Finding the maximum and minimum elements with lies

Dömötör Pálvölgyi*

Eötvös University (ELTE), Budapest

Katona '70

*Mostly joint work with Dániel Gerbner, Balázs Patkós and Gábor Wiener

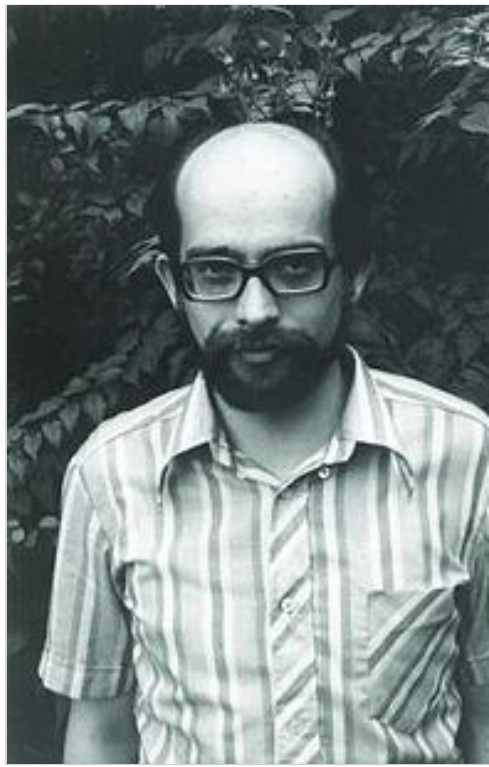
Gyula O. H. Katona – Wikipedia tiếng Việt http://vi.wikipedia.org/wiki/Gyula_

Gyula O. H. Katona

Bách khoa toàn thư mở Wikipedia

Gyula O. H. Katona sinh ngày 16.3.1941 là nhà toán học người Hungary, nổi tiếng về công trình nghiên cứu lý thuyết tổ hợp và nhất là về Định lý Kruskal–Katona cùng việc chứng minh định lý Erdős–Ko–Rado của ông.

Katona là tổng thư ký của Hội Toán học János Bolyai từ năm 1990 tới 1996. Năm 1966 và 1968 ông đoạt Giải Grünwald của Hội Toán học János Bolyai dành cho các nhà toán học trẻ xuất sắc. Năm 1976, ông đoạt Giải Alfréd Rényi của Viện Toán học Alfréd Rényi. Năm 2006, ông được bầu vào Hàn lâm viện Khoa học Hungary.



Gyula Katona tại Erlangen, 1975.

Gia đình

Gyula O.H. Katona là cha của Gyula Y. Katona, cũng là nhà toán học chuyên về lý thuyết tổ hợp như người cha.

70 (số)

Bách khoa toàn thư mở Wikipedia

Bảy mươi (70) là một số tự nhiên ngay sau 69 và ngay trước 71.

Lấy từ "http://vi.wikipedia.org/wiki/70_(s%E1%BB%91)"
Thể loại: Số nguyên

<< **70** 71 72 73 74 75 76 77 78 79 >>

0 10 20 30 40 50 60 **70** 80 90 >>

70	
Phân tích nhân tử	2 · 5
Số La Mã	LXX
Mã Unicode của số La Mã	
Biểu diễn theo Hệ nhị phân	1000
Biểu diễn theo Hệ thập lục phân	46

Finding the maximum and minimum elements with lies

Dömötör Pálvölgyi*

Eötvös University (ELTE), Budapest

Katona '70

*Mostly joint work with Dániel Gerbner, Balázs Patkós and Gábor Wiener

*Mostly joint work with Dániel Gerbner, Balázs Patkós and Gábor Wiener

Finding the maximum and minimum elements with lies

Dömötör Pálvölgyi*

Eötvös University (ELTE), Budapest

Katona 170?

*Mostly joint work with Dániel Gerbner, Balázs Patkós and Gábor Wiener
*Mostly joint work with Dániel Gerbner, Balázs Patkós and Gábor Wiener
*Mostly joint work with Dániel Gerbner, Balázs Patkós and Gábor Wiener

Finding the maximum and minimum elements with lies

Dömötör Pálvölgyi*

Eötvös University (ELTE), Budapest

Katona 1970?

*Mostly joint work with Dániel Gerbner, Balázs Patkós and Gábor Wiener
*Mostly joint work with Dániel Gerbner, Balázs Patkós and Gábor Wiener
*Mostly joint work with Dániel Gerbner, Balázs Patkós and Gábor Wiener
*Mostly joint work with Dániel Gerbner, Balázs Patkós and Gábor Wiener

Finding the maximum and minimum elements with lies

Dömötör Pálvölgyi*

Eötvös University (ELTE), Budapest

Katona 1970?

Gyula, watch out, your students are rising!

*Mostly joint work with Dániel Gerbner, Balázs Patkós and Gábor Wiener
*Mostly joint work with Dániel Gerbner, Balázs Patkós and Gábor Wiener
*Mostly joint work with Dániel Gerbner, Balázs Patkós and Gábor Wiener
*Mostly joint work with Dániel Gerbner, Balázs Patkós and Gábor Wiener
*Mostly joint work with Dániel Gerbner, Balázs Patkós and Gábor Wiener

Beamer is messing with you.

Beamer is messing with you.

You have no joint paper?

Beamer is messing with you.

You have no joint paper?

You have not taken any of his lectures?

Beamer is messing with you.

You have no joint paper?

You have not taken any of his lectures?

Maybe you should start the talk...

What is a search game? (No entry in Vietnamese wiki.)

What is a search game? (No entry in Vietnamese wiki.)

20 questions game - Find an element of a given set using at most 20 Yes-No questions.

What is a search game? (No entry in Vietnamese wiki.)

20 questions game - Find an element of a given set using at most 20 Yes-No questions.

Faulty coin problem - 27 coins, one is lighter, find it using 3 measurements on a scale.

What is a search game? (No entry in Vietnamese wiki.)

20 questions game - Find an element of a given set using at most 20 Yes-No questions.

Faulty coin problem - 27 coins, one is lighter, find it using 3 measurements on a scale.

Another puzzle - 1000 coins, at most 100 are faulty. How many Yes-No questions do we need to find a genuine coin?

What is a search game? (No entry in Vietnamese wiki.)

20 questions game - Find an element of a given set using at most 20 Yes-No questions.

Faulty coin problem - 27 coins, one is lighter, find it using 3 measurements on a scale.

Another puzzle - 1000 coins, at most 100 are faulty. How many Yes-No questions do we need to find a genuine coin?

Lies - Some of the answers might be erroneous.

What is a search game? (No entry in Vietnamese wiki.)

20 questions game - Find an element of a given set using at most 20 Yes-No questions.

Faulty coin problem - 27 coins, one is lighter, find it using 3 measurements on a scale.

Another puzzle - 1000 coins, at most 100 are faulty. How many Yes-No questions do we need to find a genuine coin?

Lies - Some of the answers might be erroneous.

In our model - Fixed number of lies, best worst-case algorithm, questions can depend on previous answers.

Our model

We are given an ordered set of size n (with no equal elements) and we can use pairwise comparisons.

Our model

We are given an ordered set of size n (with no equal elements) and we can use pairwise comparisons.

Sorting:

Our model

We are given an ordered set of size n (with no equal elements) and we can use pairwise comparisons.

Sorting: $n \log n + \Theta(n)$.

Our model

We are given an ordered set of size n (with no equal elements) and we can use pairwise comparisons.

Sorting: $n \log n + \Theta(n)$.

Finding the biggest:

Our model

We are given an ordered set of size n (with no equal elements) and we can use pairwise comparisons.

Sorting: $n \log n + \Theta(n)$.

Finding the biggest: $n - 1$.

Our model

We are given an ordered set of size n (with no equal elements) and we can use pairwise comparisons.

Sorting: $n \log n + \Theta(n)$.

Finding the biggest: $n - 1$.

Finding the biggest and the smallest:

Our model

We are given an ordered set of size n (with no equal elements) and we can use pairwise comparisons.

Sorting: $n \log n + \Theta(n)$.

Finding the biggest: $n - 1$.

Finding the biggest and the smallest: $\lceil \frac{3n}{2} \rceil - 2$.

Our model

We are given an ordered set of size n (with no equal elements) and we can use pairwise comparisons.

Sorting: $n \log n + \Theta(n)$.

Finding the biggest: $n - 1$.

Finding the biggest and the smallest: $\lceil \frac{3n}{2} \rceil - 2$.

Finding the biggest with k lies:

Our model

We are given an ordered set of size n (with no equal elements) and we can use pairwise comparisons.

Sorting: $n \log n + \Theta(n)$.

Finding the biggest: $n - 1$.

Finding the biggest and the smallest: $\lceil \frac{3n}{2} \rceil - 2$.

Finding the biggest with k lies: $(k+1)n - 1$.

Our model

We are given an ordered set of size n (with no equal elements) and we can use pairwise comparisons.

Sorting: $n \log n + \Theta(n)$.

Finding the biggest: $n - 1$.

Finding the biggest and the smallest: $\lceil \frac{3n}{2} \rceil - 2$.

Finding the biggest with k lies: $(k+1)n - 1$.

Finding the biggest and the smallest with k lies - Our main topic.

Finding the biggest and the smallest with k lies

Theorem (Aigner)

$\boxed{\textit{General } k} \leq (k + \Theta(\sqrt{k}))n + \Theta_k(1).$

Finding the biggest and the smallest with k lies

Theorem (Aigner)

$\boxed{\textit{General } k} \leq (k + \Theta(\sqrt{k}))n + \Theta_k(1).$

$\boxed{k = 1} \leq \frac{87}{32}n + \Theta(1).$

Finding the biggest and the smallest with k lies

Theorem (Aigner)

$\boxed{\textit{General } k} \leq (k + \Theta(\sqrt{k}))n + \Theta_k(1).$

$\boxed{k = 1} \leq \frac{87}{32}n + \Theta(1).$

Theorem (Gerbner, P., Patkós, Wiener)

$\boxed{k = 1} = \frac{87}{32}n + \Theta(1).$

Finding the biggest and the smallest with k lies

Theorem (Aigner)

$\boxed{\textit{General } k} \leq (k + \Theta(\sqrt{k}))n + \Theta_k(1).$

$\boxed{k = 1} \leq \frac{87}{32}n + \Theta(1).$

Theorem (Gerbner, P., Patkós, Wiener)

$\boxed{k = 1} = \frac{87}{32}n + \Theta(1).$

Theorem (Hoffmann, Matoušek, Okamoto, Zumstein)

$\boxed{\textit{General } k} \leq (k + 1 + 10)n + \Theta_k(1).$

Finding the biggest and the smallest with k lies

Theorem (Aigner)

$\boxed{\textit{General } k} \leq (k + \Theta(\sqrt{k}))n + \Theta_k(1).$

$\boxed{k = 1} \leq \frac{87}{32}n + \Theta(1).$

Theorem (Gerbner, P., Patkós, Wiener)

$\boxed{k = 1} = \frac{87}{32}n + \Theta(1).$

Theorem (Hoffmann, Matoušek, Okamoto, Zumstein)

$\boxed{\textit{General } k} \leq (k + 1 + 10)n + \Theta_k(1).$

Does "10" $\to 0$ as $k \to \infty$?

Finding the biggest and the smallest with k lies

Theorem (Aigner)

$\boxed{\textit{General } k} \leq (k + \Theta(\sqrt{k}))n + \Theta_k(1).$

$\boxed{k = 1} \leq \frac{87}{32}n + \Theta(1).$

Theorem (Gerbner, P., Patkós, Wiener)

$\boxed{k = 1} = \frac{87}{32}n + \Theta(1).$

Theorem (Hoffmann, Matoušek, Okamoto, Zumstein)

$\boxed{\textit{General } k} \leq (k + 1 + 10)n + \Theta_k(1).$

Does "10" $\to 0$ as $k \to \infty$?

Theorem (P.)

No.

Finding the biggest and the smallest with k lies

Theorem (Aigner)

$\boxed{\textit{General } k} \leq (k + \Theta(\sqrt{k}))n + \Theta_k(1).$

$\boxed{k = 1} \leq \frac{87}{32}n + \Theta(1).$

Theorem (Gerbner, P., Patkós, Wiener)

$\boxed{k = 1} = \frac{87}{32}n + \Theta(1).$

Theorem (Hoffmann, Matoušek, Okamoto, Zumstein)

$\boxed{\textit{General } k} \leq (k + 1 + 10)n + \Theta_k(1).$

Does "10" $\to 0$ as $k \to \infty$?

Theorem (P.)

No. $\boxed{\textit{General } k} \geq (k + 1.5)n + \Theta_k(1).$

Terminology

Terminology

teams - elements of the set

Terminology

teams - elements of the set

match - comparison

Terminology

teams - elements of the set

match - comparison

win - bigger in comparison

Terminology

teams - elements of the set

match - comparison

win - bigger in comparison

lose - smaller in comparison

Terminology

teams - elements of the set

match - comparison

win - bigger in comparison

lose - smaller in comparison

comparison graph - digraph, ab is an edge iff a has beaten b

Finding the biggest with k lies

Finding the biggest with k lies

Claim

This requires $(k+1)(n-1)+k$.

Finding the biggest with k lies

Claim

This requires $(k+1)(n-1)+k$.

Upper bound:

Finding the biggest with k lies

Claim

This requires $(k+1)(n-1)+k$.

Upper bound: Ask everything you would ask without lies until you get the same answer $k+1$ times.

Finding the biggest with k lies

Claim

This requires $(k+1)(n-1)+k$.

Upper bound: Ask everything you would ask without lies until you get the same answer $k+1$ times.

Easy Lower bound:

Finding the biggest with k lies

Claim

This requires $(k+1)(n-1)+k$.

Upper bound: Ask everything you would ask without lies until you get the same answer $k+1$ times.

Easy Lower bound: $< (k+1)(n-1)$ questions are not enough:

Finding the biggest with k lies

Claim

This requires $(k+1)(n-1)+k$.

Upper bound: Ask everything you would ask without lies until you get the same answer $k+1$ times.

Easy Lower bound: $< (k+1)(n-1)$ questions are not enough:

Proof.

After $< (k+1)(n-1)$ questions, there are at least two teams with in-degree $\leq k$, trivially any of these can be the best.

Finding the biggest with k lies

Claim

This requires $(k+1)(n-1)+k$.

Upper bound: Ask everything you would ask without lies until you get the same answer $k+1$ times.

Easy Lower bound: $< (k+1)(n-1)$ questions are not enough:

Proof.

After $< (k+1)(n-1)$ questions, there are at least two teams with in-degree $\leq k$, trivially any of these can be the best. Trivially???

Finding the biggest with k lies

Claim

This requires $(k+1)(n-1)+k$.

Upper bound: Ask everything you would ask without lies until you get the same answer $k+1$ times.

Easy Lower bound: $< (k+1)(n-1)$ questions are not enough:

Proof.

After $< (k+1)(n-1)$ questions, there are at least two teams with in-degree $\leq k$, trivially any of these can be the best. Trivially???
If there is a directed cycle in the comparison graph, then no!

Finding the biggest with k lies

Claim

This requires $(k+1)(n-1)+k$.

Upper bound: Ask everything you would ask without lies until you get the same answer $k+1$ times.

Easy Lower bound: $< (k+1)(n-1)$ questions are not enough:

Proof.

After $< (k+1)(n-1)$ questions, there are at least two teams with in-degree $\leq k$, trivially any of these can be the best. Trivially???
If there is a directed cycle in the comparison graph, then no!
But if the Adversary never lies, then yes. □

Finding the biggest with k lies

Claim

This requires $(k+1)(n-1)+k$.

Upper bound: Ask everything you would ask without lies until you get the same answer $k+1$ times.

Easy Lower bound: $< (k+1)(n-1)$ questions are not enough:

Proof.

After $< (k+1)(n-1)$ questions, there are at least two teams with in-degree $\leq k$, trivially any of these can be the best. Trivially???
If there is a directed cycle in the comparison graph, then no!
But if the Adversary never lies, then yes. □

Rule of thumb - The evil Adversary never lies.

Finding the biggest with k lies

Claim

This requires $(k+1)(n-1)+k$.

Upper bound: Ask everything you would ask without lies until you get the same answer $k+1$ times.

Easy Lower bound: $< (k+1)(n-1)$ questions are not enough:

Proof.

After $< (k+1)(n-1)$ questions, there are at least two teams with in-degree $\leq k$, trivially any of these can be the best. Trivially???
If there is a directed cycle in the comparison graph, then no!
But if the Adversary never lies, then yes. □

Rule of thumb - The evil Adversary never lies.
Except maybe at the end.

Finding the biggest with k lies

Claim

This requires $(k+1)(n-1)+k$.

Rule of thumb - No lies until the end.

Lower bound: $< (k+1)(n-1)+k$ questions are not enough:

Finding the biggest with k lies

Claim

This requires $(k+1)(n-1)+k$.

Rule of thumb - No lies until the end.

Lower bound: $< (k+1)(n-1)+k$ questions are not enough:

Proof.

Reply without lies until there are only two teams with in-degree $\leq k$ and one of them has in-degree 0, the other k.

Finding the biggest with k lies

Claim

This requires $(k+1)(n-1)+k$.

Rule of thumb - No lies until the end.

Lower bound: $< (k+1)(n-1)+k$ questions are not enough:

Proof.

Reply without lies until there are only two teams with in-degree $\leq k$ and one of them has in-degree 0, the other k.

After this we "change our mind" and reply such that the team with in-degree k always wins. □

General way to increase lower bound by k for k lies

General way to increase lower bound by k for k lies

Claim

Suppose goal is to determine f using some allowed questions.

General way to increase lower bound by k for k lies

Claim

Suppose goal is to determine f using some allowed questions. Suppose answers received are consistent. (Possibly no lies.)

General way to increase lower bound by k for k lies

Claim

Suppose goal is to determine f using some allowed questions.
Suppose answers received are consistent. (Possibly no lies.)
If f is yet unknown, at least $k+1$ further questions are needed.

General way to increase lower bound by k for k lies

Claim

Suppose goal is to determine f using some allowed questions.
Suppose answers received are consistent. (Possibly no lies.)
If f is yet unknown, at least $k+1$ further questions are needed.

Proof.

Consistency $\Rightarrow$ All answers are true for some x.

General way to increase lower bound by k for k lies

Claim

Suppose goal is to determine f using some allowed questions.
Suppose answers received are consistent. (Possibly no lies.)
If f is yet unknown, at least $k+1$ further questions are needed.

Proof.

Consistency $\Rightarrow$ All answers are true for some x.
Unknown $\Rightarrow$ There is y for which $\leq k$ false and $f(x) \neq f(y)$.

General way to increase lower bound by k for k lies

Claim

Suppose goal is to determine f using some allowed questions.
Suppose answers received are consistent. (Possibly no lies.)
If f is yet unknown, at least $k+1$ further questions are needed.

Proof.

Consistency $\Rightarrow$ All answers are true for some x.
Unknown $\Rightarrow$ There is y for which $\leq k$ false and $f(x) \neq f(y)$.
Answer next k questions according to y. $\square$

Finding the biggest and smallest without lies

Finding the biggest and smallest without lies

Algorithm is to take complete matching, then select best from winners and worst from losers.

Finding the biggest and smallest without lies

Algorithm is to take complete matching, then select best from winners and worst from losers. This is $\lceil \frac{3n}{2} \rceil - 2$ comparisons.

Finding the biggest and smallest without lies

Algorithm is to take complete matching, then select best from winners and worst from losers. This is $\lceil \frac{3n}{2} \rceil - 2$ comparisons.

Proof of matching lower bound is folklore:

Finding the biggest and smallest without lies

Algorithm is to take complete matching, then select best from winners and worst from losers. This is $\lceil \frac{3n}{2} \rceil - 2$ comparisons.

Proof of matching lower bound is folklore:

Proof.
Give every team a red and a blue pebble.

Finding the biggest and smallest without lies

Algorithm is to take complete matching, then select best from winners and worst from losers. This is $\lceil \frac{3n}{2} \rceil - 2$ comparisons.

Proof of matching lower bound is folklore:

Proof.
Give every team a red and a blue pebble.

Take the red if it loses, take the blue if it wins.

Finding the biggest and smallest without lies

Algorithm is to take complete matching, then select best from winners and worst from losers. This is $\lceil \frac{3n}{2} \rceil - 2$ comparisons.

Proof of matching lower bound is folklore:

Proof.
Give every team a red and a blue pebble.

Take the red if it loses, take the blue if it wins.

Game ends when there is only one pebble of each color.

Finding the biggest and smallest without lies

Algorithm is to take complete matching, then select best from winners and worst from losers. This is $\lceil \frac{3n}{2} \rceil - 2$ comparisons.

Proof of matching lower bound is folklore:

Proof.
Give every team a red and a blue pebble.

Take the red if it loses, take the blue if it wins.

Game ends when there is only one pebble of each color.

Adversary can answer such that only matches between two newbies result in loss of two pebbles

Finding the biggest and smallest without lies

Algorithm is to take complete matching, then select best from winners and worst from losers. This is $\lceil \frac{3n}{2} \rceil - 2$ comparisons.

Proof of matching lower bound is folklore:

Proof.
Give every team a red and a blue pebble.

Take the red if it loses, take the blue if it wins.

Game ends when there is only one pebble of each color.

Adversary can answer such that only matches between two newbies result in loss of two pebbles $\Rightarrow \lceil \frac{3n}{2} \rceil - 2$ questions are needed. □

Natural algorithm for 1 lie

Natural algorithm for 1 lie

Give every team 2 red and 2 blue pebbles.

Natural algorithm for 1 lie

Give every team 2 red and 2 blue pebbles.

Match teams with the same set of pebbles until possible.

Natural algorithm for 1 lie

Give every team 2 red and 2 blue pebbles.

Match teams with the same set of pebbles until possible.

After this at most 8 teams remain with at least one pebble, enough to compare them.

Natural algorithm for 1 lie

Give every team 2 red and 2 blue pebbles.

Match teams with the same set of pebbles until possible.

After this at most 8 teams remain with at least one pebble, enough to compare them.

How many comparisons are needed in total?

Potential function p

Potential function p

$p(a, b)$ measures number of questions needed in average for a team with a red and b blue pebbles to lose all its pebbles.

Potential function p

$p(a, b)$ measures number of questions needed in average for a team with a red and b blue pebbles to lose all its pebbles.

$p(0, 0) = 0$,

Potential function p

$p(a, b)$ measures number of questions needed in average for a team with a red and b blue pebbles to lose all its pebbles.

$$p(0, 0) = 0, \qquad p(a, 0) = p(0, a) = a,$$

Potential function p

$p(a, b)$ measures number of questions needed in average for a team with a red and b blue pebbles to lose all its pebbles.

$$p(0, 0) = 0, \qquad p(a, 0) = p(0, a) = a,$$

$$2p(a, b) = p(a - 1, b) + p(a, b - 1) + 1 \text{ for } a, b > 0.$$

Potential function p

$p(a, b)$ measures number of questions needed in average for a team with a red and b blue pebbles to lose all its pebbles.

$$p(0,0) = 0, \qquad p(a,0) = p(0,a) = a,$$

$$2p(a,b) = p(a-1,b) + p(a,b-1) + 1 \text{ for } a, b > 0.$$

$$p(1,1) = 1.5, p(2,1) = p(1,2) = 2.25, p(2,2) = 2.75.$$

Potential function p

$p(a, b)$ measures number of questions needed in average for a team with a red and b blue pebbles to lose all its pebbles.

$$p(0,0) = 0, \qquad p(a,0) = p(0,a) = a,$$

$$2p(a,b) = p(a-1,b) + p(a,b-1) + 1 \text{ for } a, b > 0.$$

$$p(1,1) = 1.5, p(2,1) = p(1,2) = 2.25, p(2,2) = 2.75.$$

$\sum p(2,2) + O(1) = 2.75n + O(1)$ questions are always enough.

Potential function p

$p(a, b)$ measures number of questions needed in average for a team with a red and b blue pebbles to lose all its pebbles.

$$p(0,0) = 0, \qquad p(a,0) = p(0,a) = a,$$

$$2p(a,b) = p(a-1,b) + p(a,b-1) + 1 \text{ for } a, b > 0.$$

$$p(1,1) = 1.5, p(2,1) = p(1,2) = 2.25, p(2,2) = 2.75.$$

$\sum p(2,2) + O(1) = 2.75n + O(1)$ questions are always enough.

Not optimal!

CONGRATULATIONS, YOU HAVE MADE IT HALF WAY THROUGH THE TALK!

For myself: I should check how much time is left.

CONGRATULATIONS, YOU HAVE MADE IT THROUGH ONE MORE SLIDE!

For myself: Check time if not done before.

CONGRATULATIONS, YOU HAVE MADE IT THROUGH YET ANOTHER SLIDE!

For myself: Stop this and go on with your talk.

BUT WHY NOW, WE MADE IT THROUGH ANOTHER SLIDE!

BUT WHY NOW, WE MADE IT THROUGH ANOTHER SLIDE!

PEOPLE PREFER THIS TO YOUR PATHETIC TALK...

BUT WHY NOW, WE MADE IT THROUGH ANOTHER SLIDE!

PEOPLE PREFER THIS TO YOUR PATHETIC TALK...

AS YOU WISH...

Main idea of improved algorithm for 1 lie

Main idea of improved algorithm for 1 lie

Natural algorithm cannot be improved without forcing cycles.

Main idea of improved algorithm for 1 lie

Natural algorithm cannot be improved without forcing cycles.

Make a cycle unless $\sum p(a, b)$ decreases by more than one.

Main idea of improved algorithm for 1 lie

Natural algorithm cannot be improved without forcing cycles.

Make a cycle unless $\sum p(a, b)$ decreases by more than one.

Main idea of improved algorithm for 1 lie

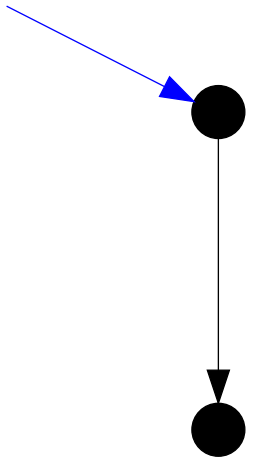
Natural algorithm cannot be improved without forcing cycles.

Make a cycle unless $\sum p(a, b)$ decreases by more than one.

Main idea of improved algorithm for 1 lie

Natural algorithm cannot be improved without forcing cycles.

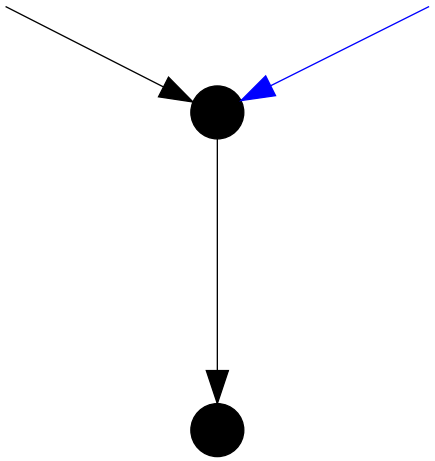
Make a cycle unless $\sum p(a, b)$ decreases by more than one.



Main idea of improved algorithm for 1 lie

Natural algorithm cannot be improved without forcing cycles.

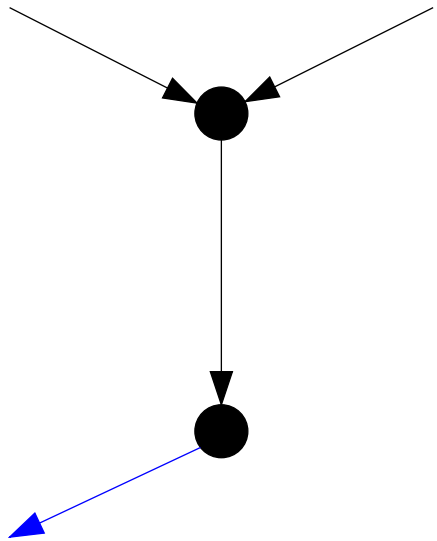
Make a cycle unless $\sum p(a, b)$ decreases by more than one.



Main idea of improved algorithm for 1 lie

Natural algorithm cannot be improved without forcing cycles.

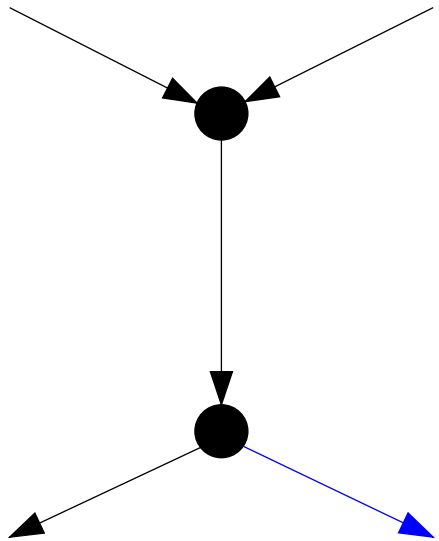
Make a cycle unless $\sum p(a, b)$ decreases by more than one.



Main idea of improved algorithm for 1 lie

Natural algorithm cannot be improved without forcing cycles.

Make a cycle unless $\sum p(a, b)$ decreases by more than one.

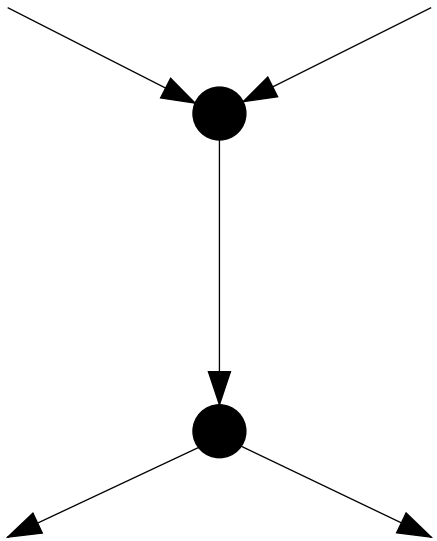


Main idea of improved algorithm for 1 lie

Natural algorithm cannot be improved without forcing cycles.

Make a cycle unless $\sum p(a, b)$ decreases by more than one.

The p of upper and lower team are both 1.



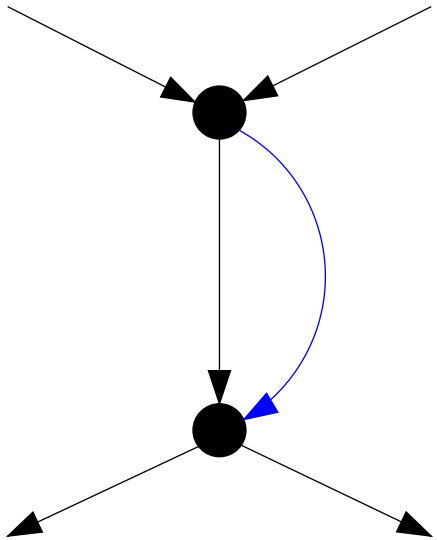
Main idea of improved algorithm for 1 lie

Natural algorithm cannot be improved without forcing cycles.

Make a cycle unless $\sum p(a, b)$ decreases by more than one.

The p of upper and lower team are both 1.

If they play, $\sum p(a, b)$ decreases by 2



Main idea of improved algorithm for 1 lie

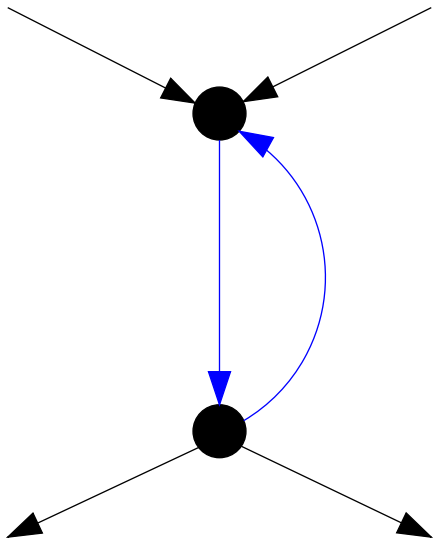
Natural algorithm cannot be improved without forcing cycles.

Make a cycle unless $\sum p(a, b)$ decreases by more than one.

The p of upper and lower team are both 1.

If they play, $\sum p(a, b)$ decreases by 2

or a lie is discovered.



Improved algorithm

Uses $(2.75 - \frac{1}{32})n + \Theta(1)$ questions.

Improved algorithm

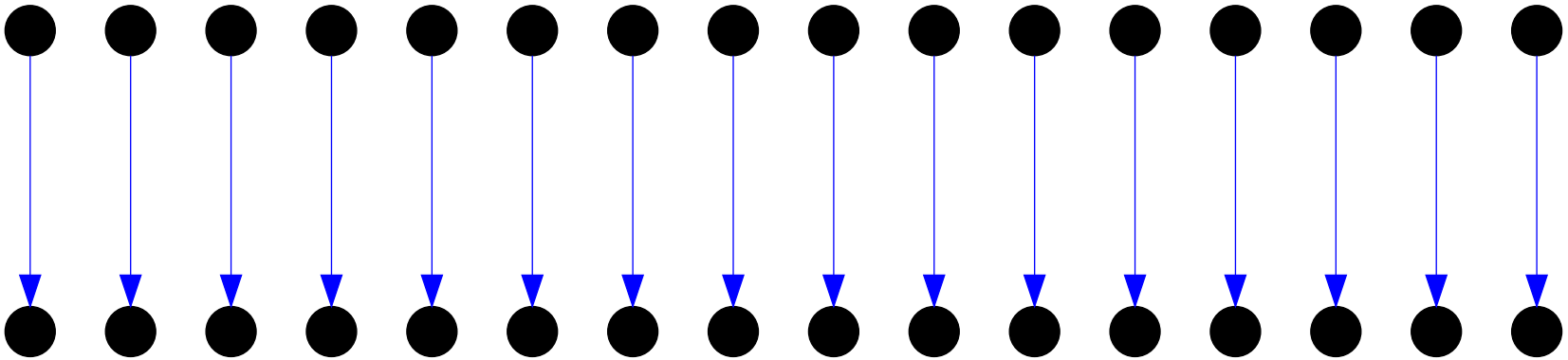
Uses $(2.75 - \frac{1}{32})n + \Theta(1)$ questions.

Groups of 32, spare a comparison in each.

Improved algorithm

Uses $(2.75 - \frac{1}{32})n + \Theta(1)$ questions.

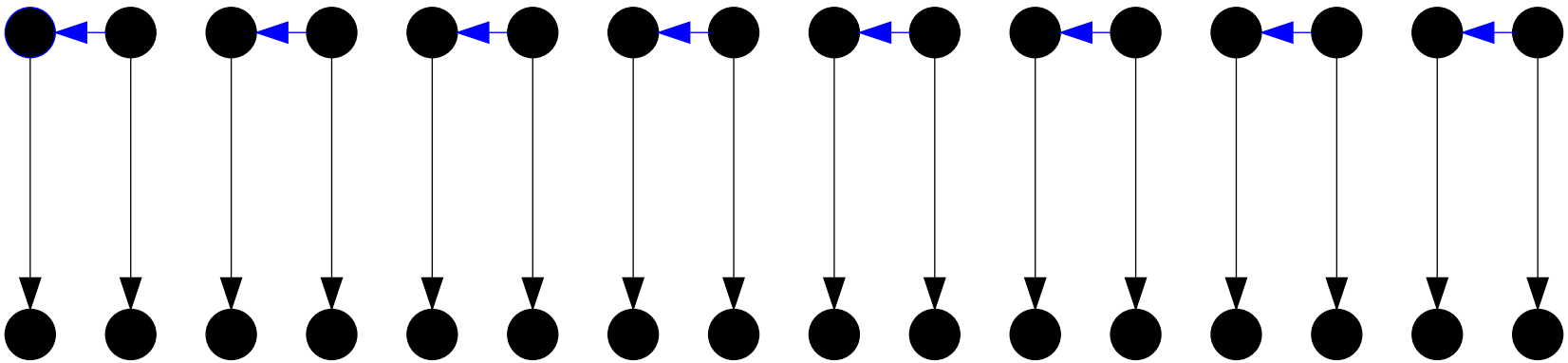
Groups of 32, spare a comparison in each.



Improved algorithm

Uses $(2.75 - \frac{1}{32})n + \Theta(1)$ questions.

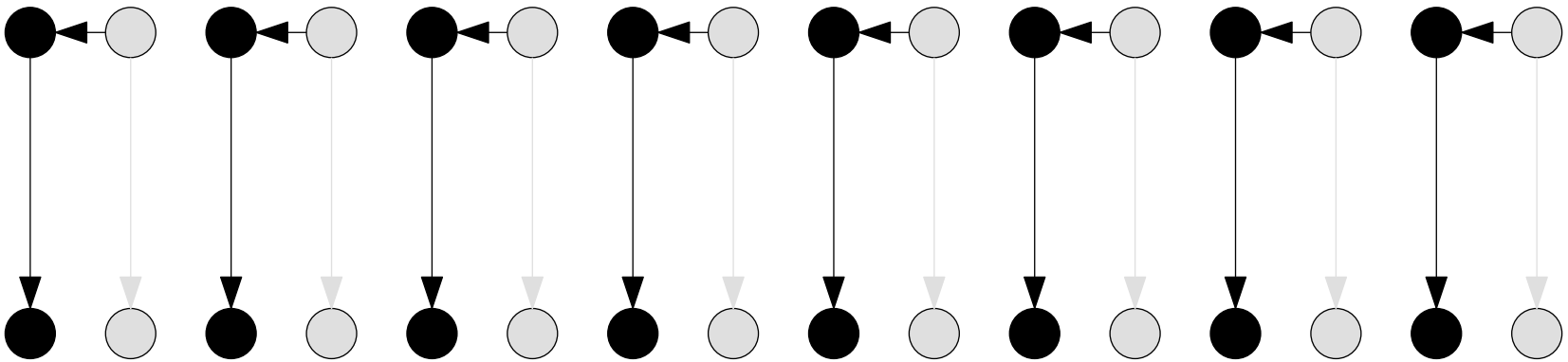
Groups of 32, spare a comparison in each.



Improved algorithm

Uses $(2.75 - \frac{1}{32})n + \Theta(1)$ questions.

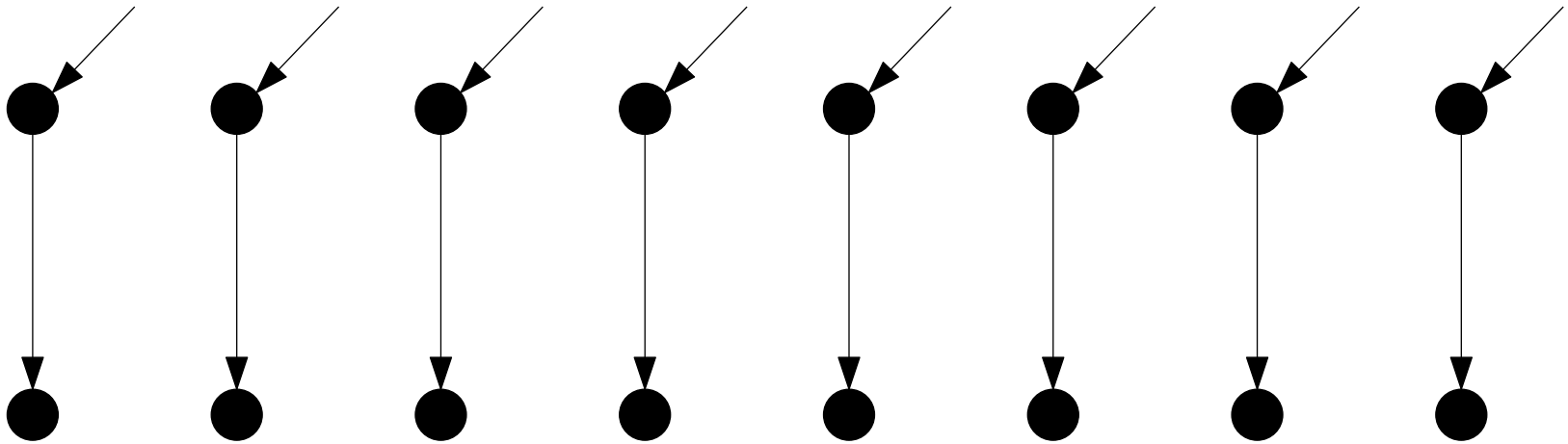
Groups of 32, spare a comparison in each.



Improved algorithm

Uses $(2.75 - \frac{1}{32})n + \Theta(1)$ questions.

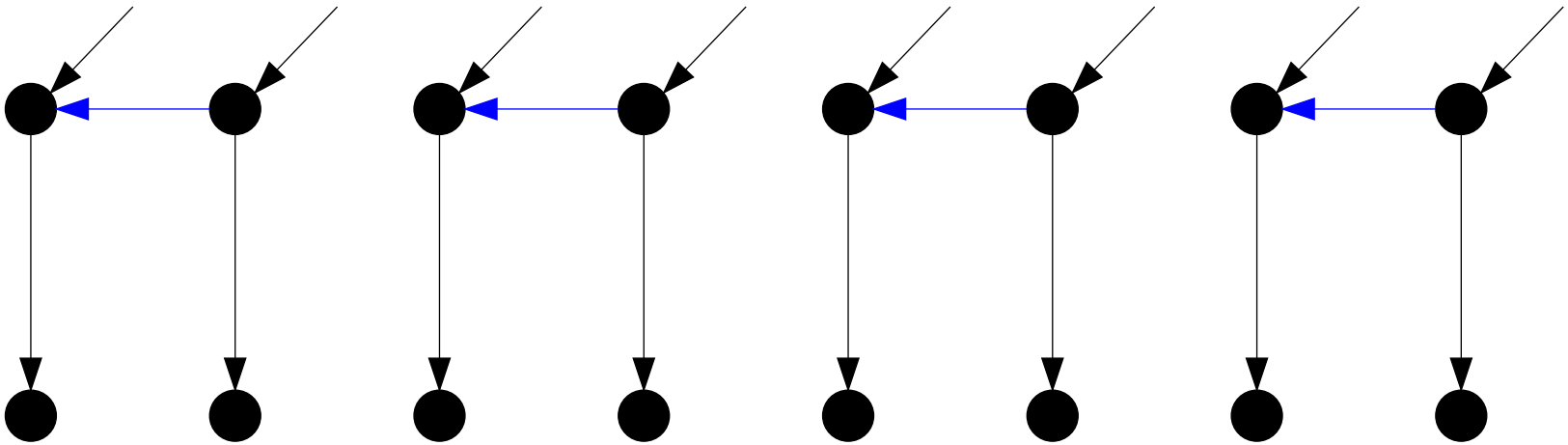
Groups of 32, spare a comparison in each.



Improved algorithm

Uses $(2.75 - \frac{1}{32})n + \Theta(1)$ questions.

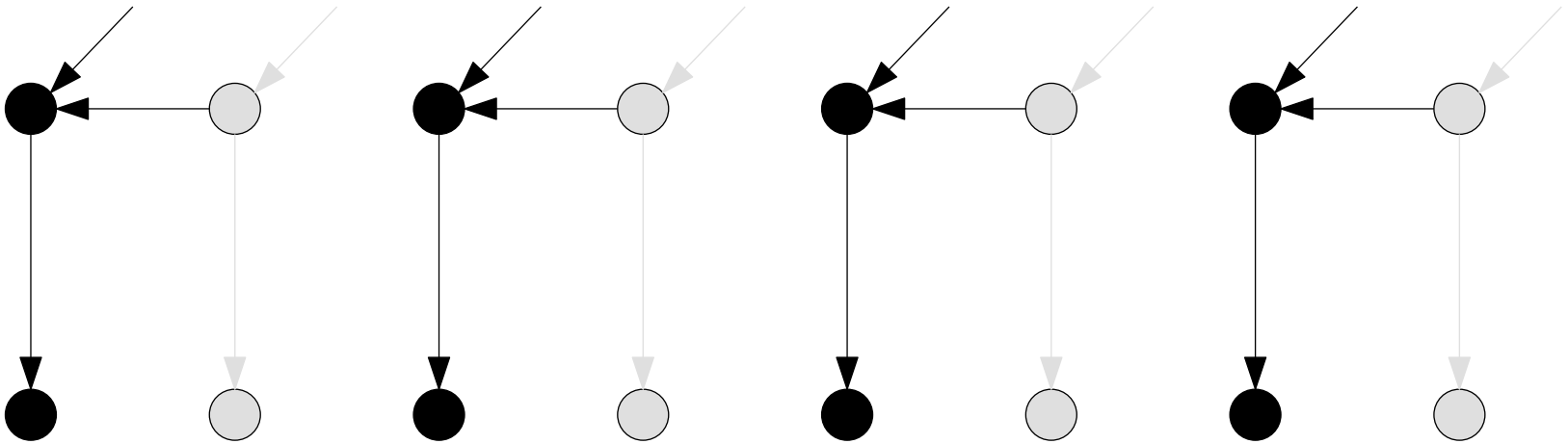
Groups of 32, spare a comparison in each.



Improved algorithm

Uses $(2.75 - \frac{1}{32})n + \Theta(1)$ questions.

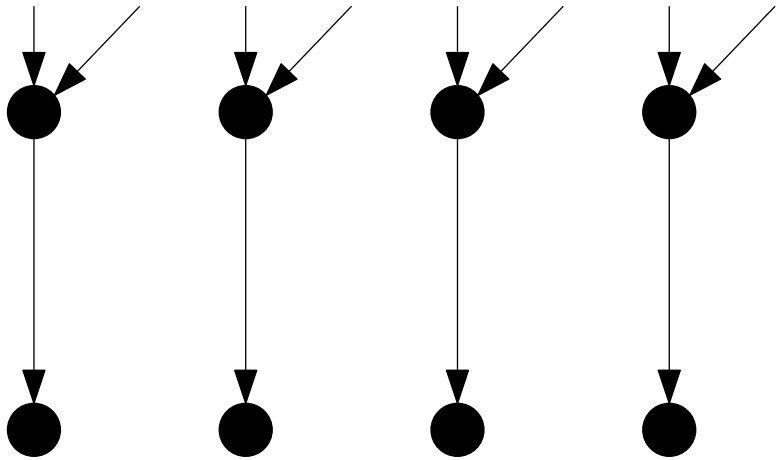
Groups of 32, spare a comparison in each.



Improved algorithm

Uses $(2.75 - \frac{1}{32})n + \Theta(1)$ questions.

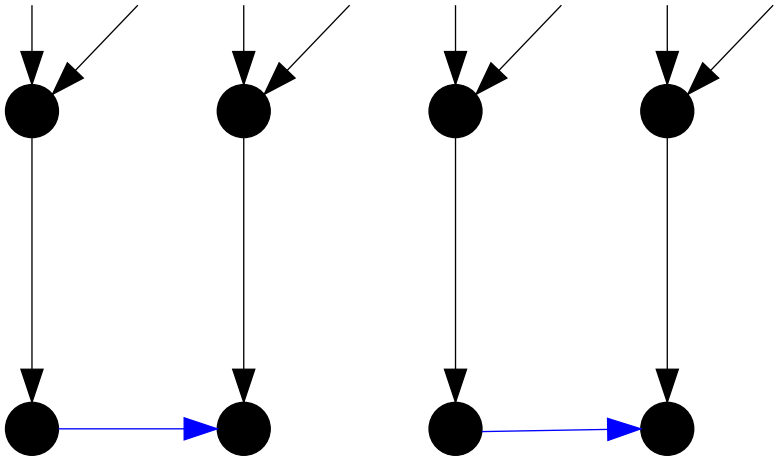
Groups of 32, spare a comparison in each.



Improved algorithm

Uses $(2.75 - \frac{1}{32})n + \Theta(1)$ questions.

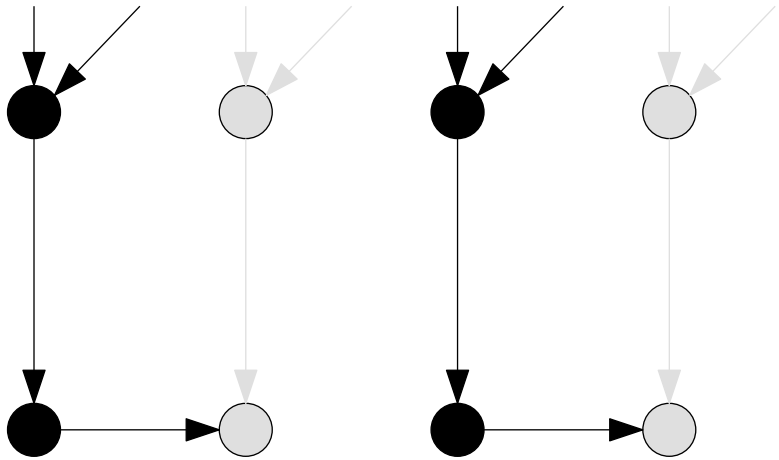
Groups of 32, spare a comparison in each.



Improved algorithm

Uses $(2.75 - \frac{1}{32})n + \Theta(1)$ questions.

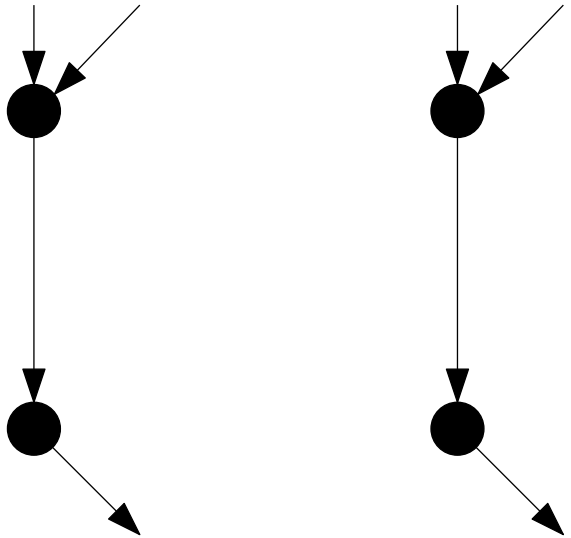
Groups of 32, spare a comparison in each.



Improved algorithm

Uses $(2.75 - \frac{1}{32})n + \Theta(1)$ questions.

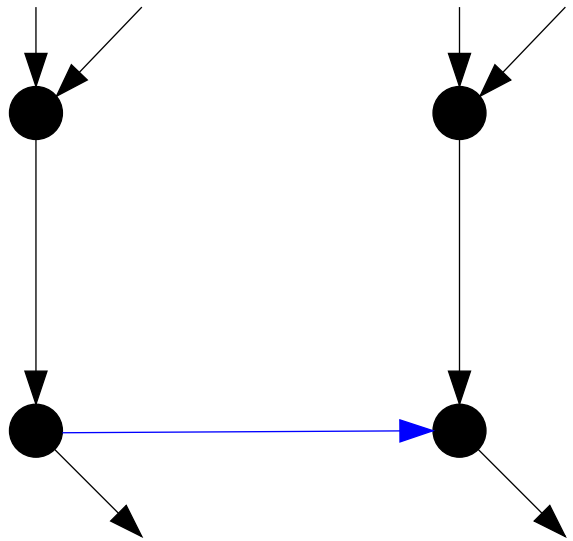
Groups of 32, spare a comparison in each.



Improved algorithm

Uses $(2.75 - \frac{1}{32})n + \Theta(1)$ questions.

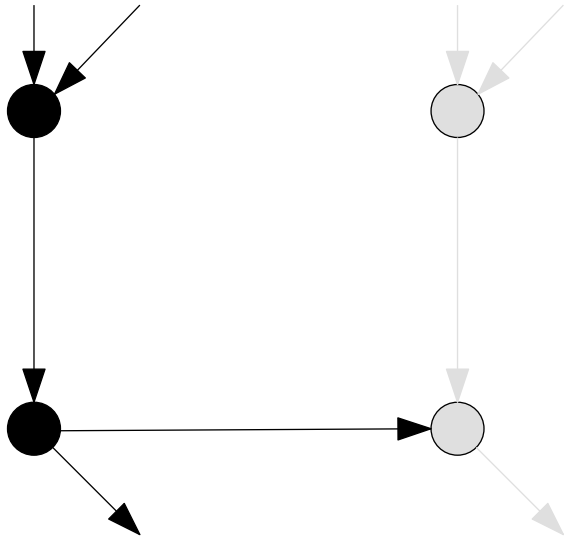
Groups of 32, spare a comparison in each.



Improved algorithm

Uses $(2.75 - \frac{1}{32})n + \Theta(1)$ questions.

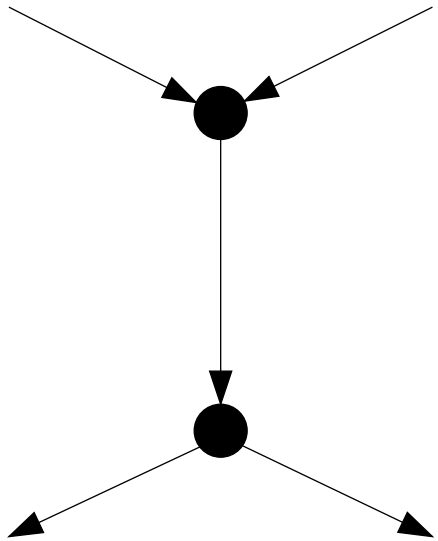
Groups of 32, spare a comparison in each.



Improved algorithm

Uses $(2.75 - \frac{1}{32})n + \Theta(1)$ questions.

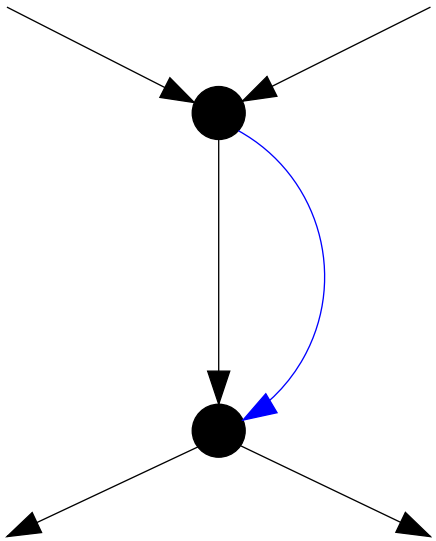
Groups of 32, spare a comparison in each.



Improved algorithm

Uses $(2.75 - \frac{1}{32})n + \Theta(1)$ questions.

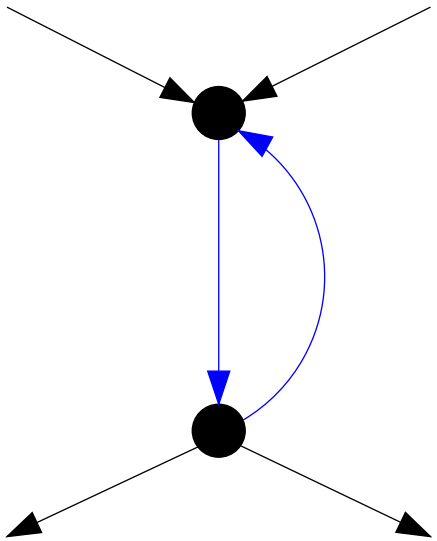
Groups of 32, spare a comparison in each.



Improved algorithm

Uses $(2.75 - \frac{1}{32})n + \Theta(1)$ questions.

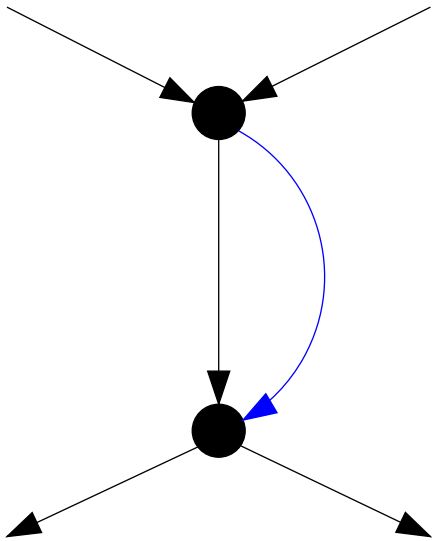
Groups of 32, spare a comparison in each.



Improved algorithm

Uses $(2.75 - \frac{1}{32})n + \Theta(1)$ questions.

Groups of 32, spare a comparison in each.



Modified potential

To obtain a matching lower bound, modify p.

Modified potential

To obtain a matching lower bound, modify p.

$$c(x,y) = \begin{cases} 2^{-5} & \text{if } x = y \text{ and have not played yet,} \\ 0 & \text{if } x \text{ was not the first opponent of } y, \\ 2^{losses(x)+wins(y)-4} & \text{if } x \text{ beat } y \text{ in their first game,} \\ & \text{and } wins(x) = 1 \text{ and } losses(y) = 1. \end{cases}$$

Examples. If in their first game x beats y: $c(x,y) = 2^{-4}$.
If later x loses twice and y wins twice, then $c(x,y) = 1$.

Adversary's strategy

Adversary's strategy

Champion's League - Teams who have won two games and were beaten only by teams in Champion's League.

Adversary's strategy

Champion's League - Teams who have won two games and were beaten only by teams in Champion's League.

Second Division - Teams who have lost two games and won only against teams in Second Division.

Adversary's strategy

Champion's League - Teams who have won two games and were beaten only by teams in Champion's League.

Second Division - Teams who have lost two games and won only against teams in Second Division.

Active - If not in Champion's League or Second Division.

Adversary's strategy

Champion's League - Teams who have won two games and were beaten only by teams in Champion's League.
Second Division - Teams who have lost two games and won only against teams in Second Division.
Active - If not in Champion's League or Second Division.

Adversary's strategy:

Adversary's strategy

Champion's League - Teams who have won two games and were beaten only by teams in Champion's League.
Second Division - Teams who have lost two games and won only against teams in Second Division.
Active - If not in Champion's League or Second Division.

Adversary's strategy:

1. Never lie.

Adversary's strategy

Champion's League - Teams who have won two games and were beaten only by teams in Champion's League.
Second Division - Teams who have lost two games and won only against teams in Second Division.
Active - If not in Champion's League or Second Division.

Adversary's strategy:

1. Never lie.
2. Decrease $\sum p(x) - c(x, y)$ by at most 1 after each question.

Adversary's strategy

Champion's League - Teams who have won two games and were beaten only by teams in Champion's League.
Second Division - Teams who have lost two games and won only against teams in Second Division.
Active - If not in Champion's League or Second Division.

Adversary's strategy:

1. Never lie.
2. Decrease $\sum p(x) - c(x, y)$ by at most 1 after each question.
3. Every opponent but the first of each active team is inactive.

Adversary's strategy

Champion's League - Teams who have won two games and were beaten only by teams in Champion's League.
Second Division - Teams who have lost two games and won only against teams in Second Division.
Active - If not in Champion's League or Second Division.

Adversary's strategy:

1. Never lie.
2. Decrease $\sum p(x) - c(x, y)$ by at most 1 after each question.
3. Every opponent but the first of each active team is inactive.

Proof is by analyzing a lot of cases.

Adversary's strategy

Champion's League - Teams who have won two games and were beaten only by teams in Champion's League.
Second Division - Teams who have lost two games and won only against teams in Second Division.
Active - If not in Champion's League or Second Division.

Adversary's strategy:

1. Never lie.
2. Decrease $\sum p(x) - c(x, y)$ by at most 1 after each question.
3. Every opponent but the first of each active team is inactive.

Proof is by analyzing a lot of cases.

Theorem (Gerbner, P., Patkós, Wiener)

$\boxed{\text{For } k = 1} = \frac{87}{32}n + \Theta(1)$.

Natural algorithm for k lies

Natural algorithm for k lies

Give every team $k+1$ red and $k+1$ blue pebbles.

Natural algorithm for k lies

Give every team $k+1$ red and $k+1$ blue pebbles.

Match teams with the same set of pebbles until possible.

Natural algorithm for k lies

Give every team $k+1$ red and $k+1$ blue pebbles.

Match teams with the same set of pebbles until possible.

After that only $\Theta_k(1)$ teams remain.

Natural algorithm for k lies

Give every team $k+1$ red and $k+1$ blue pebbles.

Match teams with the same set of pebbles until possible.

After that only $\Theta_k(1)$ teams remain.

Using same potential: $p(a,a) = a(1+\binom{2a}{a}2^{-2a})$.

Natural algorithm for k lies

Give every team $k+1$ red and $k+1$ blue pebbles.

Match teams with the same set of pebbles until possible.

After that only $\Theta_k(1)$ teams remain.

Using same potential: $p(a,a) = a(1+\binom{2a}{a}2^{-2a})$.

So $(k+\Theta(\sqrt{k}))n+\Theta_k(1)$ questions suffice.

Natural algorithm for k lies

Give every team $k+1$ red and $k+1$ blue pebbles.

Match teams with the same set of pebbles until possible.

After that only $\Theta_k(1)$ teams remain.

Using same potential: $p(a,a) = a(1+\binom{2a}{a}2^{-2a})$.

So $(k+\Theta(\sqrt{k}))n+\Theta_k(1)$ questions suffice.

Not optimal.

Main idea of improvement

Main idea of improvement

Form n/k groups of size k (national championships).

Main idea of improvement

Form n/k groups of size k (national championships).

Sort them - $k \log k \times n/k$

Main idea of improvement

Form n/k groups of size k (national championships).

Sort them - $k \log k \times n/k$

Make sure this is true order - $(k+1)(k-1) \times n/k$

Main idea of improvement

Form n/k groups of size k (national championships).

Sort them - $k \log k \times n/k$

Make sure this is true order - $(k+1)(k-1) \times n/k$

Select best from bests and worst from worsts - $2 \times (k+1)n/k$

Main idea of improvement

Form n/k groups of size k (national championships).

Sort them - $k \log k \times n/k$

Make sure this is true order - $(k+1)(k-1) \times n/k$

Select best from bests and worst from worsts - $2 \times (k+1)n/k$

Total about $(k + \log k)n$.

Better improvement

Better improvement

If sorting algorithm is good, no need for extra $(k+1)(k-1)$ to confirm.

Better improvement

If sorting algorithm is good, no need for extra $(k+1)(k-1)$ to confirm.

First find median, then quicksort with median is good.

Better improvement

If sorting algorithm is good, no need for extra $(k+1)(k-1)$ to confirm.

First find median, then quicksort with median is good.

Theorem (Hoffmann, Matoušek, Okamoto, Zumstein)

For general k $\leq (k+1+10)n + \Theta_k(1)$.

Better improvement

If sorting algorithm is good, no need for extra $(k+1)(k-1)$ to confirm.

First find median, then quicksort with median is good.

Theorem (Hoffmann, Matoušek, Okamoto, Zumstein)

$\boxed{\textit{For general } k} \leq (k+1+10)n + \Theta_k(1)$.

Does "10" $\to 0$ as $k \to \infty$?

No

No

Theorem (P.)

For general k $\geq (k + 1.5)n + \Theta_k(1)$.

No

Theorem (P.)

For general k $\geq (k + 1.5)n + \Theta_k(1)$.

Proof.

Suppose n is even and first $n/2$ comparisons are perfect matching.

No

Theorem (P.)

For general k $\geq (k + 1.5)n + \Theta_k(1)$.

Proof.

Suppose n is even and first $n/2$ comparisons are perfect matching.

First round winners will always beat first round losers.

No

Theorem (P.)

For general k $\geq (k + 1.5)n + \Theta_k(1)$.

Proof.

Suppose n is even and first $n/2$ comparisons are perfect matching.

First round winners will always beat first round losers.

Problem reduces to find maximum from $n/2$ elements and minimum from $n/2$ other elements.

No

Theorem (P.)

For general k $\geq (k + 1.5)n + \Theta_k(1)$.

Proof.

Suppose n is even and first $n/2$ comparisons are perfect matching.

First round winners will always beat first round losers.

Problem reduces to find maximum from $n/2$ elements and minimum from $n/2$ other elements.

So we need $2 \times (k + 1)(n/2 - 1)$ more comparisons.

No

Theorem (P.)

For general k $\geq (k+1.5)n + \Theta_k(1)$.

Proof.

Suppose n is even and first $n/2$ comparisons are perfect matching.

First round winners will always beat first round losers.

Problem reduces to find maximum from $n/2$ elements and minimum from $n/2$ other elements.

So we need $2 \times (k+1)(n/2-1)$ more comparisons.

Total at least $(k+1.5)(n-1)$ comparisons. □

Remarks

We can also use general method to get $+k$ in the proof.

Remarks

We can also use general method to get $+k$ in the proof.

This gives lower bound of $\lceil (k + 1.5)n \rceil - k - 2$.

Remarks

We can also use general method to get $+k$ in the proof.

This gives lower bound of $\lceil (k + 1.5)n \rceil - k - 2$.

Optimal for $k = 0$

Remarks

We can also use general method to get $+k$ in the proof.

This gives lower bound of $\lceil (k + 1.5)n \rceil - k - 2$.

Optimal for $k = 0$ but never again.

Remarks

We can also use general method to get $+k$ in the proof.

This gives lower bound of $\lceil (k+1.5)n \rceil - k - 2$.

Optimal for $k = 0$ but never again.

Does "1.5" monotone increase as k grows?

Remarks

We can also use general method to get $+k$ in the proof.

This gives lower bound of $\lceil (k + 1.5)n \rceil - k - 2$.

Optimal for $k = 0$ but never again.

Does "1.5" monotone increase as k grows?

Does it have a limit?

Remarks

We can also use general method to get $+k$ in the proof.

This gives lower bound of $\lceil (k+1.5)n \rceil - k - 2$.

Optimal for $k = 0$ but never again.

Does "1.5" monotone increase as k grows?

Does it have a limit?

Where is the truth between 1.5 and 10?

Thank you for your attention!

Thank you for your attention!

HAPPY BIRTHDAY GYULA!

Thank you for your attention!

HAPPY BIRTHDAY GYULA!