Discrete Mathematics 114 (1993) 425–433
North-Holland

Parallel searching of multidimensional cubes

Miklos Santha
Université Paris-Sud, LRI, 91405 Orsay, France

Umesh V. Vazirani*
University of California, Berkeley, CA 94720, USA

Received 6 January 1989
Revised 4 February 1990

Abstract

Santha, M. and U.V. Vazirani, Parallel searching of multidimensional cubes, Discrete Mathematics 114 (1993) 425–433.

We prove a tight lower bound of $\Omega(\log\log n)$ in the parallel decision tree model, on the complexity of searching the d-dimensional cube of side n using n^{d-1} processors. The lower bound is valid even for randomized algorithms which err with constant probability.

1. Introduction

The general problem of searching an ordered data structure using comparison queries is formulated by Linial and Saks [4] as follows: Let $(P, \leqslant)$ be a finite partially ordered set (the data structure). Let $f: P \to R$ be an order-preserving injective real function, i.e. satisfying $p \neq q \Rightarrow f(p) \neq f(q)$ and $p < q \Rightarrow f(p) < f(q)$ (f is referred to as a *storage function*). The searching problem associated with P is: Given a real number x, determine whether there exists an element $p \in P$ such that $f(p) = x$, and find such an element if it exists. An elementary step is the evaluation of f at some element of P. A sequential algorithm can make an evaluation every step, a parallel algorithm using k processors may perform k evaluations simultaneously in one step. Following Valiant's parallel decision tree model [7], we shall assume that the k queries in any round can be an arbitrarily complex function of the answers to queries from previous rounds. The complexity $c(P)$ of the problem is the minimum taken over all algorithms

Correspondence address: Miklos Santha, LRI, Universite Paris-Sud, 91405 Orsay, France.
* Part of this research done while visiting the Université Paris-Sud, LRI. Supported in part by NSF Grant CCR-8947792.

which solve the problem of the maximum number of steps taken by the algorithm. When k processors are used, the complexity is denoted by $c_k(P)$.

We will consider the searching problem when P is a product of several chains of the same length. Let $d \geqslant 2$ be an integer and let $Q_{n,d}$ be the set $\{1, 2, \ldots, n\}^d$ with the following partial order:

$$(i_1, \ldots, i_d) \leqslant (j_1, \ldots, j_d) \quad \text{if } i_1 \leqslant j_1, \ldots, i_d \leqslant j_d.$$

$Q_{n,d}$ is just a d-dimensional cube of side n in the fundamental lattice of the Euclidean space R^d, where points are compared coordinatewise. We will be concerned with $W(n, d, k) = c_k(Q_{n,d})$, the complexity of searching $Q_{n,d}$ with k processors.

The complexity of searching multidimensional cubes sequentially was investigated by Linial and Saks [3]. They proved, that, for every $d \geqslant 2$, searching $Q_{n,d}$ takes $\Theta(n^{d-1})$ evaluations.

Shearer [6] observed that the problem of searching $Q_{n,2}$ is equivalent to the merging of two lists $u_1 < \cdots < u_n$ and $v_1 < \cdots < v_n$ of distinct real numbers. A correspondence between the two problems can be established by showing that any algorithm for merging lists can be transformed into a cube-searching algorithm (and vice versa) by replacing comparisons of the form $u_i < v_j$ by the query $x > f((i, n+1-j))$ (and vice versa). Then the computational trees representing the corresponding two algorithms are identical.

The parallel complexity of the merging problem is well studied: Valiant [7] gave a deterministic algorithm for merging two lists of n elements using n processors with $O(\log \log n)$ comparisons. Borodin and Hopcroft [1] proved that $\Omega(\log \log n)$ comparisons are also necessary. Geréb-Graus and Krizanc [2] established that the $\Omega(\log \log n)$ lower- bound remains valid even for parallel randomized algorithms.

In this paper we will generalize the results obtained for the parallel searching of the square $Q_{n,2}$. We will show that $W(n, d, n^{d-1}) = \Theta(\log \log n)$, and that, moreover, any randomized algorithm using n^{d-1} processors for searching $Q_{n,d}$ takes time $\Omega(\log \log n)$. The results are stated in form of asymptotic inequalities. We will not use 'floors' and 'ceilings' in the proofs, which will not affect their validity.

2. Basic notions

A subset $I \subseteq P$ is an *ideal* if it is satisfies: $p \in I$ and $q < p$ imply $q \in I$. The complement of an ideal is a *filter*. If I is an ideal, its complement is denoted by $\bar{I}$. For every $p \in P$, the ideal $I(p)$ generated by p is the set of elements which are smaller than or equal to p. The filter $F(p)$ generated by p is the set of elements which are greater than or equal to p. If for an evaluation it turns out that $f(p) < x$, then for every $q \in I(p)$, $f(q) < x$; thus, the elements of $I(p)$ should not be searched any more. We will say that the evaluation *eliminates* $I(p)$. Similarly, if $f(p) > x$, then this evaluation eliminates $F(p)$.

The search problem induces the ideal of the elements in P which are smaller than x. It is clear that the knowledge of this ideal is sufficient to solve the problem. On the

other hand, when x is not in the image of f, identifying this ideal is necessary to solve the search problem . Thus, the search problem is actually equivalent to the identification of the ideal induced by it. We will also use this equivalent version of the problem.

A subset of P is *convex* if it is the intersection of an ideal and a filter. Linial and Saks [4] have shown that if $S \subseteq P$ is convex then $c(S) \leqslant c(P)$. Their result easily generalizes to the parallel case.

Proposition 2.1. *If S is a convex subset of P then, for every k, $c_k(S) \leqslant c_k(P)$.*

Proof. Let f_S be the storage function for which $c_k(S)$ queries are necessary. We show how to extend f_S into a storage function f_P for P such that solving this new search problem is at least as hard, thus showing that $c_k(P) \geqslant c_k(S)$. By definition, f_P is identical with f_S on S, its value being negative infinity on every point in the ideal which is not in S, and infinity everywhere else. □

If the number of elements in P is greater than the number of processors, than an adversary can always answer the evaluations in one step such that some elements are not eliminated. This gives the following result.

Proposition 2.2. *For every k, $k < |P|$ if and only if $c_k(P) > 1$.*

Two elements p and q are *incomparable* if $p \not\leqslant q$ and $q \not\leqslant p$. A set of elements are incomparable if any two elements of the set are incomparable. Two subsets A and B are incomparable if, for every $p \in A$ and $q \in B$, p and q are incomparable. Finally, a set of subsets are incomparable if any two subsets are incomparable. Clearly, the union of incomparable convex subsets is a convex subset.

We need some definitions about cubes. In the rest of the paper, by a cube we always mean a d-dimensional cube. A set of points in $Q_{d,n}$ are *coplanar* if they are in the same hyperplane of R^d. A *subcube* of $Q_{d,n}$ of side s is a subset of the form

$$\prod_{k=1}^{d} \{i_k, i_k + 1, \ldots, i_k + s - 1\},$$

where $1 \leqslant i_k \leqslant n - s + 1$ for $k = 1, \ldots, d$. The subcube which is uniquely determined by its smallest element $(i, \ldots, i_d)$ and its side s will be denoted as $C((i_1, \ldots, i_d), s)$. The center of a subcube is the point whose coordinates are the average of the corresponding coordinates of the vertices. A *main* hyperplane of a subcube is a hyperplane which contains an element of the subcube and is orthogonal to the vector $(1,1, \ldots, 1)$. The elements in a main hyperplane are incomparable. Let H be a main hyperplane of the cube and let p be an element of the cube. By definition, $p \leqslant H$ ($p \geqslant H$) if there exists $q \in H$ such that $p \leqslant q$ ($p \geqslant q$). If $p \leqslant H$ and $p \notin H$, then p is *below* H (in notation $p < H$). We note that this notion coincides with the geometrical notion of the point p being below the hyperplane H since H is a main hyperplane. Symmetrically, if $p \geqslant H$ and

$p \notin H$, then p is *above* H ($p > H$). The *central hyperplane* of a subcube is the main hyperplane which contains its center.

We will often *divide* cubes into aligned subcubes. The *division* of the cube $C((i_1, \ldots, i_d), s)$ into *aligned* subcubes of side t is the set of $(s/t)^d$ disjoint subcubes of side t of the form $C((i_1 + j_1 t, \ldots, i_d + j_d t), t)$, where $j_1, \ldots, j_d \in \{0, \ldots, (s/t) - 1\}$. A division of a cube into aligned subcubes of side t induces $d(s/t - 1) + 1$ *diagonals*, where a diagonal is the set of the aligned subcubes which have the same central hyperplane. The common central hyperplane is also called the *central hyperplane* of the diagonal.

The following proposition says that, in a subcube of side s, a main hyperplane which is not too far away from the central hyperplane contains $\Omega(s^{d-1})$ elements.

Proposition 2.3. *Let $\frac{1}{2} > \eta > 0$ be a constant. Let $C(s)$ be a subcube of side s and let $H(s)$ be a main hyperplane of $C(s)$ whose distance from its central hyperplane is $d^{1/2}((s+1)/2 - \eta s)$. Then there exists a constant $c(\eta, d) > 0$, such that*

$$\lim_{s \to \infty} \frac{|C(s) \cap H(s)|}{s^{d-1}} = c(\eta, d).$$

Proof. For any integer s, let $X_1, \ldots, X_d$ be identically distributed independent random variables, where $\Pr[X_i = t] = 1/s$ for $t = 1/s, \ldots, s/s$. Then we have

$$\begin{aligned} |C(s) \cap H(s)| &= s^d \Pr\left[\sum_{i=1}^{d} X_i = \eta d\right] \\ &= s^d \sum_{t=1}^{s} \Pr[X_d = t/s] \Pr\left[\sum_{i=1}^{d-1} X_i = \eta d - t/s\right] \\ &= s^d s^{-1} \Pr\left[\eta d - 1 \leqslant \sum_{i=1}^{d-1} X_i \leqslant \eta d - 1/s\right]. \end{aligned}$$

Let $Y_1, \ldots, Y_d$ be identically distributed independent random variables with uniform density on $[0, 1]$. When s goes to infinity, the distribution of X_i approaches the distribution of Y_i. Thus,

$$\lim_{s \to \infty} \frac{|C(s) \cap H(s)|}{S^{d-1}} = \Pr\left[\eta d - 1 \leqslant \sum_{i=1}^{d-1} Y_i \leqslant \eta d\right] = c(\eta, d). \qquad \square$$

3. Lower bound for deterministic algorithms

Theorem 3.1. *For every $d \geqslant 2$,*

$$W(n, d, n^{d-1}) = \Theta(\log \log n).$$

Proof. The upper bound is easily achieved by partitioning the cube $Q_{n,d}$ into n^{d-2} squares of side n, and searching each with n processors using Valiant's algorithm.

For the lower bound, we use an inductive argument to prove the following stronger claim: Let $T(s,d,p)=\min_k\{\#$ steps to search k incomparable cubes of side s with pk processors$\}$ (i.e. the average number of processors per cube is p). Then $T(n,d,n^{d-1})=\Omega(\log\log n)$,

Claim 3.2. *If* $s^d>p$ *then*

$$T(s,d,p)\geqslant 1+T(s/2p^{d-1},d,4dp^{d-1}).$$

Proof. By Proposition 2.2, $T(s,d,p)>1$. Let k be the number of cubes for which $T(s,d,p)$ is minimal. We divide each of these cubes into $3p$ subcubes of side $s/(3p)^{d-1}>s/2p^{d-1}$. Since the average number of processors per cube is p, at last $k/2$ cubes are each assigned no more than $2p$ processors. Consider only such cubes. In each such cube there are at least p subcubes without any evaluation. The division induces less then $d(3p)^{d-1}$ diagonals; thus, there exists a diagonal containing at least $p/d(3p)^{d-1}>p^{1-d^{-1}}/2d$ subcubes without evaluation. Let us fix in each cube such a diagonal and its subcubes without evaluation. Altogether, there are at least $kp^{1-d^{-1}}/4d$ subcubes of side at least $s/2p^{d-1}$ without any evaluation. Let r be an element in one of the fixed cubes, and let H be the central hyperplane of the diagonal fixed in this cube. If the adversary answers '$f(r)>x$' if and only if r is above H, then no element in the subcubes is eliminated. As their union forms a convex set, Proposition 2.1 implies the result. □

We are interested in $T(n,d,n^{d-1})$. Let us define two series a_i and b_i by recursion: $a_0=b_0=1$, $a_{i+1}=a_i/2b_i^{d-1}$, and $b_{i+1}=4db_i^{d-1}$. Iterating Claim 3.2, an easy induction shows the following claim to be true.

Claim 3.3. *If* $(a_{i-1}n^{d-(i-1)})^d>b_in^{(d-1)d-i}$, *then*

$$T(n,d,n^{d-1})\geqslant i+T(a_in^{d-i},\ d,\ b_{i-1}n^{(d-1)d-(i-1)}).$$

It is also simple to show by induction the following (not at all tight) bounds on a_i and b_i:

$$b_i\leqslant(4d)^i \quad\text{and}\quad a_i\geqslant 2^{-i}(4d)^{-(i-1)i/2}.$$

Proof of Theorem 3.1 (*conclusion*). To finish the proof of the theorem, we observe that, for some sufficiently small constant ε, $i=\varepsilon\log\log n$ satisfies the condition of Claim 3.3; thus, the complexity of the problem is indeed $\Omega(\log\log n)$. □

4. Lower bound for randomized algorithms

One way of proving lower bounds on randomized algorithms was suggested by Yao [8]. He pointed out that the famous minimax theorem of von Neumann [5] implies

the following: the running time of any randomized algorithms on its worst input is bounded below by the expected running time of the best deterministic algorithm for any fixed distribution on the inputs. (Actually, the minimax theorem implies that if we take infimum on the randomized algorithms, and supremum on the input distributions, then these two quantities are equal). Thus, in order to prove a $\Omega(\log\log n)$ lower bound for every randomized algorithm which searches $Q_{n,d}$, it is sufficient to provide an input distribution on which every deterministic algorithm takes $\Omega(\log\log n)$ expected time.

We shall define a distribution on ideals such that, on an average, the best deterministic algorithm requires $\Omega(\log\log n)$ rounds of probes to determine the ideal chosen. To facilitate some measure of the progress of the searching algorithm in the middle of its execution, we shall view the algorithm at any point in time as having computed an approximation to the ideal in question. This is formalized below.

Definition. Let A and B be two subsets of $Q_{n,d}$. The couple (A, B) is called an *approximation* if A is an ideal, B is a filter and $A \cap B = \emptyset$. If (A, B) is an approximation and I is an ideal, then I is *compatible* with (A, B) if $A \subseteq I$ and $B \subseteq \bar{I}$. For any subset A *of* $Q_{n,d}$, we say that an ideal I *crosses* A if $A \cap I \neq \emptyset$ and $A \cap \bar{I} \neq \emptyset$. Finally, for any family S of subsets of $Q_{n,d}$, I *crosses* S if I crosses the elements of S.

Say that (A, B) contains a set of queries if all the queries lie in $A \cup B$. Clearly, at any stage of the algorithm if (A, B) contains the queries made by the algorithm so far, then the algorithm has narrowed down its search for the ideal no better than the approximation (A, B). Indeed, to facilitate our argument showing the limits to the best approximation obtainable in i steps, we shall define the input probability distribution on ideals by first defining a probability distribution on approximations, and then picking a random ideal compatible with an approximation picked from this distribution. The probability distribution is uniform on a certain class of hierarchically defined approximations.

Definition. The unique 0-collection of cubes is the singleton containing $Q_{n,d}$. For $i > 0$, a set of cubes $S' = \{C'_1, \ldots, C'_m\}$, each of side s, is an *i-collection* if there exists an $(i-1)$-collection $S = \{C_1, \ldots, C_k\}$ such that, for the division of the cubes in S into aligned subcubes of side s, we have:

(i) For $j = 1, \ldots, m$, C'_j is one of the aligned subcubes.

(ii) For $j = 1, \ldots, k$, there exists a diagonal D_j of C_j such that

$$\{C'_t \in S' : C'_t \subseteq C_j\} \subseteq D_j.$$

The diagonal in (ii) is called the *defining* diagonal and we say that the i-collection S' is *derived* from the $(i-1)$-collection S. Clearly, the cubes in an i-collection are incomparable.

Definition. For $i=0,1,2,\ldots$, let us give a family of i-collections. If every i-collection in the family is derived from a unique $(i\text{-}1)$ collection in the family then we say that the family is *uniquely derivable* .We define now, by recursion on i, *the approximation generated* by an i-collection in a uniquely derivable family. For $i=0$, the unique 0-collection generates $(\emptyset,\emptyset)$. Let the i-collection $S'=\{C'_1,\ldots,C'_m\}$ be derived from the $(i-1)$-collection $S=\{C_1,\ldots,C_k\}$ and let us suppose that S generates (A,B). For $j=1,\ldots,k$, let H_j be the central hyperplane of the defining diagonal in C_j. Then S' generates (A',B'), where

$$A'=A\cup\bigcup_{j=1}^{k}\{x\in C_j\colon x\leqslant H_j,\ x\notin\bigcup_{l=l}^{m}C'_l\}.$$

$$B'=A\cup\bigcup_{j=1}^{k}\{x\in C_j\colon x>H_j,\ x\notin\bigcup_{l=1}^{m}C'_l\}.$$

We say that an ideal I is *compatible* with an i-collection S if I is compatible with the approximation generated by S. If an ideal is compatible with an i-collection then, for $0\leqslant l\leqslant i$, the l-collection *associated with* I is the l-collection which was used for the derivation of the i-collection I is compatible with. To facilitate the lower bound, we shall restrict the i-collections by insisting that they obey some uniformity properties. For definiteness, we shall associate some lexicographical ordering on all subcubes of a given size.

Definition. We define by recursion on i the family of *i-cubings* with size functon $s(i)$ and count function $\#(i)$. The only 0-cubing is the only 0-collection. An i-collection $S'=\{C'_1,\ldots,C'_m\}$ derived from the $(i-1)$-cubing $S=\{C_1,\ldots,C_k\}$ is an i cubing if the following two conditions are satisfied:

(i) For $j=1,\ldots,m$, C'_j has side $s(i)$.

(ii) For $j=1,\ldots,k$, S' contains exactly the first $\#(i)$ subcubes of the defining diagonal D_j.

Since the family of i-cubings is a uniquely derivable family of i-collections, the notion of an approximation generated by an i-cubing is well defined. Say that a diagonal of a cube in an $(i-1)$-cubing is *eligible* for an i-cubing if it has at least $\#(i)$ subcubes of size $s(i)$. The advantage of i-cubings is that the added uniformly conditions imply that a random ideal I consistent with a randomly chosen i-cubing (with functions $s(i)$ and $\#(i)$, satisfies the following properties:

(1) Let $0\leqslant l\leqslant i$, let C be a cube in the l-cubing associated with I, and let $D_1,\ldots,D_k$ be the eligible diagonals of C. Then, for $1\leqslant j\leqslant k$, the events $E_j=$'I crosses D_j, are disjoint and equiprobable.

(2) Let $0\leqslant l\leqslant i$, let $\{C_1,\ldots,C_m\}$ be the l-cubing associated with I, and, for $1\leqslant j\leqslant m$, let D_j be an eligible diagonal of C_j. The events $E_j=$'I crosses D_j' are mutually independent.

We now show the existence of i-cubings for $i=\varepsilon \log\log n$ and for suitable functions $s(i)$ and $\#(i)$, where ε is some sufficiently small constant. Let $b_i=(c(1/4,d),\ 10c(1/2,d))^i$, where $c(1/4,d)$ and $c(1/2,d)$ are the constants in Proposition 2.3. Let $s(i)=b_i n^{d^{-i}}$ and $\#(i)=10n^{(d-1)(d-1)d^{-i}}$ Let C be a cube of an $(i-1)$-cubing. The division of C into aligned subcubes of side $s(i)$ induces $d(b_{i-1}/b_i)n^{(d-1)d^{-i}}-d+1$ diagnals. A diagonal is called a *middle diagonal* if it is one of the $n^{(d-1)d^{-i}}$ closest diagonals to the central hyperplane of C. We claim that every middle diagonal is eligible. Indeed, as $b_i/db_{i-1}<1/2$, Proposition 2.3 implies that the number of subcubes in a middle diagonal is

$$c(1/4,d)b_{i-1}^{d-1}n^{(d-1)d^{-(i-1)}}/c(1/2,d)b_i^{d-1}n^{(d-1)d^{-i}}\geqslant \#(i).$$

Thus, i-cubings exist with size and count function defined as above for $i=\varepsilon\log\log n$, for some small enough constant ε. Each such i-cubing has $10^i n^{(d-1)(1-d^{-i})}$ cubes.

Definition. The input distribution is the uniform distribution over the ideals which are compatible with an $\varepsilon\log\log n$-cubing with size function $s(i)$ and count function $\#(i)$ defined as above.

Theorem 4.1. *For every $d\geqslant 2$, the randomized complexity of searching $Q_{n,d}$ with n^{d-1} processors is $\Omega(\log\log n)$.*

Proof. We will show that, with the above input distribution, any deterministic algorithm using n^{d-1} processors takes expected time $\Omega(\log\log n)$ for searching $Q_{n,d}$; thus, Yao's theorem implies the result. Let us consider any searching algorithm and let I be a random ideal from the above distribution. For $i=0,1,\ldots,\varepsilon\log\log n$, let A_i be the set of elements eliminated in the first i steps. We will prove by induction on i, that, after i steps, with probability at least $2(1-1/n)^{i-1}/3$ (with probability 1 if $=0$), at least $n^{(d-1)(1-d^{-i})}$ cubes from the i-cubing associated with I are included in $Q_{n,d}-A_i$. We will call these cubes *unknown*. For $i=\varepsilon\log\log n$, it will follow that, with probability $2/3-o(1)$, the algorithm makes at least $\varepsilon\log\log n$ steps; thus, its expected running time is indeed $\Omega(\log\log n)$.

The claim is true for $i=0$, let us suppose that we have proven it for some $i-1$. In the next step the algorithm makes n^{d-1} evaluations. On an average, an unknown cube gets at most $n^{(d-1)d^{-(i-1)}}$ evaluations; thus, at least half of them get at most twice as much. Let the set of cubes which get at most $2n^{(d-1)d^{-(i-1)}}$ evaluations be $\{C_1,\ldots,C_k\}$, where $k=n^{(d-1)(1-d^{-(i-1)})}/2$. For $1\leqslant j\leqslant k$, let D_j be the defining diagonal in C_j. For every j, in C_j an eligible diagonal gets on an average at most $2n^{(d-1)(d-1)d^{-i}}$ evaluations, because the number of eligible diagonals is at least the number of middle diagonals. Thus, in a fraction of at least 2/3 of the eligible diagonals at most $6n^{(d-1)(d-1)d^{-i}}$ elements are evaluated. Property (1) implies that, with probability at least 2/3, at most $6n^{(d-1)(d-1)d^{-i}}$ elements are evaluated in the union of the subcubes in D_j. As the i-cubing associated with I contains the first $\#(i)$ *subcubes of* D_j, with

probability at least 2/3, we can take $4n^{(d-1)(d-1)d^{-i}}$ of them without any evaluations. If $i=1$, any $n^{(d-1)(d-1)d^{-1}}=n^{(d-1)(1-d^{-1})}$ of them can be chosen for the claimed subcubes.

If $i>1$ then, for $1\leqslant j\leqslant k$, let X_j be a 0–1 valued random variable, where, by definition, $X_j=1$ if among the first $\#(i)$ subcubes of D_j there are $4n^{(d-1)(d-1)d^{-i}}$ subcubes without any evaluations. Property (2) implies that these are mutually independent random variables, where $\Pr[X_j=1]\geqslant 2/3$. Let $Y=\sum_{j=1}^{k}X_j$. If $Y\geqslant k/2$ then there are $\frac{1}{2}k4n^{(d-1)(d-1)d^{-i}}$ subcubes in the i-cubing associated with I without any evaluations; these are the claimed ones. Using Chernoff's bound, an easy computation shows that $\Pr[Y\geqslant k/2]>1-1/n$, which finishes the induction. □

References

[1] A. Borodin and J.E. Hopcroft, Routing, merging, and sorting on parallel models of computation J. Comput. System Sci. 30 (1985) 130–145.

[2] M. Geréb–Graus and D. Krizanc, The complexity of parallel comparisons merging, Proc. 28th FOCS (1987) 195–202.

[3] N. Linial and M.E. Saks, Searching ordered structures, J. Algorithms 6 (1985) 86–103; (1972) 31–39.

[4] N. Linial and M.E. Saks, Information bounds are good for search problems on ordered data structures, Proc. 24th FOCS (1983) 473–475.

[5] J. von Neumann, Zur Theorie der Gesellschaftsspiele, Math. Ann. 100 (1928) 295–320.

[6] J. Shearer reported in [3].

[7] L.G. Valiant, Parallelism in comparison problems, SIAM J. Comput. 4 (1975) 348–355.

[8] A. Yao, Probabilistic computations: towards a unified measure of complexity, Proc. 18th FOCS (1977) 222–227.