Multiparty Selection

Ke Chen
Department of Computer Science, University of Wisconsin–Milwaukee, USA
kechen@uwm.edu

Adrian Dumitrescu
Department of Computer Science, University of Wisconsin–Milwaukee, USA
dumitres@uwm.edu

Abstract

Given a sequence A of n numbers and an integer (target) parameter $1 \le i \le n$, the (*exact*) selection problem is that of finding the i-th smallest element in A. An element is said to be (i, j)-*mediocre* if it is neither among the top i nor among the bottom j elements of S. The *approximate* selection problem is that of finding an (i, j)-mediocre element for some given i, j; as such, this variant allows the algorithm to return any element in a prescribed range. In the first part, we revisit the selection problem in the two-party model introduced by Andrew Yao (1979) and then extend our study of exact selection to the multiparty model. In the second part, we deduce some communication complexity benefits that arise in approximate selection. In particular, we present a deterministic protocol for finding an approximate median among k players.

2012 ACM Subject Classification Theory of computation

Keywords and phrases approximate selection, mediocre element, comparison algorithm, i-th order statistic, tournaments, quantiles, communication complexity

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2020.3

1 Introduction

Given a sequence A of n numbers and an integer (selection) parameter $1 \le i \le n$, the selection problem asks to find the i-th smallest element in A. If the n elements are distinct, the i-th smallest is larger than $i - 1$ elements of A and smaller than the other $n - i$ elements of A. By symmetry, the problems of determining the i-th smallest and the i-th largest are equivalent. Together with sorting, the selection problem is one of the most fundamental problems in computer science. Whereas sorting trivially solves the selection problem in $O(n \log n)$ time, Blum et al. [7] gave an $O(n)$-time algorithm for this problem.

The selection problem, and computing the median in particular, are in close relation with the problem of finding the quantiles of a set. The h-th *quantiles* of an n-element set are the $h - 1$ order statistics that divide the sorted set in h equal-sized groups (to within 1); see, e.g., [10, p. 223]. The h-th quantiles of a set can be computed by a recursive algorithm running in $O(n \log h)$ time.

The selection problem, determining the median in particular, has been also considered from the perspective of communication complexity in the *two-party* model introduced by Andrew Yao [38]. Suppose that Alice and Bob hold subsets A and B of $[n] = \{1, 2, \ldots, n\}$, respectively, and wish to determine the median of the multiset $A \cup B$ while keeping their communication close to a minimum. Several classic protocols going back to 1980s achieve this task by exchanging $O(\log^2 n)$ bits [29, 36]. The communication complexity for this task has been subsequently reduced to $O(\log n)$ bits [9, 29, 31, 35].

Mediocre elements. Following Frances Yao [39], an element is said to be (i, j)-*mediocre* if it is neither among the top (i.e., largest) i nor among the bottom (i.e., smallest) j of a totally ordered set S of n elements. As remarked by Yao, finding a mediocre element is closely

31st International Symposium on Algorithms and Computation (ISAAC 2020).
Editors: Yixin Cao, Siu-Wing Cheng, and Minming Li; Article No. 3; pp. 3:1–3:14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

related to finding the median, in the sense that the common goal is selecting an element that is not too close to either extreme. In particular, (i, j)-mediocre elements where $i = \lfloor \frac{n-1}{2} \rfloor$, $j = \lfloor \frac{n}{2} \rfloor$ (and symmetrically exchanged), are medians of S. Previous work on *approximate selection* (in this sense) includes [5, 16].

In Section 3 we provide a protocol to find a mediocre element near the median among k players with communication complexity $O(k \log n)$. To our best knowledge, this is the first result on the mediocre element finding problem, in terms of communication complexity. In Section 4 we outline a scenario in which computing a mediocre element near the median in the two-party model can be accomplished with communication complexity $O(1)$—which is very attractive.

Background and related problems. Due to its primary importance, the selection problem has been studied extensively; see for instance [2, 6, 11, 13, 14, 15, 19, 20, 21, 22, 23, 24, 25, 26, 34, 37, 40]. A comprehensive review of early developments in selection is provided by Knuth [28]. The reader is also referred to dedicated book chapters on selection, such as those in [1, 4, 10, 12, 27] and the more recent articles [8, 17], including experimental work [3].

In many applications (e.g., sorting), it is not important to find an exact median, or any other precise order statistic, for that matter, and an approximate median suffices [18]. For instance, quick-sort type algorithms aim at finding a balanced partition rather quickly; see e.g., [22, 32, 33].

Studying the multiparty communication complexity of exact and approximate selection is relevant in the context of distributed computing [9, 31, 36, 38].

Our results. Our main results are summarized in the three theorems stated below. We first study the communication complexity of finding the median in the multiparty setting. In this model we assume that every message by one of the players is seen by all the players (i.e., it is a broadcast); as in [29, p. 83].

▶ **Theorem 1.** *For $i = 1, \ldots, k$, let player i hold a sequence (i.e., a multiset) A_i whose support is a subset of $[n]$ and $|A_i| = O(poly(n))$. There is a deterministic protocol for finding the median of $\uplus_{i=1}^{k} A_i$ (i.e., their multiset sum) with $O(k \log^2 n)$ communication complexity.*

We then study the communication complexity of finding an approximate median in the multiparty setting (under slightly stronger assumptions on the input sets).

▶ **Theorem 2.** *Let $\alpha = p/q$, where $p, q \in \mathbb{N}$, $p < q/2$, q is fixed and $0 < c \leq 1$ be a positive constant. For $i = 1, \ldots, k$, let player i hold a set $A_i \subset [n]$ that is disjoint from any other player's set. Assume that $t = |\cup_{i=1}^{k} A_i| \geq cn$. Put $\ell = \lceil \log \frac{2q}{c} \rceil$. Then an $(\alpha t, \alpha t)$-mediocre element of $\cup_{i=1}^{k} A_i$ can be found with $O(\ell \cdot k \log n) = O(k \log n)$ communication complexity.*

In particular, a $(t/3, t/3)$-mediocre element, or a $(0.49\,t, 0.49\,t)$-mediocre element, among k players can be determined with $O(k \log n)$ communication complexity.

In the final part of our paper, somewhat surprisingly, we show that (under suitable additional assumptions and a somewhat relaxed requirement) the communication complexity of finding a mediocre element in the vicinity of the median is bounded from above by a constant and is therefore independent of n.

▶ **Theorem 3.** *Let $\alpha = p/q$, where $p, q \in \mathbb{N}$, $p < q/2$, q is fixed and $0 < c \leq 1$ be a positive constant. Let Alice and Bob hold disjoint sets A and B of elements from $[n]$, where $s = |A| \leq |B| = m$. Let $t = s + m$ denote the total number of elements in $A \cup B$, where $t \geq cn$. Assume that t, c, and α are known to both players. Put $h = \lceil \frac{2q}{q-2p} \rceil$ and $\ell = \lceil \log \frac{12h}{c} \rceil$.*

Then an $(\alpha t, \alpha t)$-mediocre element can be found (by at least one player) with $O(\ell \log h) = O(1)$ communication complexity. If both players return, each element returned is $(\alpha t, \alpha t)$-mediocre; the elements found by the players need not be the same.

In particular, a $(t/3, t/3)$-mediocre element, or a $(0.49\,t, 0.49\,t)$-mediocre element, between 2 players can be determined (by at least one player) with $O(1)$ communication complexity. A simple example that falls under the scenario in Theorem 3 is one where A consists of distinct odd numbers and B consists of distinct even numbers. It is worth noting that since $m/2t \geq 1/4$, if $\alpha < 1/4$, the median of B is guaranteed to be an $(\alpha t, \alpha t)$-mediocre element of $A \cup B$. In this case, no communication is needed.

Preliminaries. A simple but effective procedure reduces the selection problem for finding the i-th smallest element out of n to one for finding the median in a slightly larger sequence. The target is the i-th smallest element in an input sequence A of size n. Assume first that $i < n/2$; in this case pad the input A with $n - 2i$ elements that are less than or equal to the minimum in the input sequence; call A' resulting sequence. Note that $|A'| = n + (n - 2i) = 2(n - i)$. It suffices to observe that the median of A' is the i-th smallest element in A: indeed, $n - 2i + i = n - i$, as required. The case $i > n/2$ is symmetric; in this case pad the input A with $2i - n$ elements that are larger than or equal to the maximum in the input sequence; call A' resulting sequence. Note that $|A'| = n + (2i - n) = 2i$. Observe that the median of A' is the i-th smallest element in A, as required. We therefore restrict our attention to the median selection problem.

Notation. Without affecting the results, the floor and ceiling functions are omitted in some instances where they are not essential. For example, we frequently write the αn-th element instead of the more precise $\lfloor \alpha n \rfloor$-th element. Unless specified otherwise, all logarithms are in base 2.

For an s-bit number x and a positive integer ℓ, where $s \geq \ell$, $\texttt{prefix}_\ell(x)$ denotes the *ℓ-bit binary prefix* of x, i.e., the number formed by the first (i.e., most significant) ℓ bits of x.

If x belongs to a sorted list and is not the minimum, $\texttt{pred}(x)$ denotes its predecessor. If x belongs to a sorted list and is not the maximum, $\texttt{succ}(x)$ denotes its successor.

2 Exact selection

In this section we prove Theorem 1. First, we set up the problem in the context of two-party communication complexity; we start with some background. In this section, each player's input is allowed to contain duplicates. Following the literature, we refer to these (potential) multisets as sets, and the union operation should be understood as multiset sum [29, Example 1.6, p. 6]. (An equivalent formulation is *merging of sequences*.)

2.1 Two players

Alice and Bob hold multisets A and B whose supports are subsets of $[n] = \{1, 2, \ldots, n\}$, respectively. It is assumed that $|A|, |B| = O\,(\text{poly}(n))$. (In a standard setup [29, Example 1.6, p. 6], A and B are subsets of $[n]$; here we extend this setup for potentially larger multisets.) The median of the multiset $A \cup B$ is denoted by $\xi = \texttt{Med}(A, B)$; as usual, the median of X is the $\lceil (|X|/2) \rceil$-th smallest element of X.

There is a simple binary-search type protocol due to M. Karchmer that takes $O(\log^2 n)$ bits of communication; see [29, Example 1.6, p. 6]. At each round Alice and Bob have an interval $[i, j]$, $i, j \in \mathbb{N}$, that contains the median. They halve the interval (repeatedly) by deciding whether the median is less than, equal to, or larger than $m = (i + j)/2$. This is done by Alice sending to Bob the number of elements in A that are less than m, equal to m, and larger than m, using $O(\log n)$ bits. Bob can now determine whether the median is less than, equal to, or larger than m, and sends this information to Alice using $O(1)$ bits. The protocol has $O(\log n)$ rounds, each requiring $O(\log n)$ bits of communication, so the overall communication complexity is $O(\log^2 n)$.

An alternative binary-search type protocol that takes $O(\log^2 n)$ bits of communication, also due to Karchmer [29, p. 168], works as follows. Assume, without loss of generality that $|A| = |B|$ and that the common size is a power of 2: this can be achieved by exchanging the sizes of their inputs ($O(\log n)$ bits) and padding them with the appropriate number of the minimal element (1) and the maximal element (n). The protocol works in rounds. During the protocol, Alice maintains a set $A' \subset A$ of elements that may still be the median (initially $A' = A$) and Bob maintains a set $B' \subset B$ of elements that may still be the median (initially $B' = B$). At each round, Alice sends Bob the value a, which is the median of A', and Bob sends Alice the value b, which is the median of B'. At this point we have $\min(a, b) \le \xi \le \max(a, b)$. If $a < b$, then Alice discards the lower half of A' (note that a is part of it) and Bob discards the upper half of B'. If $b < a$, then Bob discards the lower half of B' (note that b is part of it) and Alice discards the upper half of A'. In either case, this operation maintains the median of $A' \cup B'$ as the desired median of $A \cup B$. It should be noted that the size of $A' \cup B'$ is reduced (exactly) by a factor of 2. If $a = b$, this value is the median, and if $|A'| = |B'| = 1$, then the smaller number is the median. The protocol has $O(\log n)$ rounds, each requiring $O(\log n)$ bits of communication, and so the communication complexity is $O(\log^2 n)$.

The communication complexity of finding the median can be further reduced. A subtle refinement of the above protocol, due to Karchmer [29, Example 1.7, p. 6 and p. 168], and revised by Gasarch [30], works with $O(\log n)$ communication complexity: its key idea is to make comparisons in a bit-by-bit manner, but this requires careful bookkeeping of the progress and here we omit the technical details.

We next describe a different (folklore) protocol, running with $O(\log n)$ communication complexity, that we find simpler and subsequently refine for computing a mediocre element. The protocol implements a binary-search strategy and works in rounds. Alice maintains a set $A' \subset A$ of elements that may still be the median (initially $A' = A$) and Bob maintains a set $B' \subset B$ of elements that may still be the median (initially $B' = B$). Alice and Bob compute the medians of their current inputs (a and b, respectively). At this point we have $\min(a, b) \le \xi \le \max(a, b)$. Alice and Bob aim to determine the order relation between a and b in order to halve their input in an appropriate manner.

The protocol avoids sending these $\log n$-bit numbers at each round by avoiding making a direct comparison between a and b. The players have an interval $[i, j]$, $i, j \in \mathbb{N}$, that contains the median (initially, $[i, j] = [1, n]$). The medians a and b are compared to the middle element $h = \lfloor (i+j)/2 \rfloor$, If $a = b = h$, this element is the median of $A \cup B$ and the protocol terminates. Otherwise, if a and b are split by h, i.e., $a \le h \le b$ or $b \le h \le a$, then (by transitivity of $\le$), the relation between a and b is determined, and Alice and Bob halves their input accordingly (as in the earlier $O(\log^2 n)$ protocol). Otherwise, if a and b are on the same side of h, i.e., $a, b \le h$ or $h \le a, b$. For example, in the first case, the elements in the lower half of $A' \cup B'$ are $\le h$ and the same holds for the median of $A' \cup B'$. As such, both players shrink their common interval $[i, j]$ by (roughly) half: the resulting interval is $[i, h]$ or $[h, j]$, respectively. The sets A' and B' remain unchanged. Alice and Bob communicate each of the outcomes of the above tests in $O(1)$ bits. Each halving operation for A' and B' maintains the property that $\xi = \texttt{Med}(A \cup B) = \texttt{Med}(A' \cup B')$.

Let $\ell = \lceil \log n \rceil$. Note that after $2\ell - 1$ tests, either Alice and Bob hold singleton sets (i.e., $|A'| = |B'| = 1$), or the common interval $[i, j]$ consists of a single integer $i = j$. If $|A'| = |B'| = 1$, the smaller number is the median (or either, for equality), whereas if $i = j$, this number is the median. The number of bits exchanged before the last round of the protocol is $O(\log n)$ and is $O(\log n)$ in the last round. The resulting communication complexity is $O(\log n)$.

2.2 k players

In this subsection we show the protocol that proves Theorem 1. It is worth noting that the number of players, k is independent of n. The protocol maintains the invariant: the median of $\cup_{i=1}^{k} A_i$ in one round is the same for the updated sets in the next round. It is possible that the number of sets drops from k to a lower number; the protocol remains unchanged until the value $k = 2$ is reached, when the respective players apply the protocol in Subsection 2.1; recall that padding with extra elements may be needed. If the value $k = 1$ is reached, the remaining player computes the median in his/her own set and the game ends.

Initially, each player sorts his/her input set locally. The sorted order is used by each player in the pruning process, and if such action occurs, the sorted order is locally maintained. Each set pruning discards elements at one of the two ends of the chain (either low elements below some threshold, or high elements above some threshold).

The protocol roughly halves the size of at least one of the current participating sets; more precisely, for some $X \in \{A_1, \ldots, A_k\}$, we have $|X'| \leq \lfloor |X|/2 \rfloor$ by the end of each round. Since the size of each set is initially $O(\text{poly}(n))$, the size of each of the k sets drops to 0 in at most $O(\log n)$ iterations and consequently, the number of rounds is at most $O(k \log n)$. (Padding with extra elements when $k = 2$ is reached conforms with this bound.)

Each round of the protocol works as follows. Each player (locally) finds the median of his/her current set: $x_i \in A_i$, $i = 1, \ldots, k$. The following scheme regarding medians is used: assume that there are x sets of even size and y sets of odd size in the current round, where $x + y = k$; for the x sets of even size the first $\lceil x/2 \rceil$ use the lower median and the remaining $\lfloor x/2 \rfloor$ use the upper median (in some fixed, e.g., alphabetical, order). The idea of intermixing upper and lower medians is also present in [8]. (A scheme that uses only lower medians or only upper medians fails to guarantee that the median of the union is maintained after pruning, for instance if $k = 3$ and all three sets have even size; the smallest example of this kind is $|A_1| = |A_2| = |A_3| = 2$.)

In the first round, each player posts his/her median and set size on the communication board; this involves $O(k \log n)$ bits of communication. In the remaining rounds, two players whose sets got pruned (as further explained below) need to update their median on the communication board. Depending on the parities of the sets of these two players before and after the pruning, at most one more player may need to update his/her median to maintain the balanced scheme adopted earlier which requires $\lceil x/2 \rceil$ use the lower median and the remaining $\lfloor x/2 \rfloor$ use the upper median. Therefore, in each round, the communication complexity is $O(\log n)$.

All players are now able to determine the sorted order of the k medians. For simplicity, assume that after relabeling, this order is

$$x_1 \leq x_2 \leq \ldots \leq x_k. \tag{1}$$

It is convenient to refer to the players holding the minimum and maximum of these medians as Alice and Bob and to their corresponding sets as A and B: $x_A \equiv x_1$ and $x_B \equiv x_k$ (this relabeling is only done for the purpose of analysis).

Let P denote the poset made by the k chains $A_1, \ldots, A_k$, together with the relations in (1). Write $a = |A|$, $b = |B|$, and $t = \sum_{i=1}^{k} |A_i|$. The player holding the smaller set between Alice and Bob is in charge of the pruning operation in the current round: the same number of elements is discarded by Alice and Bob as specified below. Refer to Fig 1.

If $\min(a, b) = a$, Alice discards $\lceil a/2 \rceil$ elements in A (all $x \leq x_A$ when a is odd or x_A is the lower median, or all $x < x_A$ when x_A is the upper median), and Bob discards the highest $\lceil a/2 \rceil$ elements in B. Such operation is *charged* to Alice. Otherwise, if $\min(a, b) = b$, Bob discards $\lceil b/2 \rceil$ elements in B (all $x \geq x_B$ when b is odd or x_B is the upper median, or all $x > x_B$ when x_B is the lower median), and Alice discards the lowest $\lceil b/2 \rceil$ elements in A. Such operation is *charged* to Bob. It is worth noting that this scheme is feasible: i.e., if the indicated player discards the specified number of elements, the other player can also discard the same number of elements. Then the protocol continues with the next round. Each player keeps track of the players that are still in the game and their set cardinalities, as these can be deduced from the actions of the algorithm.

It remains to show that the same number of elements is discarded from each side of the median in each round. Let u be the number of elements in P that are above the highest discarded element of A, and v be the number of elements in P that are below the lowest discarded element of B. By slightly abusing notation, let k denote the number of players in the current round of the protocol (which may differ from the initial number). Specifically we prove the following.

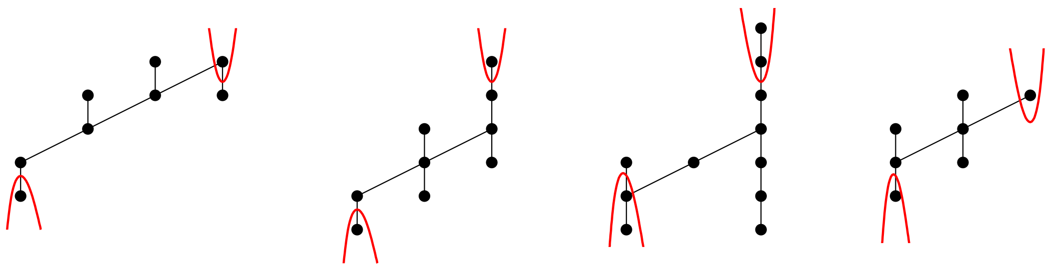

Figure 1 Pruning the poset P in the protocol for finding the median; Alice is the leftmost player and Bob is the rightmost player. (i) $k = 4$, $t = 8$, $u = 6$, $v = 5$; operation is charged to Alice. (ii) $k = 3$, $t = 9$, $u = 6$, $v = 7$; operation is charged to Alice. (iii) $t = 11$, $u = 6$, $v = 8$; operation is charged to Alice. (iv) $t = 7$, $u = 5$, $v = 4$; operation is charged to Bob.

▶ **Lemma 4.** *Consider a round of the protocol and assume that $k \geq 3$ and $t = \sum_{i=1}^{k} |A_i|$. The following inequalities for u and v hold: $u \geq \lceil \frac{t+1}{2} \rceil$ and $v \geq \lceil \frac{t}{2} \rceil$.*

Proof. For u, we start by including $|A_i|/2$ corresponding to the upper half elements in the set A_i, for $i = 1, \ldots, k$; this contributes $t/2$ to the sum. In addition we add $1/2$ for each set of odd size, thus $y/2$ over all odd sets. Then we add 1 for each set of even size that uses the lower median, thus $\lceil x/2 \rceil$ over all even sets. This procedure overcounts by 1 if the median x_A is the highest discarded element of A. Therefore, we have

$$u \geq \frac{t}{2} + \frac{y}{2} + \left\lceil \frac{x}{2} \right\rceil - 1 \geq \frac{t}{2} + \frac{y}{2} + \frac{x}{2} - 1 = \frac{t + x + y - 2}{2} = \frac{t + k - 2}{2} \geq \frac{t + 1}{2}.$$

Similarly, for v, we start by including $|A_i|/2$ corresponding to the lower half elements in the set A_i, for $i = 1, \ldots, k$; this contributes $t/2$ to the sum. In addition we add $1/2$ for each set of odd size, thus $y/2$ over all odd sets. Then we add 1 for each set of even size that uses the upper median, thus $\lfloor x/2 \rfloor$ over all even sets. This procedure overcounts by 1 if the median x_B is the lowest discarded element of B. Therefore, we have

$$v \geq \frac{t}{2} + \frac{y}{2} + \left\lfloor \frac{x}{2} \right\rfloor - 1 \geq \frac{t}{2} + \frac{y}{2} + \frac{x-1}{2} - 1 = \frac{t+x+y-3}{2} = \frac{t+k-3}{2} \geq \frac{t}{2}.$$

Since both u and v are integers, we have thereby proved that $u \geq \lceil \frac{t+1}{2} \rceil$ and $v \geq \lceil \frac{t}{2} \rceil$, as required. ◀

Proof of Theorem 1. By Lemma 4, all the elements discarded from A are below the median (of the union), and all elements discarded from B are above the median. Thus in each round, the protocol preserves the median and discards the same number of elements from each side of it. This proves the invariant of the protocol. Since the protocol takes $O(k \log n)$ rounds and the communication complexity of each round is $O(\log n)$, the overall communication complexity is $O(k \log^2 n)$, as claimed. ◀

3 Approximate selection with k players

In this section we consider the problem of finding an $(\alpha t, \alpha t)$-mediocre element among k players, where $\alpha \in (0, 1/2)$ is a fixed constant. Recall that in the setting of Theorem 2, the sets A_i, $i = 1, \ldots, k$, are pairwise disjoint. But we do *not* assume that they have the same cardinality.

The protocol works in rounds. Let $a_1 = 1$ and $b_1 = n$; and note that $[a_1, b_1]$ contains the median m, i.e., the $\lceil t/2 \rceil$-th smallest element of $\cup_{i=1}^k A_i$. For round $j = 1, 2, \ldots$, the interval $[a_{j+1}, b_{j+1}]$ is obtained from the interval $[a_j, b_j]$ by halving while maintaining the following:

Invariant: For $j = 1, 2, \ldots$, the interval $[a_j, b_j]$ contains the median m.

Equivalently, the invariant can be stated as follows. For $j = 1, 2, \ldots$,

- the number of elements in $\cup_{i=1}^k A_i$ that are $\leq a_j$ is less than $\lceil t/2 \rceil$, and
- the number of elements in $\cup_{i=1}^k A_i$ that are $\leq b_j$ is at least $\lceil t/2 \rceil$.

Specifically, in round j, let

$$c_j = \left\lfloor \frac{a_j + b_j}{2} \right\rfloor.$$

Each player communicates the number of elements in his/her set that are $\leq c_j$. Since there are k players, this takes $O(k \log n)$ bits.[1] Once this is done, each player can compute independently (by adding the k individual counts) the total number of elements in $\cup_{i=1}^k A_i$ that are $\leq c_i$. If the number is less than $\lceil t/2 \rceil$, then we set $[a_{j+1}, b_{j+1}] := [c_j, b_j]$, otherwise,

[1] It was suggested by an anonymous reviewer that using approximate counts would improve the communication complexity from $O(k \log n)$ to $O(k \log k + \log n)$. Specifically, let x_i be the number of elements in A_i that are $\leq c_j$. Instead of x_i which needs $O(\log n)$ bits, player i posts $y_i = \lfloor x_i k/((0.5 - \alpha)t) \rfloor$ which can be represented in $O(\log k)$ bits. Then each player locally computes and uses $z_i = \lceil y_i(0.5 - \alpha)t/k \rceil$ to approximate the actual count x_i. Since $0 \leq x_i - z_i < (0.5 - \alpha)t/k$, the total error among all k players is at most $(0.5 - \alpha)t$ which seems to be within the mediocre range. However, we have a counterexample showing that this change will make the protocol return an element that is not $(\alpha t, \alpha t)$-mediocre. So it appears that this "improvement" is invalid. Furthermore, we note that any inaccuracy in the counts (for example, by using even a smaller factor $\beta < 0.5 - \alpha$ in the above strategy) may still result in choosing a different half of the interval $[a_j, b_j]$ which in turn can violate the invariant that the median m is always in the current interval.

i.e., the number is at least $\lceil t/2 \rceil$, then we set $[a_{j+1}, b_{j+1}] := [a_j, c_j]$. This setting maintains the invariant.

The protocol repeatedly halves the current interval until

$$b_j - a_j \leq \left(\frac{1}{2} - \alpha\right) t. \tag{2}$$

When this occurs, since $\cup_{i=1}^k A_i$ consists of distinct elements, $[a_j, b_j]$ contains a continuous range of no more than $\left(\frac{1}{2} - \alpha\right) t$ elements of $\cup_{i=1}^k A_i$, with m being one of them. If $(0.5 - \alpha)t < 1$, then the protocol stops when $b_j - a_j = 1$ and returns b_j as the median.

Let z be any element of $\cup_{i=1}^k A_i$ contained in $[a_j, b_j]$. (The protocol will return one such element, as explained below.) Observe that

$$\frac{t}{2} - \left(\frac{1}{2} - \alpha\right) t \leq \text{rank}_{\cup A_i}(z) \leq \frac{t}{2} + \left(\frac{1}{2} - \alpha\right) t, \text{ or}$$
$$\alpha t \leq \text{rank}_{\cup A_i}(z) \leq (1 - \alpha) t. \tag{3}$$

The number of halving rounds needed to achieve the interval-length in (2) is at most

$$\left\lceil \log \frac{n}{\left(\frac{1}{2} - \alpha\right) t} \right\rceil \leq \left\lceil \log \frac{n}{\left(\frac{1}{2} - \alpha\right) cn} \right\rceil = \left\lceil \log \frac{1}{\left(\frac{1}{2} - \alpha\right) c} \right\rceil = \left\lceil \log \frac{2q}{(q - 2p)c} \right\rceil$$
$$\leq \left\lceil \log \frac{2q}{c} \right\rceil = \ell = O(1).$$

In each round, the k players communicate their counts, $O(k \log n)$ bits in total. Each player independently computes the total count for the midpoint of the current interval, and all players take the same decision on how to set the next interval in the halving process (with no further communication needed).

In the last round (i.e., when inequality (2) is satisfied), the players report in turn. If the player does not hold any element in the interval $[a_j, b_j]$, he/she outputs a zero bit and the report continues; otherwise the player outputs such an element (from his/her set) in $O(\log n)$ bits and the protocol ends. The output element is a valid choice, as justified by (3).

The total communication complexity is therefore $O(\ell\, k \log n) = O(k \log n)$ bits, as claimed. This concludes the proof of Theorem 2. ◀

4 Approximate selection with two players under special conditions

Let $t = s + m$ denote the total number of elements in $A \cup B$. Here we consider the problem of finding an $(\alpha t, \alpha t)$-mediocre element between two players, where $\alpha \in (0, 1/2)$ is a fixed constant. The protocol described in Subsection 2.1 immediately yields the following.

► **Corollary 5.** *The deterministic communication complexity of finding an $(\alpha t, \alpha t)$-mediocre element in $A \cup B \subset [n]$, where $t = |A| + |B|$ and $\alpha \in (0, 1/2)$ is a fixed constant, is $O(\log n)$.*

Interestingly enough, this communication complexity can be brought down to a constant under slightly stronger assumptions: (i) A and B have no duplicates or common elements, and (ii) $|A \cup B| \geq cn$, for some constant $c > 0$; and a somewhat relaxed requirement: at least one of the players returns an element to the process that has invoked his/her service; each element returned is $(\alpha t, \alpha t)$-mediocre. Note that this is a natural relaxation — if the set of one player does not contain any suitable element, it is impossible to communicate the final answer to this player within $O(1)$ complexity.

A natural protocol to consider would be to choose one of the median-finding protocols and execute a constant number of rounds from it. However, this seemingly promising idea does not appear to work. It is possible that one of the two sets, say A, does not contain any desired elements, namely $(\alpha t, \alpha t)$-mediocre for the given α and so at the end of the modified protocol only B' contains desired elements (and not A'). More importantly, the players apparently have no indication of which player is the lucky one. We therefore resort to a different idea of using quantiles (more precisely, a sampling technique with a similar effect).

Proof of Theorem 3. We may assume, without loss of generality that n and $1/c$ are powers of 2 (in particular, $4n$ is also a power of 2). For $n < 8q^2/c$ Alice and Bob use the earlier $O(\log n)$-protocol for finding the median; we therefore subsequently assume that $n \geq 8q^2/c$. In particular, since $q \geq 3$, we have $n \geq 24q/c$. We further assume, without loss of generality that $|A| = |B| = m$: this can be achieved by padding the smaller size set with the appropriate numbers of small elements and large elements as described below. In particular, the padding elements need also be distinct. (It is *not* assumed that the common size is a power of 2: since our protocol does not exactly halve the current set of each player at each round, such an assumption would be of no use.)

To illustrate the padding process for arbitrary set sizes, we may assume without loss of generality that the given input satisfies: $s = |A| \leq |B| = m$. Recall that s and m are known to both players. We need to pad Alice's input with $m - \lceil \frac{m+s}{2} \rceil$ small elements and $\lceil \frac{m+s}{2} \rceil - s$ large elements. Alice and Bob replace their inputs by $A + n$ and $B + n$, respectively; as a result, the elements they hold are now in the range $\{n+1, \ldots, 2n\}$. Then Alice pads her input with $\{1, 2, \ldots, m - \lceil \frac{m+s}{2} \rceil\} \subset [n]$ and $\{2n+1, \ldots, 2n + \lceil \frac{m+s}{2} \rceil - s\} \subset [3n] \setminus [2n]$. (Note that $\lceil \frac{m+s}{2} \rceil - s = m - \lfloor \frac{m+s}{2} \rfloor$.) The resulting sets have the same size m and $A \cup B$ consists of distinct elements in the range $[3n] \subset [4n]$. By subtracting n, the element(s) returned by the protocol are shifted back to the original range $[n]$ in the end (without explicitly mentioning it there).

A and B below denote the (new) padded sets (of size m). Set $h = \lceil \frac{2q}{q-2p} \rceil$ (recall that $\alpha = p/q$) and $\ell = \lceil \log \frac{12h}{c} \rceil$. By the assumption $n \geq 24q/c$ we have

$$cn \geq 24q \geq 12 \left\lceil \frac{2q}{q-2p} \right\rceil = 12h.$$

Let Q_A be the set consisting of the $i\lfloor m/h \rfloor$-th elements of A, for $i = 1, 2, \ldots, h$. Similarly, let Q_B be the set consisting of the $i\lfloor m/h \rfloor$-th elements of B, for $i = 1, 2, \ldots, h$. (These sets resemble the h-th quantiles of A and B). Note that $|Q_A| = |Q_B| = h$. Since A and B consist of pairwise distinct elements, between any two elements in Q_A (or Q_B), there are at least

$$\left\lfloor \frac{m}{h} \right\rfloor \geq \frac{m}{h} - 1 \geq \frac{t}{2h} - 1 \geq \frac{cn}{2h} - 1 \geq \frac{cn}{3h} \geq \frac{4n}{2^\ell}$$

elements. Represent each element x in Q_A (and Q_B) with $\log(4n) = \log n + 2$ bits; it follows that the elements in $\{\texttt{prefix}_\ell(x) : x \in Q_A\}$ are pairwise distinct; similarly the elements in $\{\texttt{prefix}_\ell(y) : y \in Q_B\}$ are pairwise distinct.

The protocol implements a binary-search strategy aimed at finding the median of $Q_A \cup Q_B$. Note that $|Q_A| = |Q_B| \leq h$. Alice maintains a set $Q'_A \subset Q_A$ of elements that may still be the median quantile (initially $Q'_A = Q_A$) and Bob maintains a set $Q'_B \subset Q_B$ of elements that may still be the median quantile (initially $Q'_B = Q_B$). The invariant $|Q'_A| = |Q'_B|$ will be maintained. At each round, Alice and Bob compute the medians of their current sets (x_A and x_B, respectively). If $\texttt{prefix}_\ell(x_A) < \texttt{prefix}_\ell(x_B)$ or $\texttt{prefix}_\ell(x_A) > \texttt{prefix}_\ell(x_B)$ the protocol continues with Alice and Bob halving their input as in the median-finding protocol. Specifically, if $\texttt{prefix}_\ell(x_A) < \texttt{prefix}_\ell(x_B)$ the protocol discards the $\lfloor |Q'_A|/2 \rfloor$ lower elements of Q'_A and the $\lfloor |Q'_B|/2 \rfloor$ upper elements of Q'_B. The equality case $\texttt{prefix}_\ell(x_A) = \texttt{prefix}_\ell(x_B)$ is addressed below. Observe that the above comparison can be resolved by exchanging ℓ bits in each round.

If $\texttt{prefix}_\ell(x_A) = \texttt{prefix}_\ell(x_B)$, and $|Q'_A| = |Q'_B| \geq 3$, we have $\texttt{prefix}_\ell(\texttt{pred}(x_A)) < \texttt{prefix}_\ell(x_B)$, and the protocol discards the $\lfloor (|Q'_A| - 1)/2 \rfloor$ lower elements of Q'_A and the $\lfloor (|Q'_B| - 1)/2 \rfloor$ upper elements of Q'_B. Note that this is a slight but important deviation from the standard median-finding protocol; it is aimed at handling prefix equality by discarding possibly one fewer element by each player. With this choice, the median of $Q_A \cup Q_B$ remains the median of $Q'_A \cup Q'_B$; and the invariant $|Q'_A| = |Q'_B|$ is maintained. Since the sets the players hold are almost halved at each round, the protocol terminates in $O(\log h)$ rounds, as specified below.

If $|Q'_A| = |Q'_B| = 2$, and $\texttt{prefix}_\ell(x_A) \neq \texttt{prefix}_\ell(x_B)$, the protocol continues with each player halving his/her own current set accordingly. If $|Q'_A| = |Q'_B| = 2$, and $\texttt{prefix}_\ell(x_A) = \texttt{prefix}_\ell(x_B)$, the protocol terminates with each player output his/her number (x_A and x_B, respectively). Observe that in this case, the median of $Q_A \cup Q_B$ is x_A or x_B and it will be shown below, see (7), that both elements are $(\alpha t, \alpha t)$-mediocre.

If $|Q'_A| = |Q'_B| = 1$ and $\texttt{prefix}_\ell(x_A) \neq \texttt{prefix}_\ell(x_B)$, the protocol terminates with the player that holds the smaller of x_A and x_B output that number. If $|Q'_A| = |Q'_B| = 1$ and $\texttt{prefix}_\ell(x_A) = \texttt{prefix}_\ell(x_B)$, the protocol terminates with each player output his/her number (x_A and x_B, respectively). It will be shown below, see (7), that both elements are $(\alpha t, \alpha t)$-mediocre.

Recall that $\ell = \lceil \log \frac{12h}{c} \rceil$. If $x, y \in [3n]$ and $\texttt{prefix}_\ell(x) = \texttt{prefix}_\ell(y)$ then

$$|x - y| \leq \frac{3n}{2^\ell} \leq \frac{cn}{4h} \leq \frac{t}{4h}. \tag{4}$$

Recall that the median of $Q_A \cup Q_B$ is in $Q'_A \cup Q'_B$ in the last round of the protocol. Since all elements are distinct, for x_A and x_B above, if $\texttt{prefix}_\ell(x_A) = \texttt{prefix}_\ell(x_B)$, Inequality (4) implies

$$|\text{rank}_{A\cup B}(x_A) - \text{rank}_{A\cup B}(x_B)| \leq \frac{t}{4h}. \tag{5}$$

Assume that the median of $Q_A \cup Q_B$ is $x_A \in Q_A$; then Alice returns x_A. In addition, if $\texttt{prefix}_\ell(x_A) = \texttt{prefix}_\ell(x_B)$, Bob also returns $x_B \in Q_B$. Since x_A is the median of $Q_A \cup Q_B$, it is the h-th smallest element of $Q_A \cup Q_B$. As such (by construction): (i) x_A is $\geq$ than at least

$$h \left\lfloor \frac{m}{h} \right\rfloor \geq h\left(\frac{m}{h} - 1\right) = m - h$$

elements of $A \cup B$; and similarly, (ii) x_A is $\leq$ than at least $m - h$ elements of $A \cup B$. Note that the median of $A \cup B$ has rank m and is the same as the median of the original union of the two sets. See Fig. 2.

Observe that $h = \lceil \frac{2q}{q-2p} \rceil \leq 2q$ which yields $2h^2 \leq 8q^2 \leq cn \leq t$ (recall that $n \geq 8q^2/c$). This implies

$$|\text{rank}_{A\cup B}(x_A) - m| \leq h \leq \frac{t}{2h}. \tag{6}$$

Recall that if $\texttt{prefix}_\ell(x_A) = \texttt{prefix}_\ell(x_B)$, Bob also returns $x_B \in Q_B$ and Inequality (5) applies. From (5) and (6) we deduce that the rank of any output element z satisfies (recall

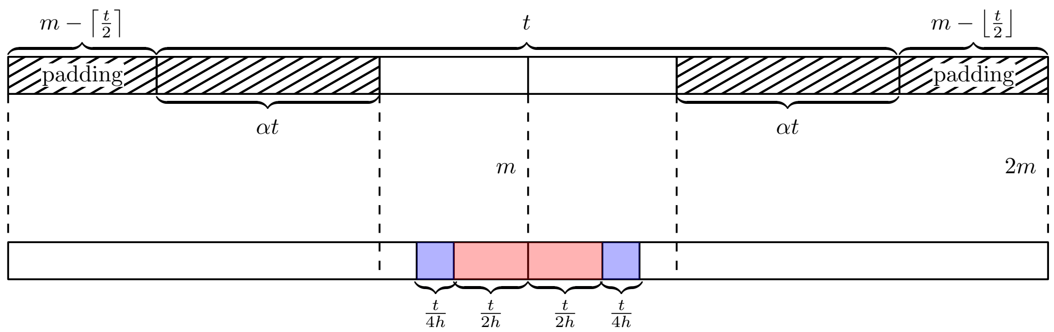

Figure 2 Above: Illustration of the original union of the two input sets with padding elements. The players need to find elements from the unshaded region in the middle. Below: The median x of $Q_A \cup Q_B$ lies within the red region. If the other player has an element y such that $\texttt{prefix}_\ell(y) = \texttt{prefix}_\ell(x)$, then y lies in the union of the red and blue regions, therefore it is also a valid output.

that $t = s + m$):

$$|\text{rank}_{A\cup B}(z) - m| \leq \frac{t}{4h} + \frac{t}{2h} \leq \frac{t}{h} \leq \frac{(q-2p)t}{2q} = \left(\frac{1}{2} - \alpha\right) t. \tag{7}$$

As such, each output element z is an $(\alpha t, \alpha t)$-mediocre element of the original union of the two sets. The elements returned are x_A or x_B (or both). Alice may return x_A and Bob may return x_B to the processes that have invoked their service; the elements returned by the players could be different. Since $q = O(1)$, we have $h, \ell = O(1)$. The number of bits exchanged is $\ell + O(1) = O(1)$ in each of the $O(\log h)$ rounds of the protocol. The overall communication complexity is $O(\ell \log h) = O(1)$, as claimed. ◀

5 Conclusion

An obvious question is whether the three-party communication complexity of median computation can be reduced to $O(\log n)$. A more general question is whether the k-party communication complexity of median computation, $k \geq 3$, can be reduced to $O(k \log n)$. We believe that the answers to both questions are in the negative. Another interesting question regarding the two-party communication complexity of approximate selection is whether the conditions in Theorems 2 and 3 can be relaxed.

References

1 Alfred V. Aho, John E. Hopcroft, and Jeffrey D. Ullman. *Data Structures and Algorithms*. Addison-Wesley, 1983.

2 Miklós Ajtai, János Komlós, William L. Steiger, and Endre Szemerédi. Optimal parallel selection has complexity $o(\log\log n)$. *Journal of Computer and System Sciences*, 38(1):125–133, 1989. URL: `https://doi.org/10.1016/0022-0000(89)90035-4`, `doi:10.1016/0022-0000(89)90035-4`.

3 Andrei Alexandrescu. Fast deterministic selection. In Costas S. Iliopoulos, Solon P. Pissis, Simon J. Puglisi, and Rajeev Raman, editors, *Proceedings of the 16th International Symposium on Experimental Algorithms, SEA 2017, London, UK, June 21-23, 2017*, volume 75 of *LIPIcs*, pages 24:1–24:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. URL: `https://doi.org/10.4230/LIPIcs.SEA.2017.24`, `doi:10.4230/LIPIcs.SEA.2017.24`.

4 Sara Baase. *Computer algorithms - introduction to design and analysis*. Addison-Wesley, 1988.

5 Sebastiano Battiato, Domenico Cantone, Dario Catalano, Gianluca Cincotti, and Micha Hofri. An efficient algorithm for the approximate median selection problem. In Gian Carlo Bongiovanni, Giorgio Gambosi, and Rossella Petreschi, editors, *Proceedings of the 4th Italian Conference on Algorithms and Complexity, CIAC 2000, Rome, Italy, March 2000*, volume 1767 of *Lecture Notes in Computer Science*, pages 226–238. Springer, 2000. URL: `https://doi.org/10.1007/3-540-46521-9_19`, `doi:10.1007/3-540-46521-9\_19`.

6 Samuel W. Bent and John W. John. Finding the median requires $2n$ comparisons. In Robert Sedgewick, editor, *Proceedings of the 17th Annual ACM Symposium on Theory of Computing, STOC 1985, Providence, Rhode Island, USA, May 6-8, 1985*, pages 213–216. ACM, 1985. URL: `https://doi.org/10.1145/22145.22169`, `doi:10.1145/22145.22169`.

7 Manuel Blum, Robert W. Floyd, Vaughan R. Pratt, Ronald L. Rivest, and Robert Endre Tarjan. Time bounds for selection. *Journal of Computer and System Sciences*, 7(4):448–461, 1973. URL: `https://doi.org/10.1016/S0022-0000(73)80033-9`, `doi:10.1016/S0022-0000(73)80033-9`.

8 Ke Chen and Adrian Dumitrescu. Selection algorithms with small groups. *International Journal of Foundations of Computer Science*, 31(3):355–369, 2020. URL: `https://doi.org/10.1142/s0129054120500136`, `doi:10.1142/s0129054120500136`.

9 Francis Y. L. Chin and Hing fung Ting. An improved algorithm for finding the median distributively. *Algorithmica*, 2:235–249, 1987. URL: `https://doi.org/10.1007/BF01840361`, `doi:10.1007/BF01840361`.

10 Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, 3rd Edition.* MIT Press, 2009. URL: `http://mitpress.mit.edu/books/introduction-algorithms`.

11 Walter Cunto and J. Ian Munro. Average case selection. *Journal of ACM*, 36(2):270–279, 1989. URL: `https://doi.org/10.1145/62044.62047`, `doi:10.1145/62044.62047`.

12 Sanjoy Dasgupta, Christos H. Papadimitriou, and Umesh V. Vazirani. *Algorithms.* McGraw-Hill, 2008.

13 Dorit Dor, Johan Håstad, Staffan Ulfberg, and Uri Zwick. On lower bounds for selecting the median. *SIAM Journal on Discrete Mathematics*, 14(3):299–311, 2001. URL: `https://doi.org/10.1137/S0895480196309481`, `doi:10.1137/S0895480196309481`.

14 Dorit Dor and Uri Zwick. Finding the αn-th largest element. *Combinatorica*, 16(1):41–58, 1996. URL: `https://doi.org/10.1007/BF01300126`, `doi:10.1007/BF01300126`.

15 Dorit Dor and Uri Zwick. Selecting the median. *SIAM Journal on Computing*, 28(5):1722–1758, 1999. URL: `https://doi.org/10.1137/S0097539795288611`, `doi:10.1137/S0097539795288611`.

16 Adrian Dumitrescu. Finding a mediocre player. In Pinar Heggernes, editor, *Proceedings of the 11th International Conference on Algorithms and Complexity, CIAC 2019, Rome, Italy, May 27-29, 2019*, volume 11485 of *Lecture Notes in Computer Science*, pages 212–223. Springer, 2019. URL: `https://doi.org/10.1007/978-3-030-17402-6_18`, `doi:10.1007/978-3-030-17402-6\_18`.

17 Adrian Dumitrescu. A selectable sloppy heap. *Algorithms, special issue on efficient data structures*, 12(3):58, 2019. URL: `https://doi.org/10.3390/a12030058`, `doi:10.3390/a12030058`.

18 Stefan Edelkamp and Armin Weiß. Worst-case efficient sorting with quickmergesort. In Stephen G. Kobourov and Henning Meyerhenke, editors, *Proceedings of the 21st Workshop on Algorithm Engineering and Experiments, ALENEX 2019, San Diego, CA, USA, January 7-8, 2019*, pages 1–14. SIAM, 2019. URL: `https://doi.org/10.1137/1.9781611975499.1`, `doi:10.1137/1.9781611975499.1`.

19 Robert W. Floyd and Ronald L. Rivest. Expected time bounds for selection. *Communications of ACM*, 18(3):165–172, 1975. URL: `https://doi.org/10.1145/360680.360691`, `doi:10.1145/360680.360691`.

20 Frank Fussenegger and Harold N. Gabow. A counting approach to lower bounds for selection problems. *Journal of ACM*, 26(2):227–238, 1979. URL: `https://doi.org/10.1145/322123.322128`, `doi:10.1145/322123.322128`.

21 Abdollah Hadian and Milton Sobel. Selecting the t-th largest using binary errorless comparisons. Technical Report No. 121, School of Statistics, University of Minnesota, 1969. URL: `http://hdl.handle.net/11299/199105`.

22 Charles Antony Richard Hoare. Algorithm 63: Partition and algorithm 65: Find. *Communications of ACM*, 4(7):321–322, 1961. URL: `http://doi.acm.org/10.1145/366622.366647`, `doi:10.1145/366622.366647`.

23 Laurent Hyafil. Bounds for selection. *SIAM Journal on Computing*, 5(1):109–114, 1976. URL: `https://doi.org/10.1137/0205010`, `doi:10.1137/0205010`.

24 John W. John. A new lower bound for the set-partitioning problem. *SIAM Journal on Computing*, 17(4):640–647, 1988. URL: `https://doi.org/10.1137/0217040`, `doi:10.1137/0217040`.

25 Haim Kaplan, László Kozma, Or Zamir, and Uri Zwick. Selection from heaps, row-sorted matrices, and $x+y$ using soft heaps. In Jeremy T. Fineman and Michael Mitzenmacher, editors, *Proceedings of the 2nd Symposium on Simplicity in Algorithms, SOSA 2019, San Diego, CA, USA, January 8-9, 2019*, volume 69 of *OASICS*, pages 5:1–5:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. URL: `https://doi.org/10.4230/OASIcs.SOSA.2019.5`, `doi:10.4230/OASIcs.SOSA.2019.5`.

26 David G. Kirkpatrick. A unified lower bound for selection and set partitioning problems. *Journal of ACM*, 28(1):150–165, 1981. URL: `https://doi.org/10.1145/322234.322245`, `doi:10.1145/322234.322245`.

27 Jon M. Kleinberg and Éva Tardos. *Algorithm design.* Addison-Wesley, 2006.

28 Donald E. Knuth. *The art of computer programming, Volume III: Sorting and Searching, 2nd Edition.* Addison-Wesley, 1998. URL: `https://www.worldcat.org/oclc/312994415`.

29 Eyal Kushilevitz and Noam Nisan. *Communication complexity.* Cambridge University Press, 1997.

30 Eyal Kushilevitz, Noam Nisan, and Bill Gasarch. Errata of communication complexity. `http://www.cs.technion.ac.il/~eyalk/book.html`.

31 S. L. Mantzaris. On "an improved algorithm for finding the median distributively". *Algorithmica*, 10(6):501–504, 1993. URL: `https://doi.org/10.1007/BF01891834`, `doi:10.1007/BF01891834`.

32 Conrado Martínez and Salvador Roura. Optimal sampling strategies in quicksort and quickselect. *SIAM Journal on Computing*, 31(3):683–705, 2001. URL: `https://doi.org/10.1137/S0097539700382108`, `doi:10.1137/S0097539700382108`.

33 Catherine C. McGeoch and J. Doug Tygar. Optimal sampling strategies for quicksort. *Random Structures & Algorithms*, 7(4):287–300, 1995. URL: `https://doi.org/10.1002/rsa.3240070403`, `doi:10.1002/rsa.3240070403`.

34 Mike Paterson. Progress in selection. In Rolf G. Karlsson and Andrzej Lingas, editors, *Proceedings of the 5th Scandinavian Workshop on Algorithm Theory SWAT 1996, Reykjavík, Iceland, July 3-5, 1996*, volume 1097 of *Lecture Notes in Computer Science*, pages 368–379. Springer, 1996. URL: `https://doi.org/10.1007/3-540-61422-2_146`, `doi:10.1007/3-540-61422-2\_146`.

35 Anup Rao and Amir Yehudayoff. *Communication complexity and applications.* Cambridge University Press, 2020. URL: `https://doi.org/10.1017/9781108671644`.

36 Michael Rodeh. Finding the median distributively. *Journal of Computer and System Sciences*, 24(2):162–166, 1982. URL: `https://doi.org/10.1016/0022-0000(82)90045-9`, `doi:10.1016/0022-0000(82)90045-9`.

37 Arnold Schönhage, Mike Paterson, and Nicholas Pippenger. Finding the median. *Journal of Computer and System Sciences*, 13(2):184–199, 1976. URL: `https://doi.org/10.1016/S0022-0000(76)80029-3`, `doi:10.1016/S0022-0000(76)80029-3`.

38 Andrew Chi-Chih Yao. Some complexity questions related to distributive computing (preliminary report). In Michael J. Fischer, Richard A. DeMillo, Nancy A. Lynch, Walter A. Burkhard, and Alfred V. Aho, editors, *Proceedings of the 11th Annual ACM Symposium on Theory of Computing, STOC 1979, Atlanta, Georgia, USA, April 30 - May 2, 1979*, pages 209–213. ACM, 1979. URL: `https://doi.org/10.1145/800135.804414`, `doi:10.1145/800135.804414`.

39 F. Frances Yao. On lower bounds for selection problems. Technical Report MAC TR-121, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA, 1974.

40 Chee-Keng Yap. New upper bounds for selection. *Communications of ACM*, 19(9):501–508, 1976. URL: `https://doi.org/10.1145/360336.360339`, `doi:10.1145/360336.360339`.