SIAM J. COMPUT.
Vol. 18, No. 2, pp. 258-267, April 1989

005

FINDING AN APPROXIMATE MAXIMUM*

N. ALON†‡ AND Y. AZAR†

Abstract. Suppose that there are n elements from a totally ordered domain. The objective is to find, in a minimum possible number of rounds, an element that belongs to the biggest $n/2$, where in each round one is allowed to ask n binary comparisons. It is shown that $\log^* n + \Theta(1)$ rounds are both necessary and sufficient in the best algorithm for this problem.

Key words. searching, approximate maximum, parallel comparison algorithms

AMS(MOS) subject classification. 68E05

1. Introduction. Parallel comparison algorithms have received much attention during the last decade. The problems that have been considered include sorting [AA87], [AA88], [AAV86], [Ak85], [AKS83], [Al85], [AV87], [BT83], [BHe85], [HH81], [HH82], [Kn73], [Kr83], [Le84], [Pi86]; merging [BHo82], [HH82], [Kr83], [SV81]; selecting [AA88], [AKSS86a], [AP89], [Pi87], [Va75]; and approximate sorting [AA88], [AAV86], [AKSS86b], [BB87], [BR82]. The common model of computation considered is the parallel comparison model, introduced by Valiant [Va75], where only comparisons are counted. In this model, during each time unit (called a *round*) a set of binary comparisons is performed. The actual set of comparisons asked is chosen according to the results of the comparisons done in previous rounds. The objective is to solve the problem at hand, trying to minimize the number of comparison rounds as well as the total number of comparisons performed. Note that this model ignores the time corresponding to deducing consequences from comparisons performed, as well as communication and memory addressing time. However, in some situations this seems to be the relevant model. Moreover, any lower bound here applies to any comparison-based algorithm. There is an obvious, useful correspondence that associates each round of any comparison algorithm in the above parallel model with a graph whose vertices form the set of elements we have. The (undirected) edges of this graph are just the pairs compared during the round. The answer to each comparison corresponds to orienting the corresponding edge from the larger element to the smaller. Thus in each round we get an acyclic orientation of the corresponding graph, and the transitive closure of the union of the r oriented graphs obtained until round r represents the set of all pairs of elements whose relative order is known at the end of round r.

In many of the problems discussed so far in the parallel comparison model, the most interesting case is the one where the number n of elements is equal to the number of comparisons performed in each round. It is well known that in this case $\Theta(\log n)$ rounds are both necessary and sufficient for sorting. The lower bound follows trivially from the serial lower bound, and the upper bound follows from, e.g., the AKS sorting networks [AKS83]. As proved by Valiant, $\Theta(\log \log n)$ rounds are both necessary and sufficient for finding the maximum. The results of [AKSS86a] and [BHo82] show that the same $\Theta(\log \log n)$ bound holds for selecting and merging, respectively.

In the present paper we consider, motivated by the research on approximate sorting, another problem called the *approximate maximum* problem. This is the problem of finding, among n elements, an element whose rank belongs to the top $n/2$ ranks.

* Received by the editors January 10, 1988; accepted for publication (in revised form) May 18, 1988.

† Department of Mathematics and Computer Science, Sackler Faculty of Exact Sciences, Tel Aviv University, Ramat Aviv, Tel Aviv 69978, Israel.

‡ The research of this author was supported in part by an Allon Fellowship and by a grant from the United States-Israel Binational Science Foundation.

It is easy to show that in the serial comparison model this problem requires $n/2$ comparisons: only a constant factor better than the problem of finding the maximum. It is therefore rather surprising that with n comparisons in each round this problem can be solved much faster than that of finding the exact maximum in the same conditions. As it turns out, $\log^* n + \Theta(1)$ rounds are both necessary and sufficient for finding an approximate maximum among n elements, using n comparisons in each round. Moreover, the gap between the upper and lower bounds we obtain is only six rounds! The precise formulation of our result is the following. For $a \geq 1$, $k \geq 0$ define $a^{(k)}$ by $a^{(0)} = 1$ and $a^{(k)} = a^{a^{(k-1)}}$ for $k \geq 1$. Also define, as usual, $\log^* n = \min\{k : 2^{(k)} \geq n\}$. Let $r(n)$ denote the worst-case number of rounds of the best deterministic algorithm that finds an approximate maximum among n elements using n comparisons in each round. Our result is the following theorem.

THEOREM 1.1. *For every* $n \geq 2$,

$$\log^* n - 4 \leq r(n) \leq \log^* n + 2.$$

The upper bound here is not by explicit algorithm, as our algorithm uses certain random graphs. However, the known results about expanders easily supply (as in, e.g., [Al85], [Al86], [Pi87]) an explicit version of the algorithm, which will take about $\log^* n + 12$ rounds.

We note that our methods can be extended to the case where we have p comparisons in each round, as well as to the problem of finding an element whose rank is in the top εn ranks for all $p \geq 1$ and $1/n \leq \varepsilon \leq \frac{1}{2}$. With some additional work we can also obtain the corresponding results for approximate sorting. However, for the sake of simplicity we present here, in §§ 2 and 3, only the proof of Theorem 1.1 and only state the more general results (the detailed proofs of which will appear elsewhere) in the final § 4.

2. The upper bound. In this section we prove the upper bound, i.e., that using n processors we can find an element whose rank belongs to the top half of the ranks of the n elements in $\log^* n + 2$ rounds. Our algorithm uses some known results. The first is the algorithm of Valiant for finding the exact maximum. The others deal with properties of some random graphs (or explicit expanders). First we state a theorem from [Va75] and two lemmas from [Pi87].

THEOREM 2.1 [Va75]. *The maximum of* $n > 4$ *elements can be bound using* n *processors in* $\lceil \log \log n \rceil$ *rounds.* □

LEMMA 2.2 [Pi87]. *For every* m *and* a, *there is a graph with* m *vertices and* $2m^2 \log m/a$ *edges in which any two disjoint sets of* $a+1$ *vertices are joined by an edge.* □

LEMMA 2.3 [Pi87]. *If* m *elements are compared according to the edges of a graph in which any two disjoint sets of* $a+1$ *vertices are joined by an edge, then for every rank all but at most* $6a \log m$ *elements will be known to be too small or too large to have that rank.* □

For our algorithm we use Theorem 2.1 and a corollary of the last two lemmas. Note that Lemma 2.2 does not give an explicit construction of a graph with the specified properties; therefore, the algorithm seems to be nonexplicit. However, we can use some of the known results about explicit expanders (as in [Pi87], [Al85], [Al86]) to obtain an explicit version of our algorithm (with a slightly bigger additive constant). As the treatment for this case is analogous, we discuss here only the algorithm that uses Lemma 2.2.

PROPOSITION 2.4. *Assume we have* m *elements and* $p = 2m^2 \log m/a$. *Then, we can find in one round (using* p *comparisons) an element whose rank belongs to the top* $7a \log m$ *ranks of elements.*

Proof. Compare the elements according to the edges of the graph supplied by Lemma 2.2. By Lemma 2.3, all but at most $6a \log m$ elements will be known to be too small or too large to have rank $m - 7a \log m$. Therefore, at least $(7-6)a \log m$ elements will be known to belong to the top $7a \log m$ elements. □

For the description of our algorithm, we define the following function: $f(x) = 2^{x^{1/4}} \cdot x$, $x \geq 1$. f is a monotone increasing function and therefore $f^{-1}(y)$ exists (for $y \geq 2$) and is also a monotone (increasing) function. Define the following two sequences:

$b_0 = 2^{16}$ $b_{i+1} = f(b_i)$, $i \geq 0$ (We can easily check that all the $b_i's$ are integers),

$a_0 = n$, $a_{i+1} = f^{-1}(a_i)$, $i \geq 0$.

Define $k(n)$ by

$$k(n) = \min\{k : b_k \geq n\}. \tag{2.1}$$

By trivial induction using the monotonicity of f^{-1}, we get

$$a_i \leq b_{k-i} \quad \text{for } 0 \leq i < k(n). \tag{2.2}$$

Our algorithm is based on the following lemma, which is a consequence of the previous proposition.

LEMMA 2.5. *Assume we have $n \geq 2^{32}$ elements, partitioned into $m = \lceil n/f^{-1}(n) \rceil$ pairwise disjoint set, the ith having $t_i \leq \lceil f^{-1}(n) \rceil$ elements. Suppose that in each set we have an element that is smaller than at most εt_i elements in this set. Then we can find in one round using n processors an element smaller than at most $(\varepsilon + c/\sqrt{f^{-1}(n)})n$ elements among the n elements, where $c = 32$.*

Proof. Choose $a = \lceil 4m/\log^3 m \rceil$. Note that $m \geq 2^{16}$, $a \geq 4m/\log^3 m \geq 2^6$. Clearly, $(2m^2 \log m)/a \leq (m \log^4 m)/2 \leq n$, because by the definition of f: $\log^4 (n/f^{-1}(n)) = f^{-1}(n)$.

Thus we can use Proposition 2.4 for the m special elements with the n processors and find an element that belongs to the top $7a \log m$ elements out of the m elements. But

$$7a \log m = 7\left\lceil \frac{4m}{\log^3 m} \right\rceil \log m \leq 7\left(\frac{4m}{\log^3 m} + 1\right) \log m \leq 7 \frac{65}{64} \cdot \frac{4m}{\log^2 m} \leq \frac{30m}{\log^2 m}.$$

Therefore, the number of elements greater than this element is at most

$$\varepsilon n + \lceil f^{-1}(n) \rceil \frac{30m}{\log^2 m} \leq \varepsilon n + \left(1 + \frac{1}{2^{16}}\right) f^{-1}(n) \cdot \frac{30(1 + 1/2^{16}) n/f^{-1}(n)}{\log^2 (n/f^{-1}(n))}$$

$$\leq n\left(\varepsilon + \frac{32}{\log^2 (n/f^{-1}(n))}\right).$$

(In the last inequalities we used the facts that $f^{-1}(n)$, $m \geq 2^{16}$.)

Now $\log^2 (n/f^{-1}(n)) = \log^2 (2^{f^{-1}(n)^{1/4}}) = \sqrt{f^{-1}(n)}$. Hence we can find an element that is smaller than at most $(\varepsilon + (c/\sqrt{f^{-1}(n)}))n$, where $c = 32$. □

Now we can describe our algorithm and prove that it works.

THEOREM 2.6. *We can find an element the rank of which belongs to the top half of the ranks of $n \geq 2$ elements in $\log^* n + 2$ rounds using n processors.*

To obtain the last theorem, we prove by induction a more exact lemma.

LEMMA 2.7. *Let n be the number of elements and the number of processors. Then we can find in $k(n)+4$ rounds (where $k(n)$ is the number defined in (2.1)) an element that is smaller than at most $cn \sum_{i=0}^{k-2} 1/\sqrt{b_i}$ elements.*

Proof. We apply induction on n. The basic case is $n \leqq b_1$. In this case, we find the exact maximum using Theorem 2.1. Here $k \leqq 1$ so $cn \sum_{i=0}^{k-2} 1/\sqrt{b_i} = 0$, and we really find the exact maximum. To calculate the number of rounds we consider three cases. If $n \leqq 4$, four rounds are more than what is needed. If $n \leqq b_0$, then by Theorem 2.1 $\lceil \log\log n \rceil \leqq \lceil \log\log b_0 \rceil = \lceil \log\log 2^{16} \rceil = 4 = k(n)+4$. Otherwise $b_0 < n \leqq b_1$, $k(n) = 1$, and $\lceil \log\log n \rceil \leqq \lceil \log\log b_1 \rceil = \lceil \log\log 2^{32} \rceil = 5 = 1+4 = k(n)+4$. Assuming that the lemma is true for every $n' < n$, we prove it for n $(n > b_1 = 2^{32},\ k(n) \geqq 2)$. Split the n elements into $m = \lceil n/f^{-1}(n) \rceil$ pairwise disjoint sets, where the jth set has size $n_j \leqq \lceil f^{-1}(n) \rceil$. Assign to each set a number of processors equal to the size of the set. Now we can use the inductive assumption for each of the new sets, and find in $k'+4 \leqq k(\lceil f^{-1}(n) \rceil)+4$ rounds an element in each set, where the one in the jth set is smaller than at most $cn_j \sum_{i=0}^{k'-2} 1/\sqrt{b_i}$ elements in his set. By the definition of k, $n \leqq b_k$ so $f^{-1}(n) \leqq f^{-1}(b_k) = b_{k-1}$ and because the right-hand side is an integer $n_j \leqq \lceil f^{-1}(n) \rceil \leqq b_{k-1}$. Hence $k' \leqq k-1$; therefore we are allowed to have one more round to find the right element (among the m special elements). For that we use Lemma 2.5 and find an element smaller than at most

$$\left(\varepsilon + \frac{c}{\sqrt{f^{-1}(n)}}\right) n \leqq \left(\sum_{i=0}^{k-3} \frac{1}{\sqrt{b_i}} + \frac{1}{\sqrt{f^{-1}(n)}}\right) \cdot cn$$

other elements. Note that $b_{k-1} < n$ so $b_{k-2} \leqq f^{-1}(n)$, and hence the last expression is at most

$$cn\left(\sum_{i=0}^{k-3} \frac{1}{\sqrt{b_i}} + \frac{1}{\sqrt{b_{k-2}}}\right) = cn \sum_{i=0}^{k-2} \frac{1}{\sqrt{b_i}}.$$

Hence, we can find an element smaller than at most $cn \sum_{i=0}^{k-2} 1/\sqrt{b_i}$ elements among all the n elements in $k(n)+4$ rounds. This completes the proof of Lemma 2.7. □

To complete the proof of the main theorem, we just have to check the following two simple facts:

$$cn \sum_{i=0}^{\infty} \frac{1}{\sqrt{b_i}} \leqq \frac{1}{4} n \left(< \frac{1}{2} n \right), \tag{2.3}$$

$$k(n)+4 \leqq \log^* n + 2. \tag{2.4}$$

Inequality (2.3) is trivial, since $b_{i+1} = 2^{(b_i)^{1/4}} \cdot b_i \geqq 2^2 b_i$ or $1/\sqrt{b_{i+1}} \leqq \frac{1}{2}/\sqrt{b_i}$. Thus

$$c \sum_{i=0}^{\infty} \frac{1}{\sqrt{b_i}} \leqq \frac{2c}{\sqrt{b_0}} = \frac{64}{2^8} = \frac{1}{4}.$$

To prove (2.4) we need the following simple lemma:

LEMMA 2.8. *$b_i \geqq 2^8(2^{(i+2)})^4$ for every $i \geqq 0$.*

Proof. The proof is by induction on i. For $i = 0$, $b_0 = 2^{16} = 2^8(2^2)^4 = 2^8 \cdot (2^{(2)})^4$. Assuming the inequality holds for i, we prove it for $i+1$. Indeed, $b_{i+1} = 2^{(b_i)^{1/4}} \cdot b_i \geqq 2^{(2^8)^{1/4} \cdot 2^{(i+2)}} \cdot 2^8(2^{(i+2)})^4$ by the definition of b_{i+1} and by the inductive assumption. But the right-hand side is $\geqq 2^8 \cdot 2^{4 \cdot 2^{(i+2)}} = 2^8(2^{(i+3)})^4$, which completes the proof. □

Taking $i = \log^* n - 2$, we get that $b_i \geqq n$ and, therefore, $k(n) \leqq \log^* n - 2$. Thus, we have the complexity of the algorithm, which is $k(n)+4 \leqq \log^* n + 2$. This completes the proof of Theorem 2.6. □

3. The lower bound. It is convenient to establish our lower bound by considering the following (full information) game, called the *orientation game*, and played by two

players, the *graphs player* and the *order player*. Let V be a fixed set of n vertices. The game consists of *rounds*. In the first round the graphs player presents an undirected graph G_1 on V with at most n edges, and the order player chooses an acyclic orientation H_1 of G_1 and shows it to the graphs player, thus ending the first round. In the second round the graphs player chooses again, an undirected graph G_2 with at most n edges on V, and the order player gives it an acyclic orientation H_2, consistent with H_1 (i.e., the union of H_1 and H_2 is also acyclic), which he presents to the graphs player. The game continues in the same manner; in round i the graphs player chooses an undirected graph G_i with at most n edges on V, and the order player gives it an acyclic orientation H_i, such that the union $H_1 \cup \cdots \cup H_i$ is also acyclic. The game ends when, after, say, round r, there is a vertex v in V whose outdegree in the *transitive closure* of $H_1 \cup \cdots \cup H_r$ is at least $n/2$. The objective of the graphs player is to end the game as early as possible, and that of the order player is to end it as late as possible. The following fact states the (obvious) connection between the orientation game and the approximate maximum problem.

PROPOSITION 3.1. *The graphs player can end the orientation game in r rounds if and only if there is a comparison algorithm that finds an approximate maximum among n elements, using n comparisons in each round in r rounds.* □

In view of the last proposition and the results of the previous section, the graphs player can always end the orientation game in at most $\log^* n + O(1)$ rounds. A proof of existence of a strategy for the order player that enables him to avoid ending the orientation game in r rounds implies that $r+1$ is a lower bound for the time complexity of the approximate maximum problem. The next proposition is our main tool for establishing the existence of such a strategy for $r = \log^* n - 5$.

PROPOSITION 3.2. *There exists a strategy for the order player to maintain, for every $d \geqq 1$, the following property $P(d)$ of the directed acyclic graph constructed during the game.*

Property $P(d)$. Let $H(d) = H_1 \cup \cdots \cup H_d$ be the union of the oriented graphs constructed in the first d rounds. Then there is a subset $V_0 \subseteq V$ of size at most $|V_0| \leqq n/8 + n/16 + \cdots + n/2^{d+2}$ and a proper $D = 2000^{(d)}$-vertex-coloring of the induced subgraph of $H(d)$ on $V - V_0$ with color classes $V_1, V_2, \cdots, V_D$ (some of which may be empty), such that for each $i > j \geqq 1$ and each $v \in V_i$, v has at most 2^{i-j-2} neighbors in V_j. Furthermore, for every $i > j \geqq 0$ any edge of $H(d)$ that joins a member of V_i to a member of V_j is directed from V_i to V_j.

Proof. We apply induction on d. For $d = 1$, the graph $G_1 = (V, E_1)$ constructed by the graphs player has at most n edges. Let V_{00} be the set of all vertices in V whose degree is at least 32. Clearly,

$$|V_{00}| \leqq n/16. \tag{3.1}$$

Put $U = V - V_{00}$ and let K be the induced subgraph of G_1 on U. As the maximum degree in K is at most 31, K has, by a standard, easy result from extremal graph theory (see, e.g., [Bo78, p. 222]) a proper vertex-coloring by 32 colors and hence, certainly, a proper vertex coloring by 2000 colors. Let $U_1, U_2, \cdots, U_{2000}$ be the color classes. For every vertex u of K, let $N(u)$ denote the set of all its neighbors in K. For a permutation π of $1, 2, \cdots, 2000$ and any vertex n of K, define the π-degree $d(\pi, u)$ of u as follows. Let i satisfy $u \in U_{\pi(i)}$; then $d(\pi, u) = \sum_{j=1}^{i-1} |N(u) \cap U_{\pi(j)}|/2^{i-j}$. We claim that the expected value of $d(\pi, u)$ over all permutations π of $\{1, \cdots, 2000\}$, is at most $31/2000$. Indeed, for a random permutation π the probability that a fixed neighbor v of u contributes $1/2^r$ to $d(\pi, u)$ is at most $1/2000$ for every fixed $r > 0$. Hence, each neighbor contributes to this expected value at most $1/2000 \sum_{r>0} 1/2^r = 1/2000$ and the desired result follows, since $|N(u)| \leqq 31$.

Now consider the sum $\sum_{u\in U} d(\pi, u)$. The expected value of this sum (over all π's) is at most $31/2000|U|$, by the preceding paragraph. Hence, there is a fixed permutation σ such that $\sum_{u\in U} d(\sigma, u) \leqq 31/2000|U|$. Put $V_{01} = \{u \in U | d(\sigma, u) > \frac{1}{4}\}$. Clearly,

$$|V_{01}| \leqq \frac{4 \cdot 31}{2000}|U| \leqq \frac{|U|}{16} \leqq \frac{n}{16}.$$

Define $V_0 = V_{00} \cup V_{01}$, $W = U - V_{01} = V - V_0$. The last inequality together with (3.1) gives

$$|V_0| \leqq n/8.$$

Let F be the induced subgraph of G_1 on W and define $V_i = U_{\sigma(i)} \cap W$ $(1 \leqq i \leqq 2000)$. The V_i's clearly form a proper vertex coloring of F. Also, for every i, $1 \leqq i \leqq 2000$ and every $v \in V_i$

$$\sum_{j=1}^{i-1} \frac{|N(v) \cap V_j|}{2^{i-j}} < \frac{1}{4}$$

and hence v has at most 2^{i-j-2} neighbors in V_j for each j, $1 \leqq j < i$. Let H_1 be any acyclic orientation of G_1 in which all the edges that join a member of V_i to a member of V_j, where $i > j \geqq 0$, are directed from V_i to V_j (the edges inside V_0 can be directed in an arbitrary acyclic manner). Clearly $H(1) = H_1$ satisfies the property $P(1)$. Thus, the order player can orient G_1 according to H_1. This completes the proof of the case $d = 1$. □

Continuing the induction, we now assume that $H(r)$ has property $P(r)$ for all $r < d$, and prove that the order player can always guarantee that $H(d)$ will have property $P(d)$. We start by proving the following simple lemma.

LEMMA 3.3. *Let F be a directed acyclic graph with a proper g-vertex coloring with color classes $W_1, W_2, \cdots, W_g$. Suppose that for each $g \geqq i > j \geqq 1$ and each $v \in W_i$, v has at most 2^{i-j-2} neighbors in W_j, and that every edge of F whose ends are in W_i and W_j for some $i > j$ is directed from W_i to W_j. Then the outdegree of every vertex of F in the transitive closure of F is smaller than 4^g.*

Proof. Let v be an arbitrary vertex of F. The outdegree of v in the transitive closure of F is obviously smaller than or equal to the total number of directed paths in F that start from v. Suppose $v \in W_i$. Each such directed path must be of the form $v, v_{i_2}, v_{i_3}, \cdots, v_{i_r}$, where $i > i_2 > i_3 > \cdots i_r \geqq 1$, $v_{i_2} \in W_{i_2}, \cdots, v_{i_r} \in W_{i_r}$. There are 2^{i-1} possibilities for choosing $i_2, i_3, \cdots, i_r$. Also, as each vertex of the path is a neighbor of the previous one, there are at most 2^{i-i_2-2} possible choices for v_{i_2}, $2^{i_2-i_3-2}$ possible choices for v_{i_3} (for each fixed choice of v_{i_2}), etc. Hence, the total number of paths is at most $2^{i-1} \cdot 2^{i-i_2-2} \cdot 2^{i_2-i_3-2} \bullet \cdots \bullet 2^{i_{r-1}-i_r-2} < 2^g \cdot 2^{i-i_r} < 4^g$. This completes the proof of the lemma. □

Returning to the proof of Proposition 3.2, recall that $d \geqq 2$ and that by the induction hypothesis $H(d-1)$ has property $P(d-1)$. Thus, there is a subset $V_0 \subseteq V$ satisfying

$$|V_0| \leqq \frac{n}{8} + \frac{n}{16} + \cdots + \frac{n}{2^{d+1}} \tag{3.2}$$

and a proper $D = 2000^{(d-1)}$-vertex-coloring of the induced subgraph of $H(d-1)$ on $V - V_0$ with color classes $V_1, V_2, \cdots, V_D$ satisfying the conditions of property $P(d-1)$. Put $U = V - V_0$, let F be the induced subgraph of $H(d-1)$ on U, and let $T = (U, E(T))$ be the transitive closure of F. Let $G_d = (V, E_d)$ be the graph constructed by the graphs

player in round number d. Let $\bar{V}_{00}$ be the set of all vertices in U whose degree in G_d is at least $2^{d+4} \cdot 4^D$ and define

$$V_{00} = \bar{V}_{00} \cup \{u \in U : \exists v \in \bar{V}_{00} \text{ with } (v, u) \in E(T)\}.$$

Since G_d has at most n edges, $|\bar{V}_{00}| \leqq n/(2^{d+3} \cdot 4^D)$. Also, by Lemma 3.3, the outdegree of each $v \in \bar{V}_{00}$ in T is at most $4^D - 1$. Hence

$$|V_{00}| \leqq n/2^{d+3}. \tag{3.3}$$

Let $\bar{G}$ be the induced subgraph of G_d on $U - V_{00}$. Then the maximum degree in $\bar{G}$ is smaller than $2^{d+4} \cdot 4^D$. For each i, $1 \leqq i \leqq D$, let $\bar{G}^i$ denote the induced subgraph of $\bar{G}$ on $(U - V_{00}) \cap V_i$. As each $\bar{G}^i$ is a subgraph of $\bar{G}$, it has a proper vertex coloring with $2^{d+4} \cdot 4^D$ colors. For each i, $1 \leqq i \leqq D$, fix a proper n_i-vertex-coloring of $\bar{G}^i$ with color classes $U_{N_i+1}, U_{N_i+2}, \cdots, U_{N_i+n_i}$ (some of which may be empty), where $N_i = \sum_{j=1}^{i-1} n_j$ and

$$n_i \geqq 100 \cdot 2^{2d+7} \cdot 16^D \quad \text{for each } 1 \leqq i \leqq D \text{ and } \sum_{i=1}^{D} n_i = 2000^D. \tag{3.4}$$

(Note that since $D = 2000^{(d-1)}$, $d \geqq 2$, there is such a choice for the n_i's.) For every vertex u of $\bar{G}$, let $N(u)$ denote the set of all its neighbors in $\bar{G}$. Let us call a permutation π of $1, 2, 3, \cdots, \sum_{i=1}^{D} n_i$ *legal* if it maps each set of the form $\{N_i + 1, \cdots, N_i + n_i\}$ into itself (and only permute the elements inside these sets among themselves). For any vertex u of $\bar{G}$ and any legal permutation π, define the π-degree $d(\pi, u)$ as follows; let k satisfy $u \in U_{\pi(k)}$, then

$$d(\pi, u) = \sum_{j=1}^{k-1} |N(u) \cap U_{\pi(j)}|/2^{k-j}.$$

We claim that the expected value of $d(\pi, u)$, over all $\prod_{i=1}^{D} n_i!$ legal permutations, is at most $|N(u)|/\min_{1 \leqq i \leqq D} n_i \leqq 1/(100 \cdot 2^{d+3} \cdot 4^D)$. Indeed, consider a fixed neighbor v of u. If v belongs, as does u, to the same graph $\bar{G}^k$, then the probability that for a random legal permutation π, v will contribute $1/2^r$ to $d(\pi, u)$ is at most $1/n_k$, for each fixed $r > 0$. Otherwise, it is easy to check that this probability is even smaller. Hence, each neighbor contributes to this expected value at most $1/n_k \sum_{r>0} 1/2^r = 1/n_k$, and the claim follows.

Consider now the sum $\sum d(\pi, u)$, where u ranges over all vertices of $\bar{G}$. The expected value of this sum (over all legal permutations π) is at most $|V(\bar{G})|/(100 \cdot 2^{d+3} \cdot 4^D) \leqq n/(100 \cdot 2^{d+3} \cdot 4^D)$. Hence, there is a fixed legal permutation σ such that $\sum_{u \in V(\bar{G})} d(\sigma, u) \leqq n/(100 \cdot 2^{d+3} \cdot 4^D)$. Define $\bar{V}_{01} = \{u \in V(\bar{G}) : d(\sigma, u) > 1/100\}$ and $V_{01} = \bar{V}_{01} \cup \{u \in V(\bar{G}) : \exists v \in \bar{V}_{01} \text{ with } (v, u) \in E(T)\}$. Clearly, $|\bar{V}_{01}| \leqq n/(2^{d+3} 4^D)$ and, hence, by Lemma 3.3,

$$|V_{01}| \leqq n/2^{d+3}. \tag{3.5}$$

Put $V_0' = V_0 \cup V_{00} \cup V_{01}$, $W = V - V_0$. By (3.2), (3.3), and (3.5)

$$|V_0'| \leqq \frac{n}{8} + \frac{n}{16} + \cdots + \frac{n}{2^{d+2}}.$$

Let $\tilde{G}$ be the induced subgraph of $\bar{G}$ on W, and define $V_i' = U_{\sigma(i)} \cap W$ $(1 \leqq i \leqq 2000^D = 2000^{(d)})$. The sets V_i' clearly form a proper vertex coloring of $\tilde{G}$. Moreover, as each U_k is an independent set in $H(d-1)$, the sets V_i' actually form a proper vertex coloring of $H(d-1)$, as well. Moreover, for every i, $1 \leqq i \leqq 2000^{(d)}$, every $v \in V_i'$ satisfies

$$\sum_{j=1}^{i-1} \frac{|N(v) \cap V_j'|}{2^{i-j}} \leqq \frac{1}{100},$$

where $N(v)$ is the set of all neighbors of v in $\bar{G}$. Thus, for each fixed j, $1 \leqq j < i$, v has

at most $2^{i-j}/100$ neighbors in V'_j. Let H_d be any acyclic orientation of the edges of G_d obtained by orienting all the edges that join a member of V'_i and a member of V'_j, where $i>j\geq 0$, from V'_i to V'_j. The edges inside V'_0 are oriented in an arbitrary acyclic order consistent with the order given on $H(d-1)$. Notice that all the edges of $H(d-1)$ that do not lie inside V'_0 are also oriented from V'_i to V'_j with $i>j\geq 0$. In order to show that $H(d)=H(d-1)\cup H_d$ has the property $P(d)$, it remains to check that for every $i>j\geq 1$, every $v\in V'_i$ has at most 2^{i-j-2} neighbors in V'_j. By the construction, v has at most $2^{i-j}/100$ neighbors in V'_j in the new graph H_d. Recall that each V'_i is a subset of one of the sets V_k corresponding to the graph $H(d-1)$. Suppose $V'_i\subseteq V_k$, $V'_j\subseteq V_l$. Then $k\geq l$. If $l=k$ or $l=k-1$, then, since v has at most $[2^{k-l-2}]=0$ neighbors in V_l in the graph $H(d-1)$, it follows that in $H(d)$ v has at most $2^{i-j}/100\leq 2^{i-j-2}$ neighbors in V'_j, as needed. If $l\leq k-2$, observe that our construction implies that

$$(i-j)\geq(k-l-1)\min_{1\leq i\leq D} n_i\geq(k-l-1)\cdot 100\cdot 2^{2d+7}\cdot 16^D>(k-l)\cdot 100\geq 200.$$

Thus, in this case, the total number of neighbors of v in V'_j is at most $2^{i-j}/100+2^{k-l-2}\leq 2^{i-j}/100+2^{(i-j)/100}\leq 2^{i-j-2}$.

We conclude that the order player can orient G_d according to H_d, and maintain the property $P(d)$ of the graph $H(d)=H(d-1)\cup H_d$. This completes the induction and the proof of Proposition 3.2. □

The main result of this section, stated in Theorem 3.5 below, is an easy consequence of Proposition 3.2 and the following simple lemma.

LEMMA 3.4. *For every* $d\geq 1$, $2^{(d+3)}\geq 32\cdot 2000^{(d)}$.

Proof. We apply induction on d. For $d=1$ the inequality is trivial, as $2^{(4)}=2^{16}>64{,}000=32\cdot 2000^{(1)}$. Assuming it holds for $d-1$, we prove it for $d\geq 2$. By assumption $2^{(d+2)}\geq 32\cdot 2000^{(d-1)}$. Hence $2^{(d+3)}=2^{2^{(d+2)}}\geq 2^{32\cdot 2000^{(d-1)}}=(2^{32})^{2000^{(d-1)}}\geq(2^{21}\cdot 2000)^{2000^{(d-1)}}=(2^{21})^{2000^{(d-1)}}\cdot(2000)^{2000^{(d-1)}}>32\cdot(2000)^{(d)}$. □

THEOREM 3.5. *The order player can avoid ending the orientation game during the first* $\log^* n-5$ *rounds. Hence, by Proposition* 3.1, *the time required for finding an approximate maximum among n elements using n comparisons in each round is at least* $\log^* n-4$.

Proof. Clearly, we may assume that $\log^* n-5\geq 0$. By Proposition 3.2, the order player can maintain the property $P(d)$ for each of the graphs $H(d)$ constructed during the algorithm. Notice that by Lemma 3.3, the outdegree of every vertex in the transitive closure of a graph that satisfies $P(d)$ is at most $4^D+n/8+n/16+\cdots+n/2^{d+2}<4^D+n/4$, where $D=2000^{(d)}$. Thus, it follows that if $4^{2000^{(r)}}\leq n/4$, then the graphs player can keep playing for at least $r+1$ rounds. Therefore, by Lemma 3.4, the assertion of the theorem will follow if for $r=\log^* n-5$ the inequality $4^{2^{(r+3)}/32}\leq n/4$ holds. Since for $r>0$ $4\cdot 4^{2^{(r+3)}/32}<2^{(r+4)}$, this follows immediately from the definition of $\log^* n$. □

4. Extensions and related results. In this section we merely state, without proof, several extensions of the results of this paper and several related results. The proofs of these results combine the methods used here with some new ideas, somewhat similar to ones used in [AV87], [AAV86], [AP89]. The detailed proofs are somewhat complicated and will appear somewhere else.

For integers $n\geq 2$ and p, $1\leq p\leq\binom{n}{2}$, and for a real number ε, $1/n\leq\varepsilon\leq\frac{1}{2}$, let $r(n,p,\varepsilon)$ denote the time complexity of the best deterministic comparison algorithm that finds, among n elements, an element whose rank belongs to the top εn ranks, using p comparisons in each round. Clearly $r(n,n,\frac{1}{2})$ is just the function $r(n)$ discussed in this paper. For $\varepsilon=1/n$, the problem is that of finding the exact maximum, and the case $p=1$ corresponds to serial algorithms. We can prove the following theorem.

THEOREM 4.1. *For all admissible* n, p, ε,

$$r(n, p, \varepsilon) = \Theta\left(\frac{n}{p} + \log \frac{\log (1/\varepsilon)}{\log (2 + p/n)} + \log^* n - \log^* \left(1 + \frac{p}{n}\right)\right).$$

Thus for all n, $p \leqq 2n$, ε

$$r(n, p, \varepsilon) = \Theta\left(\frac{n}{p} + \log \log \frac{1}{\varepsilon} + \log^* n\right)$$

and for all n, $p \geqq 2n$, ε

$$r(n, p, \varepsilon) = \Theta\left(\log \frac{\log 1/\varepsilon}{\log (p/n)} + \log^* n - \log^* \left(\frac{p}{n}\right)\right).$$

For $\varepsilon = 1/n$ this theorem reduces to Valiant's result about finding the maximum. For $\varepsilon = \frac{1}{2}$, $p = n$ this reduces to our Theorem 1.1 (with a somewhat cruder estimate).

Next we consider approximate sorting. For $n \geqq 2$, $1 \leqq p \leqq \binom{n}{2}$, and $2/n^2 \leqq \varepsilon \leqq \frac{1}{2}$, let $a(n, p, \varepsilon)$ denote the time complexity of the best deterministic comparison algorithm that uses p comparisons in each round and finds, given n elements, all the order relations between pairs but at most $\varepsilon\binom{n}{2}$. The results of [BR82], [AA88], [AKSS86b], [BB87] deal with the minimum p for which $a(n, p, \varepsilon) = 1$ for some $\varepsilon = o(1)$. Notice that a precise determination of $a(n, p, \varepsilon)$ contains all the known results about deterministic comparison sorting or approximate sorting algorithms. We can prove the following result, determining $a(n, p, \varepsilon)$, up to a constant factor, for all possible n, p, and ε.

THEOREM 4.2. *For all admissible* n, p, ε

$$a(n, p, \varepsilon) = \Theta\left(\frac{\log 1/\varepsilon}{\log (1 + p/n)} + \log^* n - \log^*\left(1 + \frac{p}{n}\right)\right).$$

Thus, for $p \leqq 2n$, $a(n, p, \varepsilon) = \Theta(n \log (1/\varepsilon)/p + \log^* n)$ *and for* $p \geqq 2n$, $a(n, p, \varepsilon) = \Theta(\log (1/\varepsilon)/\log (p/n) + \log^* n - \log^* (p/n))$.

For $\varepsilon = 2/n^2$ this theorem corresponds to sorting and gives the known $\Theta(\log n/\log(1 + p/n))$ bound (which is $\Theta(n \log n/p)$ for $p \leqq 2n$ and is $\Theta(\log n/\log (p/n))$ for $p \geqq 2n$), (see [AV87], [AAV86]). Notice that for $p = n$ and for any $\varepsilon > 1/2^{2^{\log^* n}}$, $a(n, n, \varepsilon) = \Theta(\log^* n)$. As shown in § 3, $\Omega(\log^* n)$ rounds are required (with $p = n$), even if we wish to find one element known to be greater than $n/2$ others. By the last equality, $O(\log^* n)$ rounds are already sufficient to get almost all the order relations between pairs.

Finally, we consider the problem of approximate merging. In this case the results and the methods are simpler and similar to the methods of [Va75], [BHo82]. For n, $1 \leqq p \leqq n^2$ and $1/2n^2 \leqq \varepsilon \leqq \frac{1}{2}$, let $m(n, p, \varepsilon)$ denote the time complexity of the best comparison merging algorithm that uses p comparisons in each round and finds, given two sorted lists, each of size n, all the order relation between pairs but at most εn^2.

The results of [Va75], [BHo82] deal with full merging, i.e., the case $\varepsilon < 1/n^2$. We can prove the following theorem that determines $m(n, p, \varepsilon)$, up to a constant factor, for all admissible n, p, ε.

THEOREM 4.3. *For all admissible* n, p *and* $1/n \leqq \varepsilon \leqq \frac{1}{2}$,

$$m(n, p, \varepsilon) = \Theta\left(\frac{1}{\varepsilon p} + \log \frac{\log 1/\varepsilon}{\log (2 + \varepsilon p)}\right).$$

Thus for $p \leqq 2/\varepsilon$ $m(n, p, \varepsilon) = \Theta(1/(\varepsilon p) + \log \log 1/\varepsilon)$ and for $p \geqq 2/\varepsilon$ $m(n, p, \varepsilon) = \Theta(\log (\log 1/\varepsilon/\log \varepsilon p))$. For the case $\varepsilon \leqq 1/n$, the bounds are the same as for $\varepsilon = 1/n$

(up to a constant factor), which are the same bounds as for exact merging: $\Theta(n/p + \log(\log n/\log(2+p/n)))$.

Acknowledgment. We thank N. Pippenger, who brought the problem of finding an approximate maximum to our attention.

REFERENCES

[AA87] N. ALON AND Y. AZAR, *The average complexity of deterministic and randomized parallel comparison-sorting algorithms*, in Proc. 28th Annual IEEE Symposium on Foundations of Computer Science, Los Angeles, CA, 1987, pp. 489–498; SIAM J. Comput., 17 (1988), pp. 1178–1192.

[AA88] ———, *Sorting, approximate sorting and searching in rounds*, SIAM J. Discrete Math., 1 (1988), pp. 261–276.

[AAV86] N. ALON, Y. AZAR, AND U. VISHKIN, *Tight complexity bounds for parallel comparison sorting*, in Proc. 27th Annual IEEE Symposium on Foundations of Computer Science, Toronto, Ontario, Canada, 1986, pp. 502–510.

[Ak85] S. AKL, *Parallel Sorting Algorithm*, Academic Press, New York, 1985.

[AKSS86a] M. AJTAI, J. KOMLÓS, W. L. STEIGER, AND E. SZEMERÉDI, *Deterministic selection in $O(\log\log n)$ parallel time*, Proc. 18th Annual ACM Symposium on Theory of Computing, Berkeley, CA, 1986, pp. 188–195.

[AKSS86b] ———, *Almost sorting in one round*, Adv. Comput. Res., to appear.

[AKS83] M. AJTAI, J. KOMLÓS, AND E. SZEMERÉDI, *An $O(n \log n)$ sorting network*, in Proc. 15th Annual ACM Symposium in Theory of Computing, 1983, pp. 1–9; *Sorting in $c \log n$ parallel steps*, Combinatorica, 3(1983), pp. 1–19.

[Al85] N. ALON, *Expanders, sorting in rounds and superconcentrators of limited depth*, in Proc. 17th Annual ACM Symposium on Theory of Computing, Providence, RI, 1985, pp. 98–102.

[Al86] ———, *Eigenvalues, geometric expanders, sorting in rounds and Ramsey Theory*, Combinatorica, 6(1986), pp. 207–219.

[AP89] Y. AZAR AND N. PIPPENGER, *Parallel selection*, Discrete Appl. Math., to appear.

[AV87] Y. AZAR AND U. VISHKIN, *Tight comparison bounds on the complexity of parallel sorting*, SIAM J. Comput., 3 (1987), pp. 458–464.

[BB87] B. BOLLOBÁS AND G. BRIGHTWELL, *Graphs whose every transitive orientation contains almost every relation*, Israel J. Math., 59 (1987), pp. 112–128.

[Bo78] B. BOLLOBÁS, *Extremal Graph Theory*, Academic Press, London, New York, 1978.

[BR82] B. BOLLOBÁS AND M. ROSENFELD, *Sorting in one round*, Israel J. Math., 38 (1981), pp. 154–160.

[BT83] B. BOLLOBÁS AND A. THOMASON, *Parallel sorting*, Discrete Appl. Math., 6 (1983), pp. 1–11.

[BHe85] B. BOLLOBÁS AND P. HELL, *Sorting and Graphs*, in Graphs and Orders, I. Rival, ed., D. Reidel, 1985, Boston, MA, pp. 169–184.

[BO86] B. BOLLOBÁS, *Random Graphs*, Academic Press, New York, 1986, Chap. 15.

[BHo82] A. BORODIN AND J. E. HOPCROFT, *Routing, merging and sorting on parallel models of computation*, in Proc. 14th Annual ACM Symposium on Theory of Computing, San Francisco, CA, 1982, pp. 338–344.

[HH80] R. HÄGGKVIST AND P. HELL, *Graphs and parallel comparison algorithms*, Congr. Numer., 29 (1980), pp. 497–509.

[HH81] ———, *Parallel sorting with constant time for comparisons*, SIAM J. Comput., 10 (1981), pp. 465–472.

[HH82] ———, *Sorting and merging in rounds*, SIAM J. Algebraic Discrete Methods, 3 (1982), pp. 465–473.

[Kn73] D. E. KNUTH, *The Art of Computer Programming, Vol. 3: Sorting and Searching*, Addison-Wesley, Reading, MA, 1973.

[Kr83] C. P. KRUSKAL, *Searching, merging and sorting in parallel computation*, IEEE Trans. Comput., 32 (1983), pp. 942–946.

[Le84] F. T. LEIGHTON, *Tight bounds on the complexity of parallel sorting*, in Proc. 16th Annual ACM Symposium on Theory of Computing, Washington, DC, 1984, pp. 71–80.

[Pi87] N. PIPPENGER, *Sorting and selecting in rounds*, SIAM J. Comput., 6 (1987), pp. 1032–1038.

[SV81] Y. SHILOACH AND U. VISHKIN, *Finding the maximum, merging and sorting in a parallel model of computation*, J. Algorithms, 2 (1981), pp. 88–102.

[Va75] L. G. VALIANT, *Parallelism in comparison problems*, SIAM J. Comput., 4 (1975), pp. 348–355.