\{\xi_j\}_{j<i}$ of $\{\Delta_j\}_{j<i}$. By Markov's inequality:

$$\Pr\left[1 + n_0 \geq 2^{\tau/8}\right] \leq \frac{1 + \mathbb{E}[n_0]}{2^{\tau/8}} \leq 2^{-\tau/8+1} \ . \tag{4}$$

To bound the second term in the RHS of Eq. 3, we apply the Chernoff bound in Eq. 18:

$$\Pr\left[\left|\delta_i^{(i+1)}\right| \geq \tau \mid \log(1 + n_0) \leq \tau/8\right] \leq \Pr\left[\sum_{s=1}^{\tau} Z_s \leq \frac{\tau}{8}\right] \qquad \text{(By Eq. 2)}$$

$$\leq \Pr\left[\sum_{s=1}^{\tau} Z_s \leq \frac{2}{8} \cdot \mathbb{E}\left[\sum_{s=1}^{\tau} Z_s\right]\right] \qquad (\mathbb{E}[Z_i] \geq 1/2)$$

$$\leq \exp\left(-\frac{1}{2} \cdot \mathbb{E}\left[\sum_{s=1}^{\tau} Z_s\right] \cdot \left(1 - \frac{2}{8}\right)^2\right)$$

$$\leq \exp\left(-\frac{\tau}{4} \cdot \left(\frac{3}{4}\right)^2\right) = \exp\left(-\frac{9}{64} \cdot \tau\right) \ . \tag{5}$$

From Equations 3 – 5, it follows that:

$$\Pr\left[\left|\delta_i^{(i+1)}\right| \geq \tau\right] \leq 2^{-\tau/8+1} + \exp(-9\tau/64) \leq 2^{-\tau/8+2} \ .$$

Therefore,

$$\Pr\left[\Delta_i \geq 16 \cdot (\tau + 2))\right] \leq 2^{-\tau} \ . \tag{6}$$

Note that Eq. 6 also holds under every conditioning on the realizations of $\{\Delta_j\}_{j<i}$.

Let $\Delta_i' \triangleq \frac{1}{16} \cdot \Delta_i - 1$ and note that $\Pr[\Delta_i' \geq \tau] \leq 2^{-\tau+1}$. Let $\{G_i\}_i$ denote independent geometric random variables, where $G_i \sim Ge(1/2)$. Since $\Pr[G_i \geq \tau] = 2^{-(\tau-1)}$, we conclude that Δ_i' is stochastically dominated by G_i. In fact, the random variables $\{\Delta_i'\}_{i\in[f]}$ are unconditionally sequentially dominated by $\{G_i\}_{i\in[n]}$. By [Doe18, Lemma 8.8], it follows that $\sum_{i\in[n]} \Delta_i'$ is stochastically dominated by $\sum_{i\in[n]} G_i$.[6]

---

[6] Note that RVs $\{\Delta\}_i$ are not independent and probably not even negatively associated. Hence, standard concentration bounds do not apply to $\sum \Delta_i$.

The sum of independent geometric random variables is concentrated [Jan18] and so we get:

$$\Pr\left[\sum_{i\in[n]}\Delta'_i \geq c\cdot n/2\right] \leq \Pr\left[\sum_{i\in[n]}G_i \geq c\cdot n/2\right]$$

$$\leq \exp\left(-\frac{c-1-\ln c}{4}\cdot n\right)$$

as required. □

5 Lower Bound for b^2-Bucket Sort

This section deals with proving the following lower bound on the rate R_f. By Obs. 8, this also implies a lower bound on the rate R_{b^2}.

Lemma 16. *There exists a constant $C > 0$ such that, for all $c > C$, we have that $R_f(cn) = \Omega(\sqrt{n}\log n)$, for all sufficiently large n.*

5.1 Preliminaries

Given an input X of n keys and its associated occupancy vector $\mathbf{b}(X) = (B_0, B_1, \ldots, B_{n-1})$, define $\mathcal{S}_i \triangleq \{j \in [n] \mid B_j \geq i\}$ to be the set of buckets with at least i keys assigned to them. Note that the random variables $\{|\mathcal{S}_i|\}_i$ are negatively associated because they are monotone functions of bin occupancies, which are a classical example of negatively associated RVs [DP09].

Claim 17. *For every occupancy vector $(B_0, B_1, \ldots, B_{n-1})$, the following holds:*

$$\sum_{j\in[n]}\binom{|B_j|+1}{2} = \sum_{i\in[n+1]}i\cdot|\mathcal{S}_i| \ . \tag{7}$$

Proof. Consider an $n \times n$ matrix A filled according to the following rule:

$$A_{i,j} \triangleq \begin{cases} i & \text{if } B_j \geq i \\ 0 & \text{otherwise.} \end{cases}$$

Let $S \triangleq \sum_{i,j}A_{i,j}$. The sum of entries in column j equals $\binom{|B_j|+1}{2}$. On the other hand, the sum of entries in row i equals $i\cdot|\mathcal{S}_i|$. Hence both sides of Eq. 7 equal S, and the claim follows. □

Lemma 18 states that, in order to prove Lemma 16, it suffices to prove a lower bound on the upper tail probability of the random variable $\sum_{i\in[n+1]}i\cdot|\mathcal{S}_i|$. Specifically, , we get that:

Lemma 18. *For every c, we have that*

$$\Pr\left[f(\mathbf{b}(X)) \geq \mu_f + cn\right] = \Pr\left[\sum_{i\in[n+1]}i\,|\mathcal{S}_i| \geq \frac{(3+c)n-1}{2}\right] \ .$$

Proof. By Claim 17, $f(\mathbf{b}(X)) = 2\cdot\sum_{i\in[n+1]}i\,|\mathcal{S}_i| - n$. The Lemma follows from the fact that $\mu_f = 2n-1$ [CLRS09, MU17]. □

9

Next, we upper bound $\mathbb{E}\left[|\mathcal{S}_i|\right]$. Let $E_i \triangleq \left(\frac{e}{i}\right)^i$ and note the following:

Claim 19. *For every $i \in \{1, \ldots, n\}$, we have that $\mathbb{E}\left[|\mathcal{S}_i|\right] \leq n \cdot E_i$.*

Proof. Fix i and let $X_{i,j}$ be the indicator random variable that is 1 if $B_j \geq i$ and 0 otherwise. We get that $|\mathcal{S}_i| = \sum_j X_{i,j}$. Because each key chooses a bucket independently and uniformly at random, we have that:

$$\Pr\left[B_j \geq i\right] \leq \binom{n}{i} \cdot \left(\frac{1}{n}\right)^i \leq \left(\frac{en}{i}\right)^i \cdot \left(\frac{1}{n}\right)^i = \left(\frac{e}{i}\right)^i = E_i \ .$$

The claim follows by linearity of expectation. $\qquad\square$

One can analytically show that:

Observation 20. $\sum_{i=1}^{\infty} i \cdot E_i \leq 10$.

5.2 Applying Chernoff Bounds in Different Regimes

In the proof of Lemma 16, we consider three thresholds on bin occupancies $\tau_1 \leq \tau_2 \leq \tau_3$ defined as follows:

$$\tau_1 \triangleq \max\left\{i \,\Big|\, E_i \geq \frac{c \log n}{\sqrt{n}}\right\} \ , \qquad \tau_2 \triangleq \frac{n^{1/4}}{\sqrt{\log n}} \ , \qquad \tau_3 \triangleq \sqrt{n} \ .$$

Claim 21. *For every $c > 0$, there exists a $\gamma = \gamma(c) > 0$, such that:*

$$\Pr\left[\sum_{i \leq \tau_1} i \cdot |\mathcal{S}_i| \geq cn + \sum_{i \leq \tau_1} i \cdot E_i \cdot n\right] \leq \exp\left(-\gamma\sqrt{n}\log n\right) \ .$$

Proof. Fix $i \leq \tau_1$. By the Chernoff bounds (Eq. 15-16 in Appendix C) and the definition of τ_1, for every $\delta > 0$, there exists a $c' = c'(\delta) > 0$, such that:

$$\Pr\left[|\mathcal{S}_i| \geq (1+\delta)E_i \cdot n\right] \leq \exp\left(-c' \cdot E_i \cdot n\right) \leq \exp(-c' \cdot c \cdot \sqrt{n} \cdot \log n) \ .$$

By applying a union bound over all $i \leq \tau_1$, it follows that there exists a $\gamma > 0$ such that:

$$\Pr\left[\sum_{i \leq \tau_1} i \, |\mathcal{S}_i| \geq (1+\delta) \cdot \sum_{i \leq \tau_1} iE_i \cdot n\right] \leq \exp\left(-\gamma\sqrt{n}\log n\right) \ .$$

Define $\delta \triangleq c/10$. By Obs. 20, $\delta \sum_i iE_i \leq c$, and the claim follows. $\qquad\square$

Claim 22. *For every $c > 0$, there exists a $\gamma = \gamma(c) > 0$ such that for n sufficiently large:*

$$\Pr\left[\sum_{i=\lceil \tau_1 \rceil}^{\lfloor \tau_2 \rfloor} i \, |\mathcal{S}_i| \geq cn + \sum_{i=\lceil \tau_1 \rceil}^{\lfloor \tau_2 \rfloor} iE_i \cdot n\right] \leq \exp\left(-\gamma\sqrt{n}\log n\right) \ .$$

Proof. For every $\tau_1(c) < i \leq \tau_2$, define $\delta_i \triangleq (c \log n)/(E_i \sqrt{n})$ so that

$$\sum_{i \leq \tau_2} \delta_i \cdot i E_i = \sum_{i \leq \tau_2} i \cdot \frac{c \log n}{\sqrt{n}} \leq (\tau_2)^2 \cdot \frac{c \log n}{\sqrt{n}} = c . \tag{8}$$

Since $\delta_i > 1$ for every $i > \tau_1$, by the Chernoff bound in Eq. 16:

$$\Pr \left[|\mathcal{S}_i| > (1 + \delta_i) \cdot E_i \cdot n \right] \leq \exp \left(-\delta_i \cdot n \cdot E_i / 3 \right)$$
$$= \exp \left(-c/3 \cdot \sqrt{n} \log n \right) .$$

By applying a union bound over all $\tau_1 \leq i \leq \tau_2$, it follows that there exists a constant $\gamma > 0$ such that:

$$\Pr \left[\sum_{i=\lceil \tau_1 \rceil}^{\lfloor \tau_2 \rfloor} i \, |\mathcal{S}_i| \geq \sum_{i=\lceil \tau_1 \rceil}^{\lfloor \tau_2 \rfloor} (1 + \delta_i) \cdot i E_i \cdot n \right] \leq \exp \left(-\delta \sqrt{n} \log n \right) . \tag{9}$$

The claim follows by Eq. 8 and 9. $\qquad \square$

Claim 23. *For every $c > 0$, there exists a $\gamma = \gamma(c) > 0$ such that for sufficiently large n, we have that:*

$$\Pr \left[\sum_{i=\lceil \tau_2 \rceil}^{\lfloor \tau_3 \rfloor} i \, |\mathcal{S}_i| > cn + \sum_{i=\lceil \tau_2 \rceil}^{\lfloor \tau_3 \rfloor} i E_i \cdot n \right] \leq \exp \left(-\gamma \sqrt{n} \log(n) \right) .$$

Proof. For every $\tau_2 \leq i \leq \tau_3$, define $\delta_i \triangleq \frac{c}{5} \cdot \frac{\log n}{i \log i \cdot E_i \cdot \sqrt{n}}$ so that the following holds for sufficiently large n:

$$\sum_{i=\lceil \tau_2 \rceil}^{\lfloor \tau_3 \rfloor} \delta_i \cdot i E_i = \frac{c}{5} \cdot \left(\sum_{i=\lceil \tau_2 \rceil}^{\lfloor \tau_3 \rfloor} \frac{1}{\log i} \right) \cdot \frac{\log n}{\sqrt{n}} \tag{10}$$

$$\leq \frac{c}{5} \cdot \frac{\tau_3}{\log \tau_2} \cdot \frac{\log n}{\sqrt{n}} \leq \frac{c}{5} \cdot \frac{\log n}{0.25 \log n - 0.5 \log \log n} \leq c . \tag{11}$$

For a sufficiently large n, it holds that $\delta_i > 1$; moreover $\log \delta_i \geq \Omega(i \log i)$ for every $\tau_2 \leq i \leq \tau_3$. By the Chernoff bound in Eq. 17:

$$\Pr \left[|\mathcal{S}_i| > (1 + \delta_i) \cdot n \cdot E_i \right] \leq \exp \left(-\delta_i \ln(\delta_i) \cdot n \cdot E_i / 2 \right) \leq \exp \left(-\Omega(\sqrt{n} \log n) \right) .$$

By applying a union bound over all $\tau_2 \leq i \leq \tau_3$, it follows that there exists a $\delta(c) > 0$ such that:

$$\Pr \left[\sum_{i=\lceil \tau_2 \rceil}^{\lfloor \tau_3 \rfloor} i \, |\mathcal{S}_i| > \sum_{i=\lceil \tau_2 \rceil}^{\lfloor \tau_3 \rfloor} (1 + \delta_i) \cdot i \cdot E_i \cdot n \right] \leq \exp \left(-\delta \sqrt{n} \log n \right) . \tag{12}$$

The claim follows by Eq. 11 and 12. $\qquad \square$

Claim 24. *For every $c > 0$, there exists a $\gamma = \gamma(c) > 0$ such that for sufficiently large n, we have that:*

$$\Pr \left[\sum_{i=\lceil \tau_3 \rceil}^{n} i \, |\mathcal{S}_i| > cn \right] \leq \exp \left(-\gamma \sqrt{n} \log n \right) .$$

11

Proof. We apply Markov's inequality and get that there exists a $c' > 0$ such that:

$$\Pr\left[|\mathcal{S}_i| \geq \frac{c}{i}\right] \leq \frac{i \cdot E_i \cdot n}{c} \leq \exp\left(-c'\sqrt{n}\log n\right) ,$$

where the last inequality holds because $i \geq \tau_3$. The claim follows by applying a union bound over $i \geq \tau_3$. $\qquad\square$

5.3 Proof of Lemma 16

In order to prove Lemma 16, we apply a union bound over the Claims 21–24 and use Obs. 20. It follows that for every $c \geq 0$, there exists a $\gamma = \gamma(c) > 0$ such that:

$$\Pr\left[\sum_{i=1}^{n} i\,|\mathcal{S}_i| > (10 + c) \cdot n\right] \leq \exp\left(-\gamma\sqrt{n}\log n\right) .$$

Lemma 16 then follows from Lemma 18.

6   Proof of Theorems 2, 3 and 4

To prove Theorems 2 and 3, we employ Obs. 8 that shows a reduction from R_f (and R_g, respectively) to R_{b^2} (and $R_{b\log b}$ respectively). The lower bound for R_f is discussed in Lemma 16. The lower bound for R_g follows from Lemma 1 and Theorem 15. The lower bound for R_0 follows from Cor. 14 and Theorem 15. Finally, we apply Lemma 25 to get matching upper bounds on R_f, R_g and R_0.

7   Discussion: Comparison to Quick Sort

We note that the rate of the running time of Quick Sort is smaller than that of Bucket Sort [MH96]. Here, we refer to the version of Quick Sort that picks a pivot x uniformly at random and then recurses on two subsets: the set of elements smaller than x and the set of elements greater than x. Let $T_{qs}(n)$ be the number of comparisons that Quick Sort makes on n randomly permuted distinct keys. The expectation of $T_{qs}(n)$ is denoted by μ_{qs} and equals $\Theta(n \log n)$. McDiarmid and Hayward [MH96] prove that for $\frac{1}{\ln n} < \varepsilon \leq 1$:

$$\Pr\left[|T_{qs}(n) - \mu_{qs}| \geq \varepsilon\mu_{qs}\right] = n^{-2\varepsilon(\ln\ln n - \ln(1/\varepsilon) + O(\log\log\log n))} .$$

Setting $\varepsilon = c/\ln n$ (for $c > 1$) implies that the rate R_{qs} of $T_{qs}(n)$ satisfies $R_{qs}(cn) = O(\log\log\log n)$.

One may wonder why the upper tails of Quick Sort and Bucket Sort exhibit different rates. We provide some intuition by examining the distributions of occupancies induced by Quick Sort and Bucket Sort on nodes of depth $\log n$ in a complete binary tree. Consider occupancies defined by the Quick Sort recursion tree as follows. In each recursive call, the pivot "stays" in the inner node, and the two lists are sent to the left and right children. Hence, every node is assigned a (possibly empty) list of keys. We refer to the distribution of occupancies across the n nodes of depth $\log n$ as the *Quick Sort distribution*.

The number of comparisons $T_{qs}(n)$ is bounded by $n \log n$ (a bound on the number of comparisons until level $\log n$) plus the comparisons starting from level $\log n$. Clearly, the number of comparisons starting from level $\log n$ depends on the Quick Sort distribution.

The Quick Sort distribution is very far from the distribution of $\mathbf{b}(X)$ when $X \sim \mathcal{X}_n$ (i.e., the occupancy vector in Bucket Sort when the n keys are distributed uniformly at random). Specifically,

consider the event Z that the occupancies of the $n/2$ nodes of depth $\log n$ in the left subtree are all zeros. In the Quick Sort distribution, the probability of event Z is at least $1/n$, e.g., Z occurs if the first pivot is the smallest element. In the Bucket Sort distribution, the probability of event Z is 2^{-n} (i.e., all the keys are in the interval $(1/2, 1)$).

Acknowledgments

References

[CFV01]   Julien Clément, Philippe Flajolet, and Brigitte Vallée. Dynamical sources in information theory: A general analysis of trie structures. *Algorithmica*, 29(1-2):307–369, 2001.

[CLRS09]   Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to algorithms*. MIT press, 2009.

[Dev86]   Luc Devroye. *Lecture notes on bucket algorithms*, volume 12. Birkhäuser Boston, 1986.

[Doe18]   Benjamin Doerr. Probabilistic tools for the analysis of randomized optimization heuristics. *CoRR*, abs/1801.06733, 2018.

[DP09]   Devdatt P Dubhashi and Alessandro Panconesi. *Concentration of measure for the analysis of randomized algorithms*. Cambridge University Press, 2009.

[FJ02]   James Allen Fill and Svante Janson. Quicksort asymptotics. *Journal of Algorithms*, 44(1):4–28, 2002.

[FKS82]   Michael L Fredman, János Komlós, and Endre Szemerédi. Storing a sparse table with $o(1)$ worst case access time. In *23rd Annual Symposium on Foundations of Computer Science*, pages 165–169. IEEE, 1982.

[Jan15]   Svante Janson. On the tails of the limiting quicksort distribution. *Electronic Communications in Probability*, 20, 2015.

[Jan18]   Svante Janson. Tail bounds for sums of geometric and exponential variables. *Statistics & Probability Letters*, 135:1–6, 2018.

[JR88]   Philippe Jacquet and Mireille Regnier. Normal limiting distribution for the size and the external path length of tries. 1988.

[Knu98]   Donald Ervin Knuth. *The art of computer programming, , Volume III, 2nd Edition*. Addison-Wesley, 1998.

[KPS89]   Peter Kirschenhofer, Helmut Prodinger, and Wojciech Szpankowski. On the variance of the external path length in a symmetric digital trie. *Discrete Applied Mathematics*, 25(1-2):129–143, 1989.

[MFJR00] Hosam Mahmoud, Philippe Flajolet, Philippe Jacquet, and Mireille Régnier. Analytic variations on bucket selection and sorting. *Acta Informatica*, 36(9-10):735–760, 2000.

[MH96]   Colin McDiarmid and Ryan Hayward. Large deviations for quicksort. *J. Algorithms*, 21(3):476–507, 1996.

[ML92]   Hosam M Mahmoud and George S Lueker. *Evolution of random search trees*, volume 200. Wiley New York, 1992.

[MU17]   Michael Mitzenmacher and Eli Upfal. *Probability and computing: randomization and probabilistic techniques in algorithms and data analysis. UK*. Cambridge University Press, 2017.

[Sei10]   Raimund Seidel. Data-specific analysis of string sorting. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete algorithms*, pages 1278–1286. Society for Industrial and Applied Mathematics, 2010.

[SF13]   Robert Sedgewick and Philippe Flajolet. *An introduction to the analysis of algorithms*. Pearson Education India, 2013.

[SMDD19] Peter Sanders, Kurt Mehlhorn, Martin Dietzfelbinger, and Roman Dementiev. Sorting and selection. In *Sequential and Parallel Algorithms and Data Structures*, pages 153–210. Springer, 2019.

[Szp11]   Wojciech Szpankowski. *Average case analysis of algorithms on sequences*, volume 50. John Wiley & Sons, 2011.

[VF90]   Jeffrey Scott Vitter and Philippe Flajolet. Average-case analysis of algorithms and data structures. In *Handbook of Theoretical Computer Science, Volume A: Algorithms and Complexity*, pages 431–524. 1990.

A   Upper Bounds

Lemma 25. *For every constant $c > 0$, the following hold:*

1. $R_f(cn) = O(\sqrt{n}\log n)$,

2. $R_g(cn) = O(n)$, *and*

3. $R_0(cn) = O(n)$.

Proof. To prove statements $1 - 2$, we define $\mathcal{A}_{i,j}$ to be the event that $B_j = i$ (occupancy of bin j equals i) and note that:

$$\Pr\left[\mathcal{A}_{i,j}\right] = \binom{n}{i} \cdot \left(\frac{1}{n}\right)^i \cdot \left(1 - \frac{1}{n}\right)^{n-i} \geq \left(\frac{n}{i}\right)^i \cdot \left(\frac{1}{n}\right)^i \cdot \left(1 - \frac{1}{n}\right)^n$$

$$\geq \frac{1}{4} \cdot \left(\frac{1}{i}\right)^i = 2^{-2-i\log i} .$$

Fix a bin j. It follows that:

$$\Pr\left[f(\mathbf{b}(X)) \geq cn\right] \geq \Pr\left[\mathcal{A}_{\sqrt{cn},j}\right] \geq \exp\left(-\Omega(\sqrt{n}\log n)\right) .$$

This implies that $R_f(cn) = O(\sqrt{n}\log n)$ for every constant $c > 0$.

For i such that $i \log i = \Omega(n)$ (i.e., $i = \Theta(n/\log n)$), we have that:

$$\Pr\left[g(\mathbf{b}(X)) = \Omega(n)\right] \geq \Pr\left[\mathcal{A}_{i,j}\right] \geq \exp\left(-\Omega(n)\right) .$$

This implies that $R_g(cn) = O(n)$ for every constant $c > 0$.

Now we consider lower bounding $\Pr\left[p_0(L) \geq \mu_0 + cn\right]$ for every $c > 0$ (statement 3). Consider the event $\mathcal{A}$ in which the set L contains two binary strings α_1 and α_2 that share a common prefix of length $\left(\frac{c}{2}+1\right) \cdot n$. When $L \sim \mathcal{L}_n$, we have that $\Pr\left[\mathcal{A}\right] \geq 2^{-(c/2+1)n}$.

On the other hand, we have that, if event $\mathcal{A}$ happens, then the depth of nodes $\pi(\alpha_1)$ and $\pi(\alpha_2)$ is more than $(c/2+2) \cdot n$ in the trie $T(\varphi_0(L))$ (i.e., at least $c/2+2$ bits are required to separate α_1 and α_2). For the rest $n-2$ binary strings, we use Obs. 10 and get that we needed at least $(n-2)\log(n-2)$ bits to separate them. Since $\mu_0 \leq n \log n + 2n$, we get that event $\mathcal{A}$ implies that

$$p_0(L \mid \mathcal{A}) \geq 2 \cdot \left(\frac{c}{2}+2\right) \cdot n + (n-2)\log(n-2) \geq n \log n + 2n + cn \geq \mu_0 + cn .$$

In other words, $\Pr\left[p_0(L) \geq \mu_0 + cn\right] \geq \Pr\left[A\right] \geq 2^{-(c/2+1)n}$, hence $R_0(cn) = O(n)$. $\qquad\square$

B Proof of Observation 8

Proof. Consider the first inequality that we need to prove:

$$R_f(\delta_1 n) \leq R_{b^2}(cn) . \tag{13}$$

Recall that $\mu_f = \mathbb{E}\left[f(\mathbf{b}(X))\right] = 2n - 1$, and that $\mu_{b^2} = \mathbb{E}\left[T_{b^2}(X)\right]$. Equation 13 is equivalent to the inequality (where $X \sim \mathcal{X}_n$).

$$\Pr\left[T_{b^2}(X) \geq \mu_{b^2} + cn\right] \leq \Pr\left[f(\mathbf{b}(X)) \geq \mu_f + \delta_1 n\right] .$$

Recall that $T_{b^2}(X) = \Theta(n + f(\mathbf{b}(X)))$. Let $0 < c_1 \leq c_2$ be constants such that, for sufficiently large values of n,

$$c_1 \cdot (n + f(\mathbf{b}(X))) \leq T_{b^2}(X) \leq c_2 \cdot (n + f(\mathbf{b}(X))) .$$

Let $\delta_1 \triangleq \frac{c - 3(c_2 - c_1)}{c_2}$. Note that $\delta_1 = \Theta(c)$. Then,

$$
\begin{aligned}
\Pr\left[T_{b^2}(X) \geq \mu_{b^2} + c \cdot n\right] &\leq \Pr\left[c_2 \cdot (n + f(\mathbf{b}(X))) \geq \mu_{b^2} + c \cdot n\right] \\
&\leq \Pr\left[c_2 \cdot (n + f(\mathbf{b}(X))) \geq c_1 n + c_1 \mu_f + cn\right] \\
&\leq \Pr\left[f(\mathbf{b}(X)) \geq \mu_f + \frac{c - 3(c_2 - c_1)}{c_2} \cdot n\right] \\
&= \Pr\left[f(\mathbf{b}(X)) \geq \mu_f + \delta_1 n\right] .
\end{aligned}
$$

The second inequality is proved in a similar fashion. $\qquad\square$

C Variants of Chernoff Bounds

Theorem 26. *Let $X_1, \ldots, X_n$ be independent binary random variables . Let $X = \sum_{i=1}^{n} X_i$ and $\mu = \mathbb{E}[X]$. Then the following Chernoff bounds hold:*

1. For every $\delta > 0$:

$$\Pr\left[X \geq (1 + \delta)\mu\right] \leq \left(\frac{e^\delta}{(1 + \delta)^{(1+\delta)}}\right)^\mu . \tag{14}$$

2. For $0 < \delta \leq 1$,

$$\Pr\left[X \geq (1 + \delta)\mu\right] \leq e^{-\mu\delta^2/3} . \tag{15}$$

3. For $\delta \geq 1$,

$$\Pr\left[X \geq (1 + \delta)\mu\right] \leq e^{-\mu\delta/3} . \tag{16}$$

4. For $\delta \geq e$,

$$\Pr\left[X \geq (1 + \delta)\mu\right] \leq e^{-\mu\delta\ln(\delta)/2} . \tag{17}$$

5. For $0 < \delta \leq 1$,

$$\Pr\left[X \geq (1 - \delta)\mu\right] \leq e^{-\mu\delta^2/3} . \tag{18}$$

Proof. The bounds in Eqs. 14, 15 and 18 are proved in [MU17]. For $\delta > 0$, define the function $f(\delta) \triangleq (1 + \delta)\ln(1 + \delta) - \delta$ and note that Eq. 14 states that $\Pr\left[X \geq (1 + \delta)\mu\right] \leq \exp(-\mu \cdot f(\delta))$. For $\delta \geq 1$, we have that $f(\delta) \geq \delta/3$, which proves Eq. 16. For every $\delta > 0$, $f(\delta) \geq \delta\ln(\delta)/2$, which proves Eq. 17. The bound in Eq. 17 and its proof also appear in Chapter 10.1.1 in [Doe18]. □

We note that the bounds 14–17 hold even when the parameter μ is an upper bound on $\mathbb{E}[X]$. Moreover, they also hold when the random variables $X_1, \ldots, X_n$ are negatively associated [DP09, Thm. 3.1]. Indeed, in the proofs of the Claims 21 – 24, we apply Eq. 14 – 17 to the random variable $|S_i|$, which is a sum of negatively associated indicator variables.