On Smoothed Analysis of Quicksort and Hoare's Find

Mahmoud Fouz[1], Manfred Kufleitner[2], Bodo Manthey[1], and Nima Zeini Jahromi[1]

[1] Saarland University, Department of Computer Science
Postfach 151150, 66041 Saarbrücken, Germany
mfouz/manthey@cs.uni-saarland.de, nzeini@studcs.uni-saarland.de
[2] Universität Stuttgart, FMI
Universitätsstraße 38, 70569 Stuttgart, Germany
manfred.kufleitner@fmi.uni-stuttgart.de

Abstract. We provide a smoothed analysis of Hoare's find algorithm and we revisit the smoothed analysis of quicksort. Hoare's find algorithm – often called quickselect – is an easy-to-implement algorithm for finding the k-th smallest element of a sequence. While the worst-case number of comparisons that Hoare's find needs is $\Theta(n^2)$, the average-case number is $\Theta(n)$. We analyze what happens between these two extremes by providing a smoothed analysis of the algorithm in terms of two different perturbation models: additive noise and partial permutations.
In the first model, an adversary specifies a sequence of n numbers of $[0,1]$, and then each number is perturbed by adding a random number drawn from the interval $[0,d]$. We prove that Hoare's find needs $\Theta(\frac{n}{d+1}\sqrt{n/d}+n)$ comparisons in expectation if the adversary may also specify the element that we would like to find. Furthermore, we show that Hoare's find needs fewer comparisons for finding the median.
In the second model, each element is marked with probability p and then a random permutation is applied to the marked elements. We prove that the expected number of comparisons to find the median is in $\Omega\big((1-p)\frac{n}{p}\log n\big)$, which is again tight.
Finally, we provide lower bounds for the smoothed number of comparisons of quicksort and Hoare's find for the median-of-three pivot rule, which usually yields faster algorithms than always selecting the first element: The pivot is the median of the first, middle, and last element of the sequence. We show that median-of-three does not yield a significant improvement over the classic rule: the lower bounds for the classic rule carry over to median-of-three.

1 Introduction

To explain the discrepancy between average-case and worst-case behavior of the simplex algorithm, Spielman and Teng introduced the notion of *smoothed analysis* [17]. Smoothed analysis interpolates between average-case and worst-case analysis: Instead of taking a worst-case instance, we analyze the expected worst-case running time subject to slight random perturbations. The more influence

Proc. 15th Int. Comp. and Combinatorics Conf. (COCOON 2009) – © Springer

we allow for perturbations, the closer we come to the average case analysis of the algorithm. Therefore, smoothed analysis is a hybrid of worst-case and average-case analysis. In practice, neither can we assume that all instances are equally likely, nor that instances are precisely worst-case instances. The goal of smoothed analysis is to capture the notion of a *typical* instance mathematically. Typical instances are, in contrast to worst-case instances, often subject to measurement or rounding errors. On the other hand, typical instances still have some (adversarial) structure, which instances drawn completely at random do not. Spielman and Teng [18] give a survey of results and open problems in smoothed analysis.

In this paper, we provide a smoothed analysis of Hoare's find [7], which is a simple algorithm for finding the k-th smallest element of a sequence of numbers: Pick the first element as the pivot and compare it to all $n-1$ remaining elements. Assume that $\ell - 1$ elements are smaller than the pivot. If $\ell = k$, then the pivot is the element that we are looking for. If $\ell > k$, then we recurse to find the k-th smallest element of the list of the smaller elements. If $\ell < k$, then we recurse to find the $(k - \ell)$-th smallest element among the larger elements. The number of comparisons to find the specified element is $\Theta(n^2)$ in the worst case and $\Theta(n)$ on average. Furthermore, the variance of the number of comparisons is $\Theta(n^2)$ [8]. As our first result, we close the gap between the quadratic worst-case running-time and the expected linear running-time by providing a smoothed analysis.

Hoare's find is closely related to quicksort [6], which needs $\Theta(n^2)$ comparisons in the worst case and $\Theta(n \log n)$ on average [10, Section 5.2.2]. The smoothed number of comparisons that quicksort needs has already been analyzed [12]. Choosing the first element as the pivot element, however, results in poor running-time if the sequence is nearly sorted. There are two common approaches to circumvent this problem: First, one can choose the pivot randomly among the elements. However, randomness is needed to do so, which is sometimes expensive. Second, without any randomness, a common approach to circumvent this problem is to compute the median of the first, middle, and last element of the sequence and then to use this median as the pivot [15,16]. This method is faster in practice since it yields more balanced partitions and it makes the worst-case behavior much more unlikely [10, Section 5.5]. It is also faster both in average and in worst case, albeit only by constant factors [4, 14]. Quicksort with the median-of-three rule is widely used, for instance in the `qsort()` implementation in the GNU standard C library `glibc` [13] and also in a recent very efficient implementation of quicksort on a GPU [2]. The median-of-three rule has also been used for Hoare's find, and the expected number of comparisons has been analyzed precisely [9]. Our second goal is a smoothed analysis of both quicksort and Hoare's find with the median-of-three rule to get a thorough understanding of this variant of these two algorithms.

1.1 Preliminaries

We denote *sequences* of real numbers by $s = (s_1, \ldots, s_n)$, where $s_i \in \mathbb{R}$. For $n \in \mathbb{N}$, we set $[n] = \{1, \ldots, n\}$. Let $U = \{i_1, \ldots, i_\ell\} \subseteq [n]$ with $i_1 < i_2 < \ldots < i_\ell$.

Then $s_U = (s_{i_1}, s_{i_2}, \ldots, s_{i_\ell})$ denotes the *subsequence* of s of the elements at positions in U. We denote probabilities by $\mathbb{P}$ and expected values by $\mathbb{E}$.

Throughout the paper, we will assume for the sake of clarity that numbers like $\sqrt{n}$ are integers and we do not write down the tedious floor and ceiling functions that are actually necessary. Since we are interested in asymptotic bounds, this does not affect the validity of the proofs.

Pivot Rules. Given a sequence s, a *pivot rule* simply selects one element of s as the *pivot element*. The pivot element will be the one to which we compare all other elements of s. In this paper, we consider four pivot rules, two of which play only a helper role (the acronyms of the rules are in parentheses):

Classic rule (c): The first element s_1 of s is the pivot element.
Median-of-three rule (m3): The median of the first, middle, and last element is the pivot element, i.e., $\text{median}(s_1, s_{\lceil n/2 \rceil}, s_n)$.
Maximum-of-two rule (max2): The maximum of the first and the last element becomes the pivot element, i.e., $\max(s_1, s_n)$.
Minimum-of-two rule (min2): The minimum of the first and the last element becomes the pivot element, i.e., $\min(s_1, s_n)$.

The first pivot rule is the easiest-to-analyze and easiest-to-implement pivot rule for quicksort and Hoare's find. Its major drawback is that it yields poor running-times of quicksort and Hoare's find for nearly sorted sequences. The advantages of the median-of-three rule has already been discussed above. The last two pivot rules are only used as tools for analyzing the median-of-three rule.

Quicksort, Hoare's Find, Left-to-right Maxima. Let s be a sequence of length n consisting of pairwise distinct numbers. Let p be the pivot element of s according to some rule. For the following definitions, let $L = \{i \in \{1, \ldots, n\} \mid s_i < p\}$ be the set of positions of elements smaller than the pivot, and let $R = \{i \in \{1, \ldots, n\} \mid s_i > p\}$ be the set of positions of elements greater than the pivot.

Quicksort is the following sorting algorithm: Given s, we construct s_L and s_R by comparing all elements to the pivot p. Then we sort s_L and s_R recursively to obtain s'_L and s'_R, respectively. Finally, we output $s' = (s'_L, p, s'_R)$. The number $\text{sort}(s)$ of comparisons needed to sort s is thus $\text{sort}(s) = (n-1) + \text{sort}(s_L) + \text{sort}(s_R)$ if s has a length of $n \geq 1$, and $\text{sort}(s) = 0$ when s is the empty sequence. We do not count the number of comparisons needed to find the pivot element. Since this number is $O(1)$ per recursive call for the pivot rules considered here, this does not change the asymptotics.

Hoare's find aims at finding the k-th smallest element of s. Let $\ell = |s_L|$. If $\ell = k-1$, then p is the k-th smallest element. If $\ell \geq k$, then we search for the k-th smallest element of s_L. If $\ell < k-1$, then we search for the $(k-\ell)$-th smallest element of s_R. Let $\text{find}(s, k)$ denote the number of comparisons needed to find the k-th smallest element of s, and let $\text{find}(s) = \max_{k \in [n]} \text{find}(s, k)$.

The number of *scan maxima* of s is the number of maxima seen when scanning s according to some pivot rule: let $\text{scan}(s) = 1 + \text{scan}(s_R)$, and let $\text{scan}(s) = 0$ when s is the empty sequence. If we use the classic pivot rule, the number of

scan maxima is just the number of *left-to-right maxima*, i.e., the number of new maxima that we see if we scan s from left to right. The number of scan maxima is a useful tool for analyzing quicksort and Hoare's find, and has applications, e.g., in motion complexity [3].

We write c-scan(s), m3-scan(s), max2-scan(s), and min2-scan(s) to denote the number of scan maxima according to the classic, median-of-three, maximum, or minimum pivot rule, respectively. Similar notation is used for quicksort and Hoare's find.

Perturbation Model: Additive noise. The first perturbation model that we consider is *additive noise*. Let $d > 0$. Given a sequence $s \in [0,1]^n$, i.e., the numbers $s_1, \dots, s_n$ lie in the interval $[0,1]$, we obtain the perturbed sequence $\overline{s} = (\overline{s}_1, \dots, \overline{s}_n)$ by drawing $\nu_1, \dots, \nu_n$ uniformly and independently from the interval $[0,d]$ and setting $\overline{s}_i = s_i + \nu_i$. Note that $d = d(n)$ may be a function of the number n of elements, although this will not always be mentioned explicitly.

We denote by $\text{scan}_d(s)$, $\text{sort}_d(s)$ and $\text{find}_d(s)$ the (random) number of scan maxima, quicksort comparisons, and comparisons of Hoare's find of $\overline{s}$, preceded by the acronym of the pivot rule used.

Our goal is to prove bounds for the smoothed number of comparisons that Hoare's find needs, i.e., $\max_{s\in[0,1]^n} \mathbb{E}\big(\text{c-find}_d(s)\big)$, as well as for Hoare's find and quicksort with the median-of-three pivot rule, i.e., $\max_{s\in[0,1]^n} \mathbb{E}\big(\text{m3-find}_d(s)\big)$ and $\max_{s\in[0,1]^n} \mathbb{E}\big(\text{m3-sort}_d(s)\big)$. The max reflects that the sequence s is chosen by an adversary.

If $d < 1/n$, the sequence s can be chosen such that the order of the elements is unaffected by the perturbation. Thus, in the following, we assume $d \geq 1/n$. If d is large, the noise will swamp out the original instance, and the order of the elements of $\overline{s}$ will basically depend only on the noise rather than the original instance. For intermediate d, we interpolate between the two extremes.

The choice of the intervals for the adversarial part and the noise is arbitrary. All that matters is the ratio of the sizes of the intervals: For $a < b$, we have $\max_{s\in[a,b]^n} \mathbb{E}\big(\text{find}_{d\cdot(b-a)}(s)\big) = \max_{s\in[0,1]^n} \mathbb{E}\big(\text{find}_d(s)\big)$. In other words, we can scale (and also shift) the intervals, and the results depend only on the ratio of the interval sizes and the number of elements. The same holds for all other measures that we consider. We will exploit this in the analysis of Hoare's find.

Perturbation Model: Partial Permutations. The second perturbation model that we consider is *partial permutations*, introduced by Banderier, Beier, and Mehlhorn [1]. Here, the elements are left unchanged. Instead, we permute a random subsets of the elements.

Without loss of generality, we can assume that s is a permutation of a set of n numbers, say, $\{1, \dots, n\}$. The perturbation parameter is $p \in [0,1]$. Any element s_i (or, equivalently, any position i) is marked independently of the others with a probability of p. After that, all marked positions are randomly permuted: Let M be the set of positions that are marked, and let $\pi : M \to M$ be a permutation drawn uniformly at random. Then $\overline{s}_i = s_{\pi(i)}$ if $i \in M$ and $\overline{s}_i = s_i$ otherwise. If $p = 0$, no element is marked, and we obtain worst-case bounds. If $p = 1$,

all elements are marked, and $\overline{s}$ is a uniformly drawn random permutation. We denote by pp-find$_p(s)$ the random number of comparisons that Hoare's find needs with the classic pivot rule when s is perturbed.

1.2 Known Results

Additive noise is perhaps the most basic and natural perturbation model for smoothed analysis. In particular, Spielman and Teng added random numbers to the entries of the adversarial matrix in their smoothed analysis of the simplex algorithm [17]. Damerow et al. [3] analyzed the smoothed number of left-to-right maxima of a sequence under additive noise. They obtained upper bounds of $O\big(\sqrt{\frac{n}{d}\log n} + \log n\big)$ for a variety of distributions and a lower bound of $\Omega(\sqrt{n}+\log n)$. Manthey and Tantau tightened their bounds for uniform noise to $O\big(\sqrt{n/d}+\log n\big)$. Furthermore, they proved that the same bounds hold for the smoothed tree height. Finally, they showed that quicksort needs $O\big(\frac{n}{d+1}\cdot\sqrt{\frac{n}{d}}\big)$ comparisons in expectation, and this bound is also tight [12].

Banderier et al. [1] introduced partial permutations as a perturbation model for ordering problems like left-to-right maxima or quicksort. They proved that a sequence of n numbers has, after partial permutation, an expected number of $O\big(\sqrt{\frac{n}{p}\log n}\big)$ left-to-right maxima, and they proved a lower bound of $\Omega\big(\sqrt{n/p}\big)$ for $p \le \frac{1}{2}$. This has later been tightened by Manthey and Reischuk [11] to $\Theta\big((1-p)\cdot\sqrt{n/p}\big)$. They transferred this to the height of binary search trees, for which they obtained the same bounds. Banderier et al. [1] also analyzed quicksort, for which they proved an upper bound of $O\big(\frac{n}{p}\log n\big)$.

1.3 New Results

We give a smoothed analysis of Hoare's find under additive noise. We consider both finding an arbitrary element and finding the median. First, we analyze finding arbitrary elements, i.e., the adversary specifies k, and we have to find the k-th smallest element (Section 2). For this variant, we prove tight bounds of $\Theta\big(\frac{n}{d+1}\sqrt{n/d}+n\big)$ for the expected number of comparisons. This means that already for very small $d \in \omega(1/n)$, the smoothed number of comparisons is reduced compared to the worst case. If d is a small constant, i.e., the noise is a small percentage of the data values like 1%, then $O(n^{3/2})$ comparisons suffice.

If the adversary is to choose k, our lower bound suggests that we will have either $k = 1$ or $k = n$. The main task of Hoare's find, however, is to find medians. Thus, second, we give a separate analysis of how much comparisons are needed to find the median (Section 2.2). It turns out that under additive noise, finding medians is arguably easier than finding maximums or minimums: For $d \le 1/2$, we have the same bounds as above. For $d \in (\frac{1}{2}, 2)$, we prove a lower bound of $\Omega\big(n^{3/2}\cdot(1-\sqrt{d/2})\big)$, which again matches the upper bound of Section 2 that of course still applies. For $d > 2$, we prove that a linear number of comparisons suffices, which is considerably less than the $\Omega\big((n/d)^{3/2}\big)$ general lower bound of Section 2. For the special value $d = 2$, we prove a tight bound of $\Theta(n\log n)$.

algorithm	$d \le 1/2$	$d \in (1/2, 2)$	$d = 2$	$d > 2$
quicksort (c)	$\Theta\big(n\sqrt{n/d}\big)$	$\Theta\big(n^{3/2}\big)$	$\Theta\big(n^{3/2}\big)$	$\Theta\big((n/d)^{3/2}\big)$
quicksort (m3)	$\Omega\big(n\sqrt{n/d}\big)$	$\Omega\big(n^{3/2}\big)$	$\Omega\big(n^{3/2}\big)$	$\Omega\big((n/d)^{3/2}\big)$
Hoare's find (median, c)	$\Theta\big(n\sqrt{n/d}\big)$	$\Omega\big(n^{3/2}(1-\sqrt{d/2})\big)$	$\Theta(n \log n)$	$O\big(\frac{d}{d-2} \cdot n\big)$
Hoare's find (general, c)	$\Theta\big(n\sqrt{n/d}\big)$	$\Theta\big(n^{3/2}\big)$	$\Theta\big(n^{3/2}\big)$	$\Theta\big((n/d)^{3/2}\big)$
Hoare's find (general, m3)	$\Theta\big(n\sqrt{n/d}\big)$	$\Theta\big(n^{3/2}\big)$	$\Theta\big(n^{3/2}\big)$	$\Theta\big((n/d)^{3/2}\big)$
scan maxima (c)	$\Theta\big(\sqrt{n/d}\big)$	$\Theta\big(\sqrt{n}\big)$	$\Theta\big(\sqrt{n}\big)$	$\Theta\big(\sqrt{n/d}\big)$
scan maxima (m3)	$\Theta\big(\sqrt{n/d}\big)$	$\Theta\big(\sqrt{n}\big)$	$\Theta\big(\sqrt{n}\big)$	$\Theta\big(\sqrt{n/d}\big)$

Table 1. Overview of bounds for additive noise. The bounds for quicksort and scan maxima with classic pivot rule are by Manthey and Tantau [12]. The upper bounds for Hoare's find in general apply also to Hoare's find for finding the median. Note that, even for large d, the precise bounds for quicksort, Hoare's find, and scan maxima never drop below $\Omega(n \log n)$, $\Omega(n)$, and $\Omega(\log n)$, respectively.

algorithm	bound
quicksort	$O\big((n/p) \log n\big)$
Hoare's find	$\Omega\big((1-p)(n/p) \log n\big)$
scan maxima	$\Theta\big((1-p)\sqrt{n/p}\big)$
binary search trees	$\Theta\big((1-p)\sqrt{n/p}\big)$

Table 2. Overview of bounds for partial permutations. All results are for the classic pivot rule. The results about quicksort, scan maxima, and binary search trees are by Banderier et al. [1] and Manthey and Reischuk [11]. The upper bound for quicksort also holds for Hoare's find, while the lower bound for Hoare's find also applies to quicksort.

After that, we aim at analyzing different pivot rules, namely the median-of-three rule. As a tool, we analyze the number of scan maxima under the maximum-of-two, minimum-of-two, and median-of-three rule (Section 3). We essentially show that the same bounds as for the classic rule carry over to these rules. Then we apply these findings to quicksort and Hoare's find (Section 4). Again, we prove a lower bound that matches the lower bound for the classic rule. Thus, the median-of-three does not seem to help much under additive noise. The results concerning additive noise are summarized in Table 1.

Finally, and to contrast our findings for additive noise, we analyze Hoare's find under partial permutations (Section 5). We prove that there exists a sequence on which Hoare's find needs an expected number of $\Omega\big((1-p) \cdot \frac{n}{p} \cdot \log n\big)$ comparisons. Since this matches the upper bound for quicksort [1] up to a factor of $O(1-p)$, this lower bound is essentially tight. For completeness, Table 2 gives an overview of the results for partial permutations.

Due to lack of space, proofs are omitted. For complete proofs, we refer to the full version of this paper [5].

2 Smoothed Analysis of Hoare's Find

2.1 General Bounds

In this section, we state tight bounds for the smoothed number of comparisons that Hoare's find needs using the classic pivot rule.

Theorem 1. *For $d \geq 1/n$, we have*

$$\max_{s\in[0,1]^n} \mathbb{E}\big(\text{c-find}_d(s)\big) \in \Theta\big(\tfrac{n}{d+1}\sqrt{n/d} + n\big).$$

Since $\text{find}(s) \leq \text{sort}(s)$ for any s, we already have an upper bound for the smoothed number of comparisons that quicksort needs [12]. This bound is $O\big(\frac{n}{d+1} \cdot \sqrt{n/d} + n\log n\big)$, which matches the bound of Theorem 1 for $d \in O\big(n^{1/3} \cdot \log^{-2/3} n\big)$. Thus for the proof of the theorem, $d \in \Omega\big(n^{1/3} \cdot \log^{-2/3} n\big)$ remains to be analyzed. The proof of the lower bound is similar to Manthey and Tantau's lower bound proof for quicksort [12].

2.2 Finding the Median

In this section, we provide tight bounds for the special case of finding the median of a sequence using Hoare's find. Somewhat surprisingly, finding the median seems to be easier in the sense that fewer comparisons suffice.

Theorem 2. *Depending on d, we have the following bounds for*

$$\max_{s\in[0,1]^n} \mathbb{E}\big(\text{c-find}_d(s, \lceil n/2\rceil)\big) :$$

For $d \leq \frac{1}{2}$, we have $\Theta\big(n \cdot \sqrt{n/d}\big)$. For $\frac{1}{2} < d < 2$, we have $\Omega\big(\big(1-\sqrt{d/2}\big) \cdot n^{3/2}\big)$ and $O\big(n^{3/2}\big)$. For $d = 2$, we have $\Theta\big(n \cdot \log n\big)$. For $d > 2$, we have $O\big(\frac{d}{d-2} \cdot n\big)$.

The upper bound of $O(n \cdot \sqrt{n/d})$ for $d < 2$ follows from our general upper bound (Theorem 1). For $d \leq \frac{1}{2}$, our lower bound construction for the general bounds also works: The median is among the last $n/2$ elements, which are the big ones. (We might want to have $\lceil n/2\rceil$ or $n/2+1$ large elements to assure this.) The rest of the proof remains the same. For $d > 2$, Theorem 2 states a linear bound, which is asymptotically equal to the average-case bound. Thus, we do not need a lower bound in this case.

First, we state a crucial fact about the value of the median: Intuitively, the median should be around $d/2$ if all elements of s are 0, and it should be around $1 + d/2$ if all elements of s are 1. We make this precise: Independent of the input sequence, the median will be neither much smaller than $d/2$ nor much greater than $1 + d/2$ with high probability.

Lemma 1. *Let $s \in [0,1]^n$, and let $d > 0$. Let $\xi = c\sqrt{\log n/n}$. Let m be the median of $\overline{s}$. Then $\mathbb{P}\big(m \notin \big[d/2 - \xi, 1 + d/2+\big]\big) \leq 4 \cdot \exp\big(-2c^2 \log n/d^2\big)$.*

The idea to prove the upper bound for $d > 2$ is as follows: Since $d > 2$ and according to Lemma 1 above, it is likely that any element can assume a value greater or smaller than the median. Thus, after we have seen a few number of pivots (for which we "pay" with $O(\frac{d}{d-2}n)$ comparisons), all elements that are not already cut off are within some small interval around the median. These elements are uniformly distributed. Thus, the linear average-case bound applies.

Lemma 2. *Let $d > 2$ be bounded away from 2. Then*

$$\max_{s\in[0,1]^n} \mathbb{E}\big(\text{c-find}_d(s, \lceil n/2\rceil)\big) \in O\big(\tfrac{d}{d-2} \cdot n\big).$$

3 Scan Maxima with Median-of-three Rule

The results in this section serve as a basis for the analysis of both quicksort and Hoare's find with the median-of-three rule. In order to analyze the number of scan maxima with the median-of-three rule, we analyze this number with the maximum and minimum of two rules. This is justified since, for every sequence s, we have $\text{max2-scan}(s) \le \text{m3-scan}(s) \le \text{min2-scan}(s)$.

The reason for considering max2-scan and min2-scan is that it is hard to keep track where the middle element with median-of-three rule lies: Depending on which element actually becomes the pivot and which elements are greater than the pivot, the new middle position can be on the far left or on the far right of the previous middle. From $\mathbb{E}\big(\text{max2-scan}_d(s)\big) \in \Omega\left(\sqrt{\frac{n}{d}} + \log n\right)$ and $\mathbb{E}\big(\text{min2-scan}_d(s)\big) \in O\left(\sqrt{\frac{n}{d}} + \log n\right)$, we get our bounds for m3-scan.

Theorem 3. *For every $d \ge 1/n$, we have*

$$\max_{s\in[0,1]^n} \mathbb{E}\big(\text{m3-scan}_d(s)\big) \in \Theta\big(\sqrt{n/d} + \log n\big).$$

4 Quicksort and Hoare's Find with Median-of-three Rule

Now we use our results about scan maxima from the previous section to provide lower bounds for the number of comparisons that quicksort and Hoare's find need using the median-of-three pivot rule. We only give lower bounds here since they match already the upper bounds for the classic pivot rule. We strongly believe that the median-of-three rule does not yield worse bounds than the classic rule and, hence, that our bounds are tight. The goal of this section is to establish a lower bound for Hoare's find, which then carries over to quicksort.

Theorem 4. *For $d \ge 1/n$, we have*

$$\max_{s\in[0,1]^n} \mathbb{E}\big(\text{m3-find}_d(s)\big) \in \Omega\big(\tfrac{n}{d+1}\sqrt{n/d} + n\big) \text{ and}$$
$$\max_{s\in[0,1]^n} \mathbb{E}\big(\text{m3-sort}_d(s)\big) \in \Theta\big(\tfrac{n}{d+1}\sqrt{n/d} + n\log n\big).$$

5 Hoare's Find Under Partial Permutations

To complement our findings about Hoare's find, we analyze the number of comparisons subject to partial permutations. For this model, we already have an upper bound of $O(\frac{n}{p}\log n)$, since that bound has been proved for quicksort by Banderier et al. [1]. We show that this is asymptotically tight (up to factors depending only on p) by proving that Hoare's find needs a smoothed number of $\Omega\big((1-p)\frac{n}{p}\cdot\log n\big)$ comparisons. The main idea behind the proof of the following theorem is as follows: We aim at finding the median. The first few elements are close to and smaller than the median. Thus, it is unlikely that one of them is permuted further to the left. This implies that all unmarked of the first few elements become pivot elements. Then they have to be compared to many of the $\Omega(n)$ elements larger than the median, which yields our lower bound.

Theorem 5. *Let $p \in (0,1)$ be a constant. There exist sequences s of length n such that under partial permutations we have*

$$\mathbb{E}\big(\text{pp-find}_p(s)\big) \in \Omega\big((1-p)\cdot\tfrac{n}{p}\cdot\log n\big).$$

For completeness, to conclude this section, and as a contrast to Sections 2 and 2.2, let us remark that for partial permutations, finding the maximum using Hoare's find seems to be easier than finding the median: The lower bound constructed above for finding the median requires that there are elements on either side of the element we aim for. If we aim at finding the maximum, all elements are on the same side of the target element. In fact, we believe that for finding the maximum, an expected number of $O(f(p)\cdot n)$ for some function f depending on p suffices.

6 Concluding Remarks

We have shown tight bounds for the smoothed number of comparisons for Hoare's find under additive noise and under partial permutations. Somewhat surprisingly, it turned out that, under additive noise, Hoare's find needs (asymptotically) more comparisons for finding the maximum than for finding the median. Furthermore, we analyzed quicksort and Hoare's find with the median-of-three pivot rule, and we proved that median-of-three does not yield an asymptotically better bound. Let us remark that also the lower bounds for left-to-right maxima as well as for the height of binary search trees [11] can be transferred to median-of-three. The bounds remain equal.

A natural question regarding additive noise is what happens when the noise is drawn according to an arbitrary distribution rather than the uniform distribution. Some first results on this for left-to-right maxima were obtained by Damerow et al. [3]. We conjecture the following: If the adversary is allowed to specify a density function bounded by ϕ, then all upper bounds still hold with $d = 1/\phi$ (the maximum density of the uniform distribution on $[0,d]$ is $1/d$). However, as Manthey and Tantau point out [12], a direct transfer of the results for uniform noise to arbitrary noise might be difficult.

References

1. Cyril Banderier, René Beier, and Kurt Mehlhorn. Smoothed analysis of three combinatorial problems. In *Proc. of the 28th Int. Symp. on Mathematical Foundations of Computer Science (MFCS)*, vol. 2747 of *Lecture Notes in Comput. Sci.*, pp. 198–207. Springer, 2003.
2. Daniel Cederman and Philippas Tsigas. A practical quicksort algorithm for graphics processors. In *Proc. of the 16th Ann. European Symp. on Algorithms (ESA)*, vol. 5193 of *Lecture Notes in Comput. Sci.*, pp. 246–258. Springer, 2008.
3. Valentina Damerow, Friedhelm Meyer auf der Heide, Harald Räcke, Christian Scheideler, and Christian Sohler. Smoothed motion complexity. In *Proc. of the 11th Ann. European Symp. on Algorithms (ESA)*, vol. 2832 of *Lecture Notes in Comput. Sci.*, pp. 161–171. Springer, 2003.
4. Hannu Erkiö. The worst case permutation for median-of-three quicksort. *The Computer Journal*, 27(3):276–277, 1984.
5. Mahmoud Fouz, Manfred Kufleitner, Bodo Manthey, and Nima Zeini Jahromi. On smoothed analysis of quicksort and Hoare's find. Computing Research Repository, arXiv:0904.3898 [cs.DS], 2009.
6. C. A. R. Hoare. Algorithm 64: Quicksort. *Comm. ACM*, 4(7):322, 1961.
7. C. A. R. Hoare. Algorithm 65: Find. *Comm. ACM*, 4(7):321–322, 1961.
8. Peter Kirschenhofer and Helmut Prodinger. Comparisons in Hoare's find algorithm. *Combin. Probab. Comput.*, 7(1):111–120, 1998.
9. Peter Kirschenhofer, Helmut Prodinger, and Conrado Martinez. Analysis of Hoare's find algorithm with median-of-three partition. *Random Structures Algorithms*, 10(1-2):143–156, 1997.
10. Donald E. Knuth. *Sorting and Searching*, vol. 3 of *The Art of Computer Programming*. Addison-Wesley, 2nd edition, 1998.
11. Bodo Manthey and Rüdiger Reischuk. Smoothed analysis of binary search trees. *Theoret. Comput. Sci.*, 378(3):292–315, 2007.
12. Bodo Manthey and Till Tantau. Smoothed analysis of binary search trees and quicksort under additive noise. In *Proc. of the 33rd Int. Symp. on Mathematical Foundations of Computer Science (MFCS)*, vol. 5162 of *Lecture Notes in Comput. Sci.*, pp. 467–478. Springer, 2008.
13. Douglas C. Schmidt. `qsort.c`. C standard library `stdlib` within `glibc 2.7`, available at `http://ftp.gnu.org/gnu/glibc/`, 2007.
14. Robert Sedgewick. The analysis of quicksort programs. *Acta Inform.*, 7(4):327–355, 1977.
15. Robert Sedgewick. Implementing quicksort programs. *Comm. ACM*, 21(10):847–857, 1978.
16. Richard C. Singleton. Algorithm 347: An efficient algorithm for sorting with minimal storage. *Comm. ACM*, 12(3):185–186, 1969.
17. Daniel A. Spielman and Shang-Hua Teng. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *J. ACM*, 51(3):385–463, 2004.
18. Daniel A. Spielman and Shang-Hua Teng. Smoothed analysis of algorithms and heuristics: Progress and open questions. In *Foundations of Computational Mathematics, Santander 2005*, pp. 274–342. Cambridge University Press, 2006.