Finding the maximum and minimum elements with one lie

Dániel Gerbner
Rényi Institute of Mathematics, Hungarian Academy of Sciences
Reáltanoda utca 13-15., 1053 Budapest, Hungary
gerbner@renyi.hu

Dömötör Pálvölgyi*
Department of Computer Science, Eötvös University
Pázmány Péter sétány 1/c, 1117 Budapest, Hungary
dom@cs.elte.hu

Balázs Patkós†
Department of Computer Science, Eötvös University
Pázmány Péter sétány 1/c, 1117 Budapest, Hungary
patkosb@cs.elte.hu

Gábor Wiener‡
Department of Computer Science and Information Theory,
Budapest University of Technology
1117 Budapest, Magyar tudósok körútja 2.
wiener@cs.bme.hu

*Supported by OTKA NK 67867

†Supported by OTKA NK 67867

‡Supported by the Hungarian National Research Fund and by the Hungarian National Office for Research and Technology. (Grant number: OTKA 67651)

Abstract

In this paper we deal with the problem of finding the smallest and the largest elements of an ordered set of size n using pairwise comparisons if one of the comparisons might be erroneous and prove a conjecture of Aigner stating that the minimum number of comparisons needed is $\frac{87n}{32} + c$ for some constant c. We also address some related problems.

Keywords: Searching, Erroneous information, Sorting
AMS Subject Classification: 94A50

1 Introduction

Search problems when some of the answers may be lies have been studied by various researchers (for a list of references see the surveys by Deppe [2] and Pelc [3]). Three models have attracted the most attention. In the first model a fixed number k of the answers may be false, in the second model a fixed proportion p of the answers may be erroneous, while in the third model every answer turns out to be a lie with probability p independently from all other answers.

The problem of finding the maximum or the minimum element is solved in [5]. Aigner in [1] considered the problem of finding both the maximum and minimum elements (which we will later also refer to as the extremal elements). He obtained asymptotically tight results for the second model, but only upper and lower bounds for the first model. In this paper we address the problem of finding the extremal elements of an ordered set of size n using pairwise comparisons in the first model. That is, we are given distinct numbers $x_1, x_2, ..., x_n$ along with a positive integer k and at each step of our algorithm, we can ask whether $x_i < x_j$ or $x_i > x_j$ holds for any $i \neq j$, and during the process at most k answers might turn out to be false.

If all answers have to be correct then the minimum number of comparisons needed is $\lceil \frac{3n}{2} \rceil - 2$ (see [4]) as can be easily seen using the following observation: if at the beginning of the algorithm we give each x_i a red and a blue pebble and after each comparison we remove the red pebble of the smaller element and the blue pebble of the larger element (if they still possess it), then the algorithm terminates if and only if we have only one element having a red pebble (the largest element) and another element having a blue pebble (the smallest element).

One could think that if k erroneous answers are allowed, then all one has to do is to use $k + 1$ red and blue pebbles instead of one, as if an element has been said to be the larger one in $k+1$ comparisons, then in at least one of these it was indeed the larger and hence cannot be the smallest element. Unfortunately an if-and-only-if-type statement does not hold now, but before explaining this let us introduce some notations and the "soccer terminology".

The element x_i will be called the ith *team*, a comparison will be called a *match* which is a *win* for the team of the larger element and a *loss* for the team of the smaller element. We will also say that x_i beats x_j if the match between x_i and x_j ended with a win for x_i. For a team x, let $w(x)$ denote the number of wins of x, and let $l(x)$ be the number of losses of x. In the case of k erroneous answers, we put $wl_k(x) = (\max\{k+1-w(x), 0\}, \max\{k+1-l(x), 0\})$, the number of wins and losses that are still needed in order to prove that x is neither the maximal nor the minimal element. We also use the notation $a^+ = \max\{a, 0\}$, so for example we can write $wl_k(x) = ((k+1-w(x))^+, (k+1-l(x))^+)$.

Let us define the *championship graph* G as follows: the vertex set of this directed multigraph is the set of teams, and for each match a directed edge is given to the graph oriented from the loser toward the winner.

If the championship graph contains a directed cycle, then we know that for one of the matches corresponding to the edges of the cycle we were given an erroneous result. Therefore if we forget about the results corresponding to the edges of the cycle, we know that among the other results (including the forthcoming ones) there can be at most $k-1$ lies. This is the reason why the above-mentioned if-and-only-if-type statement is not true in this case. However, the obvious direction still holds as stated in the following claim.

Claim 1.1. *If at most k erroneous answers are allowed, then a team x with $wl_k(x) = (0, 0)$ cannot be the maximum or the minimum element.*

Corollary 1.2. *Suppose that at most k erroneous answers are allowed and we have exactly two elements x with $wl_k(x) \neq (0, 0)$. If for both of these elements either the number of losses or the number of wins is k, then they are the extremal elements.*

Corollary 1.2 will serve to prove upper bounds on the number of comparisons needed to find the extremal elements in different models. To provide lower bounds we will use the notion of an *Adversary.* A strategy of an Adversary is a function that tells us what the Adversary answers for a query in the view of previous queries and answers. To obtain lower bounds we will have to prove that there exists an Adversary's strategy that answers any sequence of queries in such a way that until at least D comparisons asked, no strategy of queries determines both the maximum and the minimum elements. How can one guarantee that a sequence of queries and answers does not determine the extremal elements? Observe that a championship graph may consist of true answers if and only if it is acyclic. Furthermore, it is obvious that if in a directed acyclic graph G one changes the orientation of all incoming (outgoing) edges that are adjacent to a fixed vertex v, then the resulting graph G' is also acyclic. These two easy observations give us the following Corollary.

Corollary 1.3. *If at most k erroneous answers are allowed, and if there exists a strategy of an Adversary that can assure that after D queries the championship graph is acyclic and there exists at least two vertices either both with in-degree at most k or both with out-degree at most k, then the number of comparisons needed to find the maximum and the minimum elements is at least $D+1$.*

The rest of the paper is organized as follows: in Section 2 we present a simple (and not optimal) algorithm and bound the number of comparisons it uses for arbitrary k. This algorithm was described already by Aigner in [1], but Aigner's proof for the number of comparisons used in the algorithm is somewhat different from ours and the method of our proof is used later to give an almost matching lower bound in the case $k = 1$. In Section 3, we address the original problem with at most one lie allowed and prove the following main result of the present paper.

Theorem 1.4. *For the minimum number $M(n)$ of comparisons needed to find the extremal elements among n elements if there might be one erroneous answer, we have*

$$\lceil \frac{87n}{32} \rceil - 3 \leq M(n) \leq \lceil \frac{87n}{32} \rceil + 12.$$

Aigner in [1] stated the upper bound and conjectured it to be optimal, thus Theorem 1.4 verifies his conjecture. In Section 4 we gather some open problems and concluding remarks.

2 Algorithm for arbitrary k

In this section we give an algorithm that does not use the possible additional information that might be gained from the existence of directed cycles in the championship graph. First let us introduce a slightly different version of the problem, when the algorithm "cannot use" this additional information.

We are given n teams $x_1, ..., x_n$ and every team x_i possesses an ordered pair $wl_k(x_i) = (a_i, b_i)$. At the beginning of the procedure $a_i = b_i = k+1$ for all

$1 \le i \le n$. A query in this version is a pair of teams $\{x_i, x_j\}$ and there are two possible "answers": either $wl_k(x_i) = ((a_i - 1)^+, b_i), wl_k(x_j) = (a_j, (b_j - 1)^+)$ or $wl_k(x_i) = (a_i, (b_i - 1)^+), wl_k(x_j) = ((a_j - 1)^+, b_j)$ but there always must be a team with a positive a_i and another one with a positive b_i. The process ends when all but two (a_i, b_i) pairs are $(0, 0)$ and from the remaining two, at least one has a zero a_i or b_i. Denote the minimum number of queries needed to obtain this situation by $N(k, n)$. In the remainder of this section we will prove the following theorem.

Theorem 2.1.

$$N(k, n) = (k+1)(1+\binom{2(k+1)}{k+1}2^{-2(k+1)})n+\Theta_k(1) = (k+\Theta(\sqrt{k}))n+\Theta_k(1).$$

It is clear that any upper bound on $N(k, n)$ is also an upper bound for the number of comparisons needed in the original problem, since every algorithm that solves this problem, also solves the original one because of Corollary 1.2.

Proof: We define a symmetric potential function $p : \mathbb{N}\times\mathbb{N} \to \mathbb{N}$. Let $p(a, 0) = p(0, a) = a$ for any $a \in \mathbb{N}$ and let us define the other values recursively by the equation

$$2p(a, b) = p(a - 1, b) + p(a, b - 1) + 1. \tag{1}$$

Now we determine the value $p(k, k)$. Putting $g(a, b) = 2^{a+b}p(a, b) - (a + b)2^{a+b-1}$, equation (1) transforms to

$$g(a, b) = g(a - 1, b) + g(a, b - 1) \tag{2}$$

with $g(a, 0) = a2^{a-1}$. Here we see the same recursion as for the binomial coefficients, but unfortunately the initial values differ. For $a, b > 0$ we have

$$g(a, b) = \sum_{i=1}^{a} g(i, 0)\binom{a - i + b - 1}{b - 1} + \sum_{j=1}^{b} g(0, j)\binom{a - 1 + b - j}{a - 1}.$$

From this we can determine the value of $g(k, k)$.

$$g(k, k) = 2\sum_{i=1}^{k} g(i, 0)\binom{2k - 1 - i}{k - 1} = \sum_{i=1}^{k} i2^i\binom{2k - 1 - i}{k - 1}.$$

This can be transformed into a nice, explicit form using properties of binomial coefficients.

Lemma 2.2. $\sum_{i=1}^{k} i2^i\binom{2k-1-i}{k-1} = \binom{2k}{k}k$.

Proof.

$$2\sum_{i=1}^{k} i2^{i-1}\binom{2k-1-i}{k-1} = 2\sum_{i=1}^{k}\binom{2k-1-i}{k-1}\cdot\sum_{j=0}^{i} i\binom{i-1}{j} =$$

$$2\sum_{i=1}^{k}\binom{2k-1-i}{k-1}\cdot\sum_{j=0}^{i} j\binom{i}{j} = 2\sum_{j=1}^{k} j\sum_{i=j}^{k}\binom{i}{j}\binom{2k-1-i}{k-1}.$$

For the inner part we have

$$\sum_{i=j}^{k}\binom{i}{j}\binom{2k-1-i}{k-1} = \binom{2k}{k+j},$$

because both sides count the number of 0–1 sequences of length $2k$ with $k+j$ 1-coordinates. (Each part of the sum on the left hand side counts the sequences in which the $(j+1)$st 1 is in the $(i+1)$st position.) Using this we obtain

$$2\sum_{j=1}^{k} j\sum_{i=j}^{k}\binom{i}{j}\binom{2k-1-i}{k-1} = 2\sum_{j=1}^{k} j\binom{2k}{k+j} =$$

$$2\left(\sum_{j=1}^{k}(k+j)\binom{2k}{k+j} - \sum_{j=1}^{k} k\binom{2k}{k+j}\right) =$$

$$2\left(2k\sum_{j=1}^{k}\binom{2k-1}{k+j-1} - k\sum_{j=1}^{k}\binom{2k}{k+j}\right) =$$

$$2\left(k2^{2k-1} - (k2^{2k-1} - \frac{1}{2}k\binom{2k}{k})\right) = k\binom{2k}{k}.$$

□

This implies $p(k,k) = k + g(k,k)/2^{2k} = k(1 + \binom{2k}{k}/2^{2k})$.

Put $p(x) = p(wl_k(x))$ and observe the following:

1. If a query involves x and y with $wl_k(x) = wl_k(y) \neq (0,0)$, then because of (1) the sum $\sum_{i=1}^{n} p(x_i)$ decreases by exactly 1. Until at most $(k+1)^2$ teams remain with $wl_k(x) \neq (0,0)$, we can always find such a query by the pigeonhole principle, therefore we obtain our desired situation using at most $p(k+1,k+1)n + c_k$ queries, which gives the upper bound of the theorem.

2. If a query involves teams x and y with $wl_k(x) = (a,b)$, $wl_k(y) = (c,d)$, then the possible "outcomes" are $wl_k(x) = ((a-1)^+, b)$, $wl_k(y) = (c,(d-1)^+)$

and $wl_k(x) = (a, (b-1)^+)$, $wl_k(y) = ((c-1)^+, d)$. Again by (1), it is clear that the decrease of $\sum_{i=1}^{n} p(x_i)$ is 2 if we add up the decrease of both possible cases, so with one of the possible outcomes this sum will decrease by at most 1. If the Adversary's strategy is to answer all queries in such a way that the sum decreases by at most 1, then it is obvious that one needs at least $p(k+1, k+1)n - p(k+1, 0) - p(k+1, k+1) \geq p(k+1, k+1)n - 3k - 3$ queries, which gives the lower bound of the theorem.

3 Selection with one lie

In this section we prove Theorem 1.4. In the first subsection we describe an optimal algorithm (within an additive constant) to find the maximum and minimum elements despite at most one erroneous answer. In the second subsection we modify the potential function used in the proof of Theorem 2.1 to prove the lower bound in Theorem 1.4. We use the notation $wl(x) = wl_1(x) = ((2-w(x))^+, (2-l(x))^+)$.

3.1 Upper bound

In this subsection we describe an algorithm that finds the smallest and the largest elements of a set of size n using not more than $\lceil (11/4 - 1/32)n \rceil + 12$ comparisons if at most one of the comparisons may turn out to be erroneous. Note that the algorithm of the previous section only gives an algorithm that uses $2.75n + O(1)$ questions. For the sake of simplicity we will omit all ceiling signs.

We describe our algorithm in rounds. A round is a set of matches that can be played at the same time. In the first round we consider an arbitrary maximum matching of the teams, therefore with $n/2$ matches played we will have a set X of $n/2$ teams with $wl(x) = (1, 2)$ for all $x \in X$ and a set Y of $n/2$ teams with $wl(y) = (2, 1)$ for all $y \in Y$. In the second round we consider a maximum matching of the teams of X. With this additional $n/4$ matches X will be divided into X_1 and X_2 such that $|X_1|, |X_2| = n/4$ and $wl(x_1) = (0, 2)$ for all $x_1 \in X_1$ and $wl(x_2) = (1, 1)$ for all $x_2 \in X_2$.

In the third round of our algorithm we divide X_2 into two using a matching of $n/8$ additional matches. We obtain $X = X_1 \cup X_2^1 \cup X_2^2$ with $|X_2^1|, |X_2^2| = n/8$ and $wl(x) = (0, 1)$ for all $x \in X_2^1$ and $wl(x) = (1, 0)$ for all $x \in X_2^2$.

Let Y' be the set of teams in Y that were matched in the first round with teams in X_2^2. The fourth round of our algorithm consists of a matching of Y such that any team of Y' plays another team from Y' (i.e. we use a matching of Y that is an expansion of a matching of Y'). After the $n/4$ matches of the fourth round we will have $Y = Y_1 \cup Y_2$ with $|Y_1|, |Y_2| = n/4$, $|Y_2 \cap Y'| = n/16$ and $wl(y) = (2, 0)$ for all $y \in Y_1$ and $wl(y) = (1, 1)$ for all $y \in Y_2$.

In the fifth round of our algorithm we use a matching of Y_2 that is an expansion of a matching of $Y_2 \cap Y'$. After these $n/8$ matches we will have $Y = Y_1 \cup Y_2^1 \cup Y_2^2$ with $|Y_2^1| = |Y_2^2| = n/8$, $|Y_2^2 \cap Y'| = n/32$ and $wl(y) = (1, 2)$ for all $y \in Y_2^1$ and $wl(y) = (2, 1)$ for all $y \in Y_2^2$.

The sixth round is where our algorithm gains the extra $n/32$ matches. In this round the matches that were played in the first round between teams of $Y_2^2 \cap Y'$ and their opponents are replayed. Recall that those matches were won by the teams in X_2^2. If for all matches the same results are obtained as in the first round then for any team x involved in this round we have $wl(x) = (0, 0)$. In this case after the $n/2+n/4+n/8+n/4+n/8+n/32 = 41n/32$ matches of the first six rounds we will have $n/16$ teams with $wl(x) = (0, 0)$, the number of teams with $wl(x) = (0, 2)$ or $(2, 0)$ is $n/4$ each, while the number of teams with $wl(x) = (0, 1)$ or $(1, 0)$ is $7n/32$ each. If n is not a power of 2, then at the matching of each round there could have been an unmatched team, so there can be an additional team for all possible values of $wl(x)$. The total number of "missing" wins and losses to reach the situation of Corollary 1.2 is (at most) $2 \cdot (n/4+n/4)+1 \cdot (7n/32+7n/32)+14 = 46n/32+14$ and with every further match we can decrease this number by one provided all matches are played by two teams either both having two wins or both having two losses and no team x with $wl(x) = (0, 0)$ is playing. Thus the total number of matches played during our algorithm is at most $41n/32+46n/32+12 = 87n/32+12$.

All that remains is to consider what happens if any match in round six ends with a different result than it ended in round one (since only one lie is allowed, there can be at most one such match). In this case we do not know the "real result" of this match, but we know that the results of all the other matches (including the forthcoming ones) are correct, so deleting the two contradicting scores leaves us in finding the smallest and the largest element *without* lies. Therefore for every team x, we can replace $wl(x) = (a, b)$ by $wl_0(x) = ((a-1)^+, (b-1)^+)$. In this way, after the $41n/32$ matches of the first six round, all teams x have $wl_0(x) = (1, 0), (0, 1)$ or $(0, 0)$ (if n is odd, then there is an additional team y that misses all rounds and thus have $wl_0(y) = (1, 1)$), therefore we can finish our algorithm with at most n queries which gives a total of $73n/32$ queries.

It is wirth mentioning that the constant 12 of the upper bound can be decreased to -2, the details are omitted.

3.2 Lower bound

In this subsection we describe a strategy for the Adversary that shows that at least $87n/32-3$ queries are necessary to find both the maximum and the minimum elements. Because of the observation made in the introduction, this strategy should avoid making directed cycles in the championship graph until the very end of the algorithm. We will use a potential function p just as in Section 2, but as the answer for this problem is different from that of the problem in Section 2 we have to modify this function a bit using a "correction function" c. For convenience's sake we first enumerate the values of $p(x) = p(wl(x))$ that we need: $p(0,0) = 0$, $p(1,0) = p(0,1) = 1$, $p(2,0) = p(0,2) = 2$, $p(1,1) = 1.5$, $p(2,1) = p(1,2) = 2.25$, $p(2,2) = 2.75$. Note that if any x and y play each other, then there is a possible outcome such that $p(x) + p(y)$ decreases by at most one.

The function c is defined for each ordered pair of teams, including the case when the two teams are the same.

Let us define $c(x,x) = 1/32$ if $wl(x) = (2,2)$, i.e. if the team x has not played any matches yet, and $c(x,x) = 0$ if $wl(x) \neq (2,2)$.

Let x and y be two distinct teams. If x and y has played their very first game against each other, x has beaten y, and since that x has not won and y has not lost any matches, then x and y are said to be pairs of each other. If this is the case, then let $c(x,y) = (1/2)^{(2-l(x))^{+}+(2-w(y))^{+}}$, otherwise let $c(x,y) = 0$.

With this modification the Adversary will have a strategy avoiding directed cycles such that $\sum p(x) - \sum c(x,y)$ decreases by at most 1 after each comparison. At the beginning of the algorithm $\sum p(x) - \sum c(x,y) = n(p(2,2) - 1/32) = 87n/32$ and at the end of the algorithm this sum is at most $p(2,0) + p(0,2) = 4$, thus Corollary 1.3 gives $87n/32 - 4$ as a lower bound (at the end of the subsection we strengthen this bound by 1 to obtain the statement of the theorem).

Now we define some special subsets of teams that will change during the game. The *Champions' League* and the *Second Division* are both empty at the beginning and if a team becomes an element of one of them, it stays there forever. After each comparison, a team becomes an element of the Champions' League if it is not yet in the Second Division, it was only beaten by teams who are now in the Champions' League and it has two wins. Similarly, after each comparison, a team becomes an element of the Second Division if it is not yet in the Champions' League, it only won against teams who are now in the Second Division and it has two losses. Note that not only the winner (loser) of a comparison may move into the Champions' League (Second Division), e.g. if $wl(x) = (1,0)$ and the only team beaten by x moves into the Second Division, then x moves there as well. If a team is not an element of the Champions' League or the Second Division, we say that it is *active*. We say that an active team is *in reach* of the Champions' League (or of the Second Division) if it only needs one more win (loss) to become a member. We would like to find an Adversary's strategy such that during the whole process every team that has already played a game, is either a member of the Champions' League or of the Second Division or is in reach of (at least) one of them (condition 1). Furthermore, every previous opponent of each active team will be inactive except maybe its pair (if it has any) (condition 2).

Now we describe the strategy of the Adversary, that is, we exhibit a function that decides who is winning which game such that $S = \sum p(x) - \sum c(x, y)$ decreases by at most 1 after each comparison and the above mentioned conditions hold.

If a team gets into the Champions' League, then from that on it will win every match against teams that were not in the Champions' League at the moment of its qualification (the moment when it became a member of the Champions' League). Similarly, if a team gets into the Second Division, then it will lose every further match against teams that were not in the Second Division at the moment when it got there. Obviosly, this kind of matches cannot give directed cycles.

If two active, pairless teams play, then there always exists an answer that decreases S by at most 1. This answer cannot give a directed cycle since all their previous opponents were already inactive. Also note that, unless this was the first game for both teams, one of the teams becomes inactive.

The only case that remains is when an active team x who has an active pair y is playing another active team z. Without loss of generality, suppose that x has beaten y in their first game. By condition 1, this implies that x is in reach of the Champions' League and y is in reach of the Second Division. The possible values of $wl(x)$ are $(1, 2), (1, 1)$ and $(1, 0)$, while the possible values of $wl(z)$ are $(2, 1), (1, 1)$ and $(0, 1)$.

CASE 0: $z = y$. To avoid a cycle of length two, x has to win the game. S decreases by $p(x) + p(y) - c(x, y)$ (since $c(x, y)$ vanishes after the game) and it is easy to check that this is at most 1.

CASE 1: z has no pair. This means that z cannot have two wins or losses (otherwise it would not be active).

CASE 1.1: $l(z) \geq w(z)$.

Case 1.1.1: $wl(x) \neq (1,0)$. If x wins, then $p(z)$ decreases by at most 0.5, $p(x)$ also decreases by at most 0.5, $c(x,y)$ vanishes.

Case 1.1.2: $wl(x) = (1,0)$.

Case 1.1.2.1: $wl(z) = (1,1)$. If z wins, it moves into the Champions' League, x and y remain unaffected.

Case 1.1.2.2: $wl(z) = (2,1)$. If x wins, $p(z)$ decreases by 0.25, $p(x)$ decreases by 1, but $c(x,y) \geq 1/4$ vanishes, since x moves into the Champions' League.

Case 1.1.2.3: $wl(z) = (2,2)$. Now we need z to win but this would not make it move into the Champions' League ruining condition 2. We solve this problem by giving some more information: we answer the same question again without being asked, this way $wl(z)$ becomes $(0,2)$ and z moves into the Champions' League. Of course we are not allowed to count the question twice, but we do not need to if we can show that S decreases by at most 1 after the two answers. Indeed, $p(z)$ only decreases by 0.75, while x and y are unaffected.

Case 1.2: $l(z) < w(z)$. This means that $wl(z) = (1,2)$.

Case 1.2.1: $wl(x) = (1,0)$. If z wins, it moves into the Champions' League, x and y remain unaffected.

Case 1.2.2: $wl(x) = (1,2)$. If x wins, it moves into the Champions' League, $p(z)$ decreases by 0.75, $p(x)$ decreases by 0.25, $c(x,y)$ vanishes.

Case 1.2.3: $wl(x) = (1,1)$.

Case 1.2.3.1: $c(x,y) \geq 1/4$. If x wins, it moves into the Champions' League, $p(z)$ decreases by 0.75, $p(x)$ decreases by 0.5, $c(x,y)$ vanishes.

Case 1.2.3.2: $c(x,y) = 1/8$. If z wins, it moves into the Champions' League, $p(z)$ decreases by 0.25, $p(x)$ decreases by 0.5, $c(x,y)$ increases by 1/8.

Case 2: z has a pair q who was beaten by z. Now either x or z moves into the Champions' League and either $c(x,y)$ or $c(z,q)$ vanishes. Note that in this case the roles of x and z are symmetric, this eliminates some cases.

Case 2.1: $wl(x) = wl(z)$. If $c(x,y) \leq c(z,q)$, then z wins, otherwise x wins, so $p(x) + p(z)$ decreases by 1, $c(x,y) + c(z,q)$ does not increase.

Case 2.2: $wl(x) = (1,0)$. If z wins, $p(z) - c(z,q)$ decreases by at most 1, x and y are unaffected.

CASE 2.2': $wl(z) = (1, 0)$ is analogous to 2.2.

CASE 2.3: $wl(x) = (1, 1)$, $wl(z) = (1, 2)$.

CASE 2.3.1: $c(x, y) \leq c(z, q) + 1/4$. If z wins, $p(x) + p(z)$ decreases by 0.75, $c(x, y)$ increases by at most $c(z, q) + 1/4$, while $c(z, q)$ vanishes.

CASE 2.3.2: $c(x, y) = 1/2$, $c(z, q) < 1/4$. If x wins, $p(x) + p(z)$ decreases by 1.25, $c(z, q)$ increases by less than $1/4$, while $c(z, q)$ vanishes.

CASE 2.3': $wl(x) = (1, 2)$, $wl(z) = (1, 1)$ is analogous to 2.3.

CASE 3: z has a pair q who has beaten z.

CASE 3.1: $wl(z) = (2, 1)$. If x wins, $p(z)$ decreases by 0.25 and $p(x) - c(x, y)$ decreases by at most 0.75.

CASE 3.2: $wl(z) = (1, 1)$.

CASE 3.2.1: $wl(x) \neq (1, 0)$. If x wins, $p(z)$ decreases by 0.5, $p(x)$ also decreases by at most 0.5, $c(x, y)$ and $c(q, z)$ vanish.

CASE 3.2.2: $wl(x) = (1, 0)$.

CASE 3.2.2.1: If $c(x, y) + c(q, z) \geq 1/2$, then let x win, so $p(z)$ decreases by 0.5, $p(x)$ decreases by 1, but $c(x, y)$ and $c(q, z)$ vanish.

CASE 3.2.2.2: If $c(x, y) + c(q, z) < 1/2$, then $c(x, y) = 1/4$ and $c(q, z) = 1/8$, thus we have $wl(q) = (1, 2)$ and $wl(y) = (2, 1)$. If z wins, then $p(z)$ decreases by 0.5 and $c(q, z)$ increases by $1/8$. Condition 2 is ruined, so we give some more information just like in case 1.1.2.3. We give the additional information that q has beaten y, making all the involved teams inactive. Now $p(q)$ and $p(y)$ decrease by 0.25 each, while $c(x, y)$ and $c(q, z)$ vanish.

CASE 3.3: $wl(z) = (0, 1)$.

CASE 3.3.1: If $wl(x) \neq (1, 0)$, then this is the same situation as 3.1 or 3.2.2, just swap the roles of x and z and the wins and losses.

CASE 3.3.2: $wl(x) = (1, 0)$. If z wins, then S remains unchanged, but condition 2 is ruined. We again use the trick of giving unwanted information, we say that q has beaten z (for a second time). This way they both go into the Champions' League and S decreases by at most 1.

We checked all the cases, which proves the bound $M(n) \geq 87n/32 - 4$.

Now we show how to strengthen this bound by 1 to match the lower bound of Theorem 1.4.

According to the Adversary's strategy we have described, in the very last match either a team with only one win wins (so it cannot be the minimum) or a team with only one loss loses (so it cannot be the maximum). Now we change the answer of the Adversary to this last question. We claim that in this way either the minimum or the maximum element remains unknown, hence another question is needed, which proves the lower bound of Theorem 1.4. We may suppose that a team x with only one win is beaten by a team y. Now we have two different possibilities to make the championship graph acyclic by changing the orientation of at most one edge: either we change the edge corresponding to this last match or we change the edge corresponding to the match won by x earlier. It is easy to see that the minimum elements are different for the two cases, thus we need (at least) one more question to find the minimum element.

4 Further results and remarks

In this final section we gather further results and open problems related to Theorem 1.4. The most important unsolved question is of course to find the minimum number of comparisons needed in the cases $k > 1$. How much better can one do than the simple upper bound of Theorem 2.1?

In what follows, we enumerate some models where either restrictions are posed for the possible comparisons or for the relation of the possible erroneous answers. One way that a restriction can be posed is if one can ask a pair $\{x_i, x_j\}$ to be compared at most once. We call this restricted model the *Gentlemen's model* (because gentlemen do not question each other's answers...). With this restriction one cannot find the maximum provided one lie is allowed even if every possible pair is compared. To see this, just observe that if the one and only erroneous answer is when the maximum is compared to the third largest element, then clearly one cannot tell the difference between the three largest elements.

However, one can find algorithms that provide solutions for the following problems:

(i) Find 3 elements such that one of them is the largest.
(ii) Find an element which is one of the three largest.

The next theorem gives the exact solution for the first problem.

Theorem 4.1. *In the Gentlemen's model the minimum number of comparisons needed to find 3 elements such that one of them is the largest is* $2n-5$*,* *if* $n > 3$.

Proof: First we describe the optimal algorithm.

STEP 1: The teams x_1 and x_2 play a match. Let x_1 be the loser.
STEP 2: Two teams without a match play. Denote the loser by x_3.
STEP 3: x_1 and x_3 play.
STEP 4: Delete the loser of the previous match and the two matches it has lost. Denote the winner of the previous match by x_1, since it is the loser of the only remaining edge. Go to STEP 2.

We continue this procedure until there are only 3 undeleted elements.

In every execution of STEP 2 an element is deleted that cannot be the largest because it has two losses. Therefore one of the remaining 3 elements is the largest. There are $n-3$ elements deleted, hence STEP 4, STEP 2 and STEP 3 are executed $n-3$ times and STEP 1 only once. Comparisons occur once in STEP 1, STEP 2 and STEP 3, so there are at most $2n-5$ of them.

For the lower bound we describe an Adversary's strategy. The order of the elements will be determined after the first question and the Adversary will never lie. The winner of the first match will be the largest element, the loser the second largest and fix an arbitrary order for the rest.

Clearly there will be no directed cycles in the championship graph. Hence an element x can be the largest one if and only if it has lost at most one match. When someone names three elements such that the largest element is among them, all the other $n-3$ elements must have lost at least two matches. We also know that the second largest lost exactly one match, so there have been at least $2n-5$ matches. This finishes the proof.

We mention an upper bound for the second problem without proof:

Theorem 4.2. *In the Gentlemen's model the minimum number of comparisons needed to find an element which is one of the three largest is at most* $2n - \log n + O(1)$.

Problems that we dealt with in Section 2 and 3 were about to find the maximum and the minimum element. In the Gentlemen's model, we cannot ask for an algorithm that would provide us these elements, but we could ask for an algorithm that gives 6 elements that contain the maximum and the minimum (in fact, 4 elements would suffice). Note that the algorithm presented in Theorem 2.1 can be arranged in such a way that no comparisons are asked twice, therefore $11/4n$ is a trivial upper bound and the lower bound $(11/4-1/32)n$ of Theorem 1.3 obviously remains valid in the more restrictive

Gentlemen's model. Again we state a better upper bound without proof that we conjecture to be (asymptotically) optimal.

Theorem 4.3. *In the Gentlemen's model the minimum number of comparisons needed to find six elements which contains the maximum and the minimum is at most* $(11/4 - 1/96)n + O(1)$.

In our last model an unlimited number of erroneous answers may occur, but every element may be involved in at most one erroneous comparison. We call this model the *1-factor model* (as the edges in the championship graph corresponding to the lies form a (partial) matching). As Claim 1.1 remains valid in this model, the trivial upper bound $11/4n$ of Theorem 2.1 holds and the lower bound $87n/32$ of Theorem 1.4 is also true. For the first thought, one might conjecture that in this model the trivial upper bound could be closer to the truth as there can be much more erroneous answers. Contrary to this, the following theorem holds.

Theorem 4.4. *In the 1-factor model the minimum number of comparisons needed to find the maximum and the minimum is* $87n/32 + \Theta(1)$.

Proof: The lower bound follows from Theorem 1.4. For the upper bound we have to describe an algorithm. We use again the potential function p introduced in Section 2. We will say that at a match we *gain* c *(or lose* c*)* if the sum $\sum p(x)$ decreases by $1 + c$ (or $1 - c$) at that match. Note that if teams x and y play such that $wl(x) = wl(y)$ then we do not lose or gain anything.

At the beginning of our algorithm, we pick 8 teams $x_1, x_2, x_3, x_4, y_1, y_2, y_3, y_4$ and x_i plays y_i for all $1 \leq i \leq 4$. We may suppose that the x_i's win and now x_1 plays x_2 and x_3 plays x_4. Finally the losers of these two matches play. We may assume that x_1 is the team that won its first match and lost the other two. Note that until now we did not lose or gain anything as at every match the wl-value of the playing teams were the same.

Now $wl(x_1) = (1, 0)$ and $wl(y_1) = (2, 1)$ and we replay their match. If x_1 wins again, then we gain $1/4$ and repeat this procedure with the next 8 teams. If this time y_1 beats x_1, then $wl(x_1)$ stays $(1, 0)$, while $wl(y_1)$ becomes $(1, 1)$, thus we lose $1/4$, but we know that any further match involving x_1 or y_1 will give the true result. To exploit this fact we pick 5 more teams u, v_1, v_2, w_1, w_2 with wl-value $(2, 2)$. Let y_1 play with u and v_i play with w_i for $i = 1, 2$. At the match between y_1 and u we gain $1/4$ as $wl(u)$ will be $(0, 2)$ or $(2, 0)$, since the result of this match cannot be a lie. At the matches between the v_i's and the w_i's we do not gain or lose anything, but then x_1 should play one of the losers (the team with wl-value $(2, 1)$) and y_1 should play the other loser if y_1 lost to u (i.e. $wl(y_1) = (1, 0)$) and with a winner if y_1 beats u. It is easy to verify that because these matches cannot have erroneous results, we will gain 1/4 at each of these matches, thus in total we gain $3 \cdot 1/4 - 1/4 = 1/2$ at matches involving these 13 teams.

So we obtained that depending on the answer we got for the replay between x_1 and y_1, we can gain 1/4 at matches involving 8 teams or 1/2 at matches involving 13 teams. Therefore we can gain at least $n/8 \cdot 1/4 = n/32$ which gives the upper bound of the theorem, since we can finish the algorithm in such a way that until the last few matches every match is played between teams with the same wl-value.

References

[1] M. Aigner, Finding the maximum and the minimum, *Discrete Applied Mathematics* 74 (1997), 1-12.

[2] C. Deppe, Coding with Feedback and Searching with Lies, in *Entropy, Search, Complexity*, Bolyai Society Mathematical Studies, 16 (2007), 27-70.

[3] A. Pelc, Searching games with errors – Fifty years of coping with liars, *Theoretical Computer Science* 270 (2002), 71-109.

[4] I. Pohl, A sorting problem and its complexity, *Communications of the ACM* 15 (1972), 462-464.

[5] B. Ravikumar, K. Ganesan, K.B. Lakshmanan, On Selecting the Largest Element in Spite of Erroneous Information, STACS 1987, 88-99.