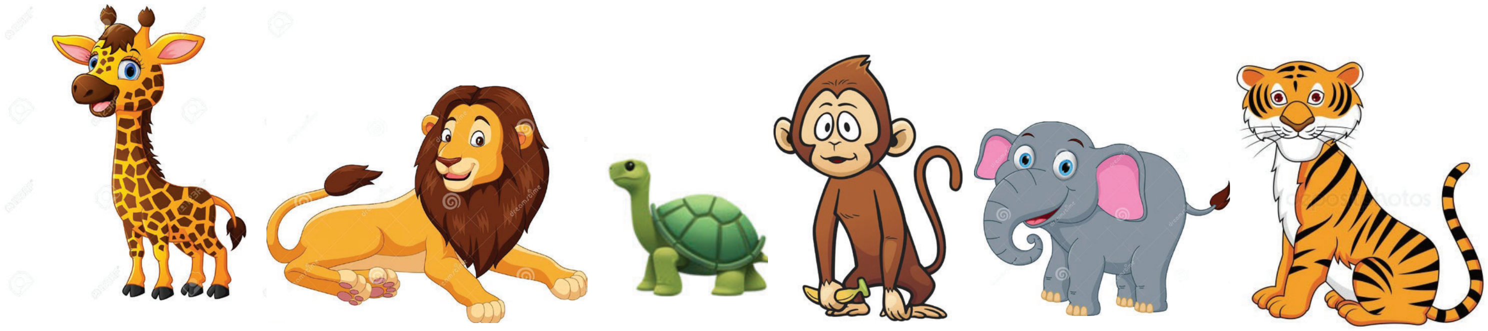
CS 580: Algorithm Design and Analysis

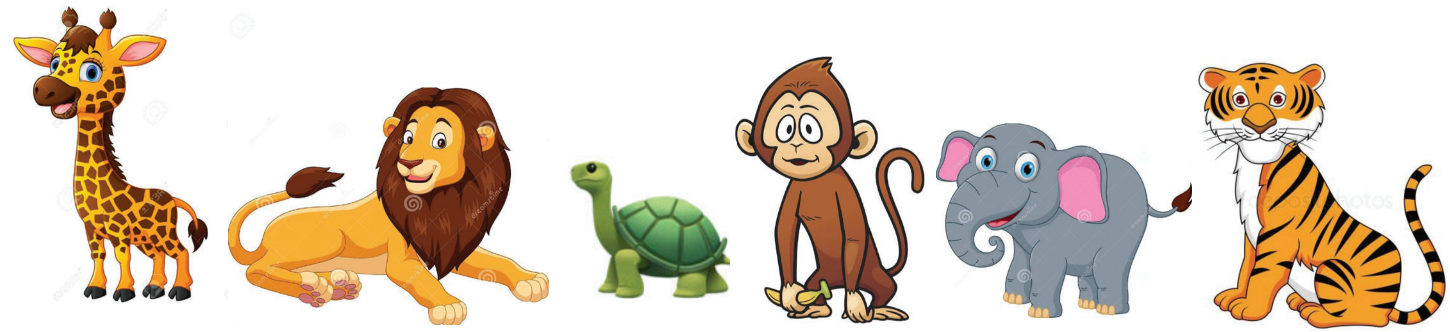
Finding the maximum

Input: 

Goal: find the maximum element (or one of them if multiple ones exist) using comparison queries.

Finding the maximum

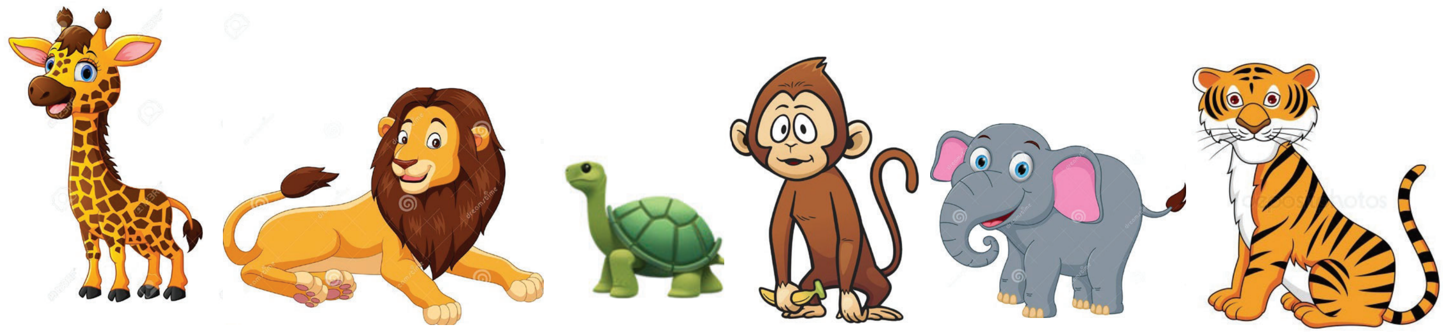
Input:



Goal: find the maximum element (or one of them if multiple ones exist) using comparison queries.

Q: How many comparisons are needed?

Finding the maximum

Input: 

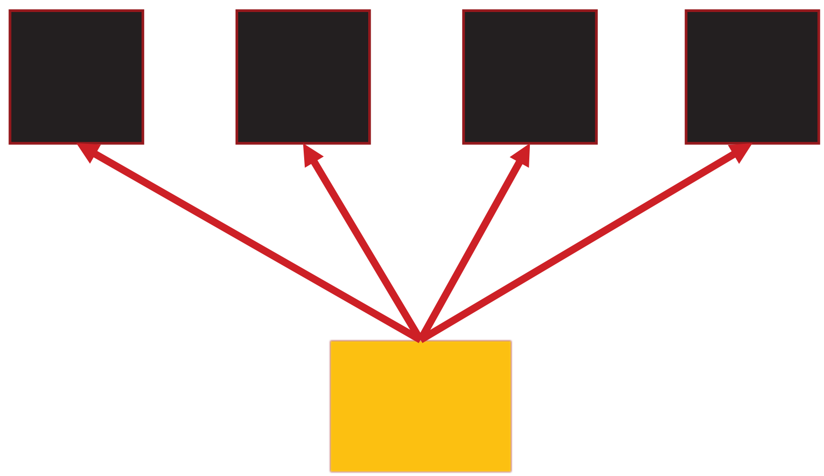
Goal: find the maximum element (or one of them if multiple ones exist) using comparison queries.

Q: How many comparisons are needed?

A: Depends on the number of **rounds** allowed.

Algorithms in rounds

Algorithm that runs in k rounds can also be seen as: central machine issues in each round j a set of queries, one to each processor, then waits for the answers before issuing the next set of parallel queries in round j+1.



Query complexity in k rounds informs how many processors are needed to achieve a parallel time of k.

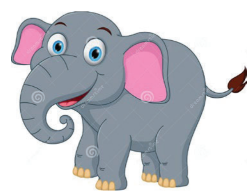
Why rounds?

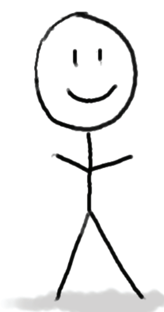
- Minimizing the number of rounds is important when computation is done by many (small) computers that interact over a network – e.g. phones, laptops

- Scenarios where rounds are expensive: crowdsourcing, blockchain

Recall the comparison model

- An algorithm A in the comparison model gets input vector x
- A can do anything except open up the contents of the entries x_i
- Algorithm A has access to an oracle O that, given any query $x_i < x_j$? can return True/False (i.e., O does the work of inspecting the elements)

Is  larger than  ?



Algorithm

NO

Algorithms with rounds

Input: vector $x = (x_1, \ldots, x_n)$

{Internal computation}

Submit set of queries S_1 to O, then get back the answers

{Internal computation}

Submit set of queries S_2 to O, then get back the answers

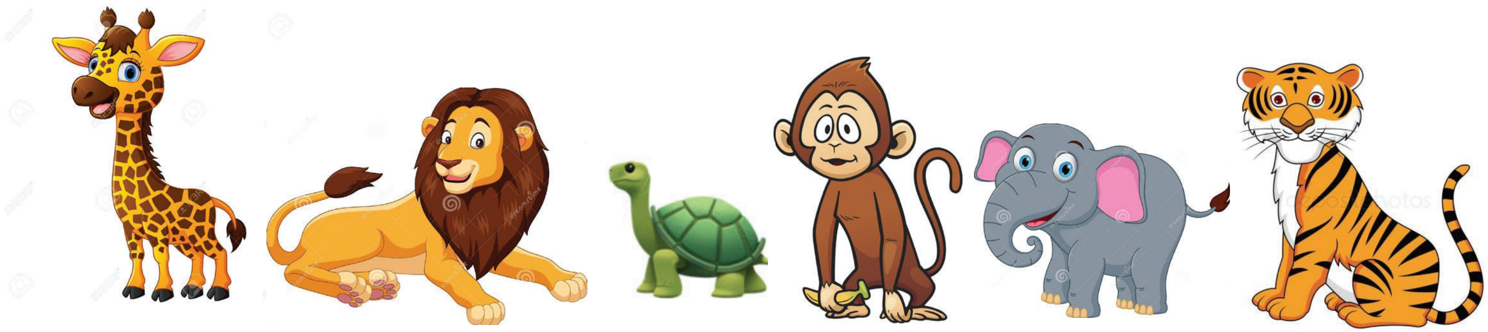
. . .

Submit set of queries S_k to O, then get back the answers

{Internal computation}

Output

Finding the maximum

Input: 

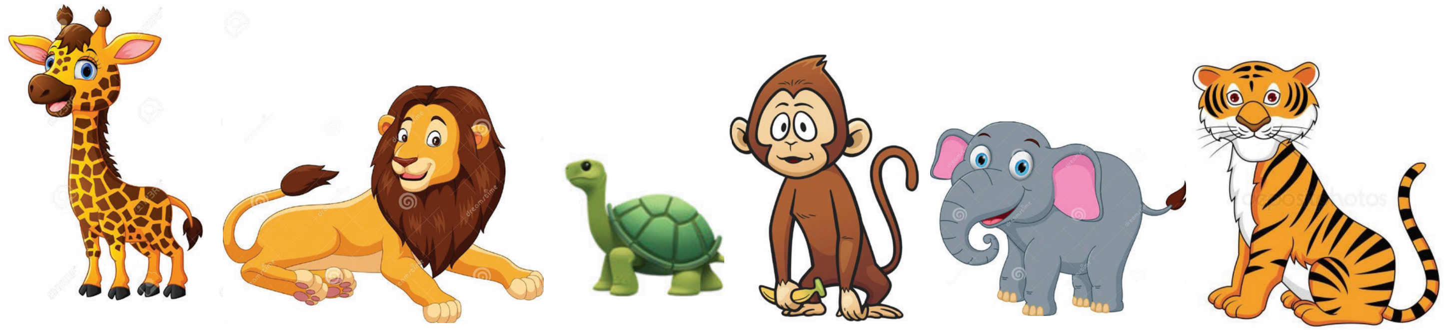
Goal: given input vector x and upper bound k on the allowed number of rounds of interaction, find the maximum element

- using as few comparisons as possible (count total)
- using at most k rounds of interaction with the oracle

Question: Given vector $x = (x_1, \dots, x_n)$ and number $k \in \{1, \dots, n\}$, how many comparisons do we need to find the maximum of x in k rounds?

Finding the maximum

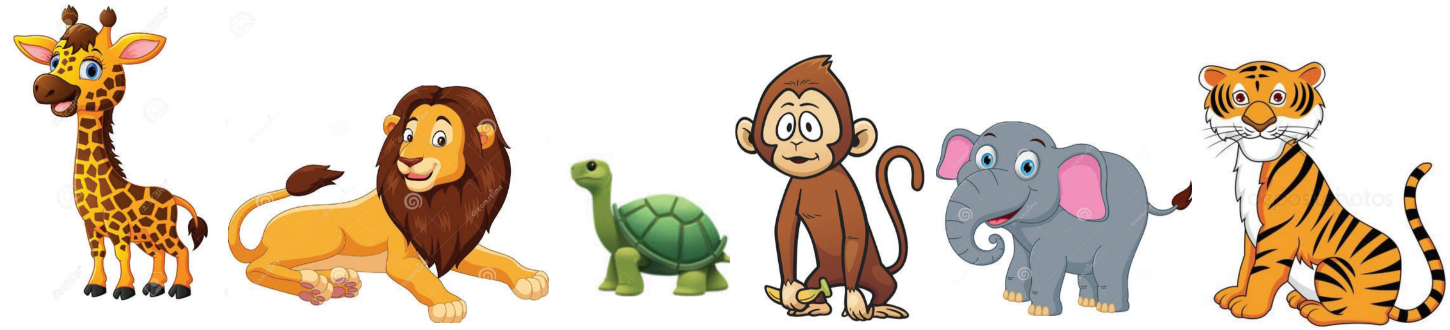
Input:



Unlimited number of rounds (fully adaptive)?

Finding the maximum

Input:



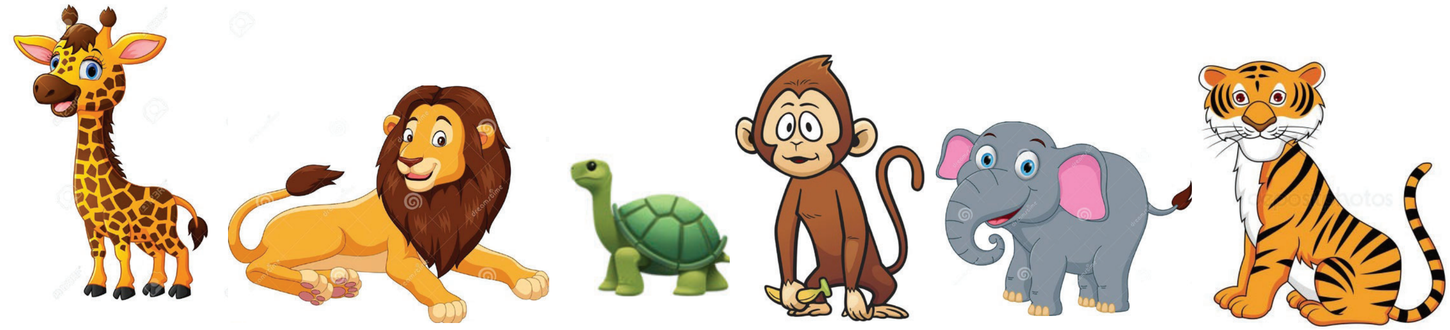
Unlimited number of rounds (fully adaptive):

- **Upper bound:**

- **Lower bound:**

Finding the maximum

Input:

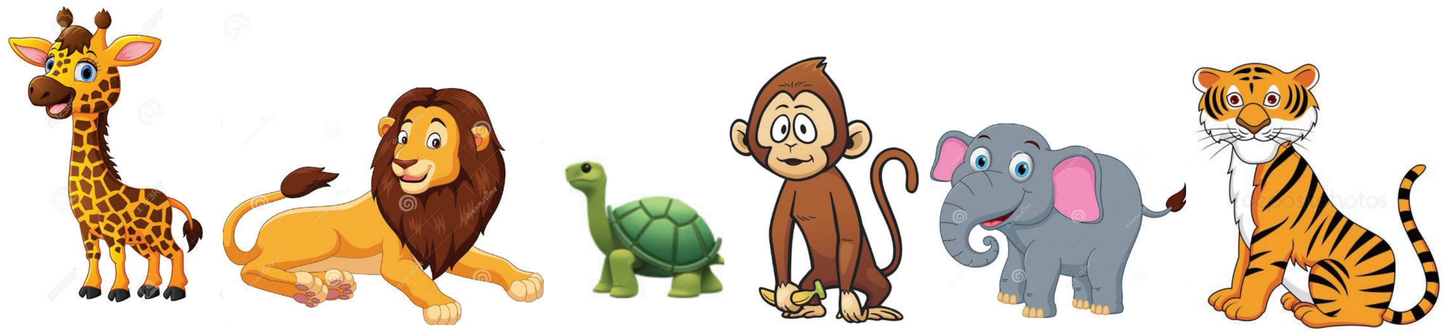


Unlimited number of rounds (fully adaptive):

- **Upper bound:** Go through each element and remember the max seen so far => n-1 comparisons.

- **Lower bound:**

Finding the maximum

Input: 

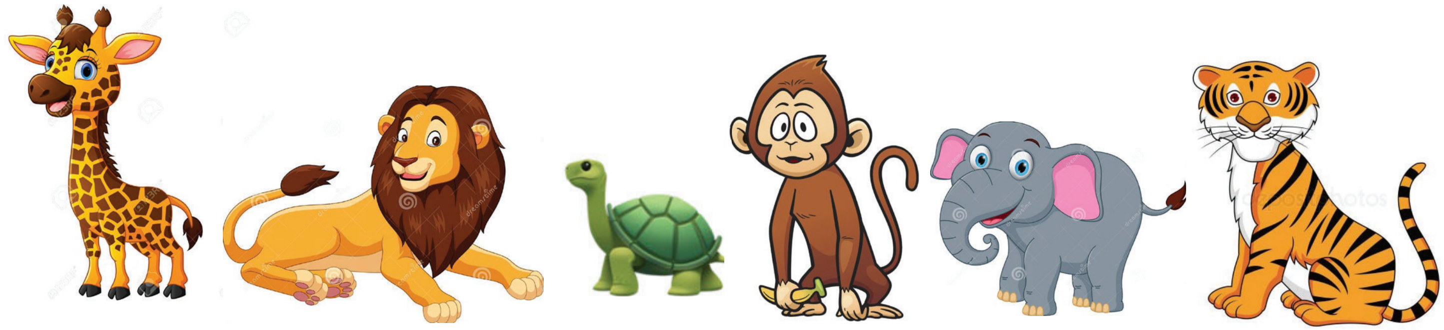
Unlimited number of rounds (fully adaptive):

- **Upper bound:** Go through each element and remember the max seen so far => n-1 comparisons.

- **Lower bound:** If an element is not compared to anything, then it could be the maximum. The comparisons give a graph on n elements that must be connected => need at least n-1 comparisons.

Finding the maximum

Input:



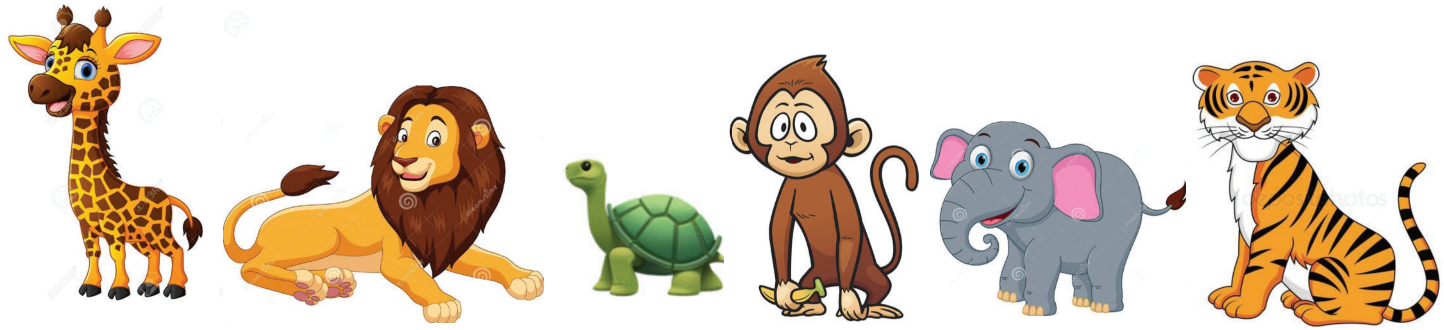
Unlimited number of rounds (fully adaptive): $\Theta(n)$

- **Upper bound:** Go through each element and remember the max seen so far => n-1 comparisons.

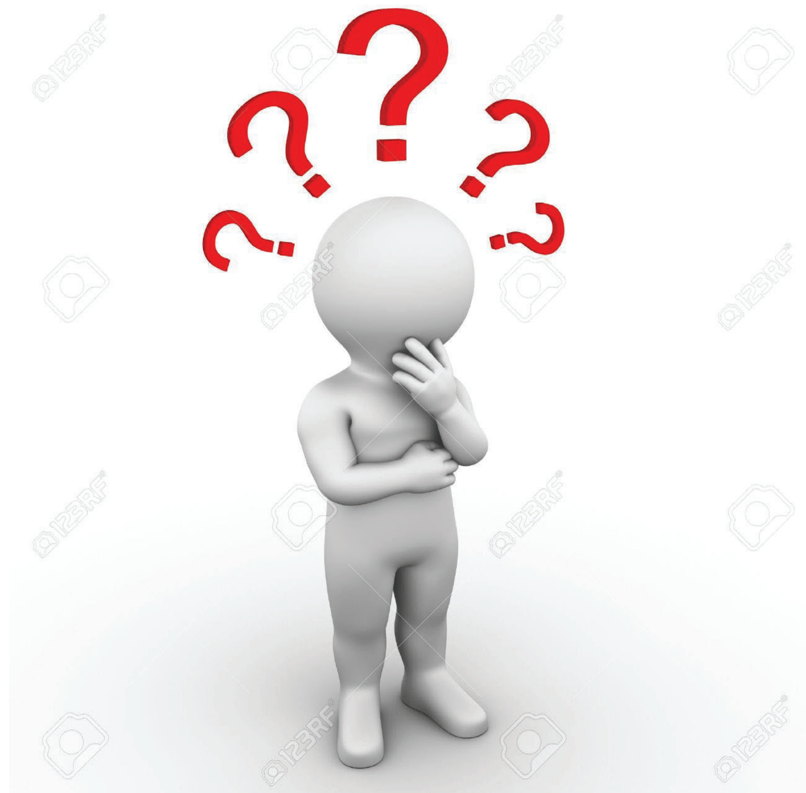
- **Lower bound:** If an element is not compared to anything, then it could be the maximum. The comparisons give a graph on n elements that must be connected => need at least n-1 comparisons.

Finding the maximum

Input:

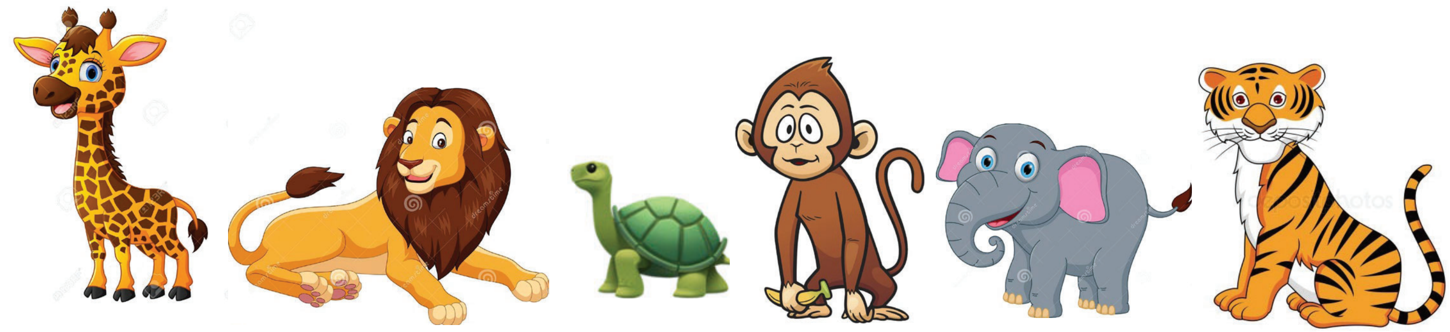


r = 1 rounds: How many comparisons?



Finding the maximum

Input:



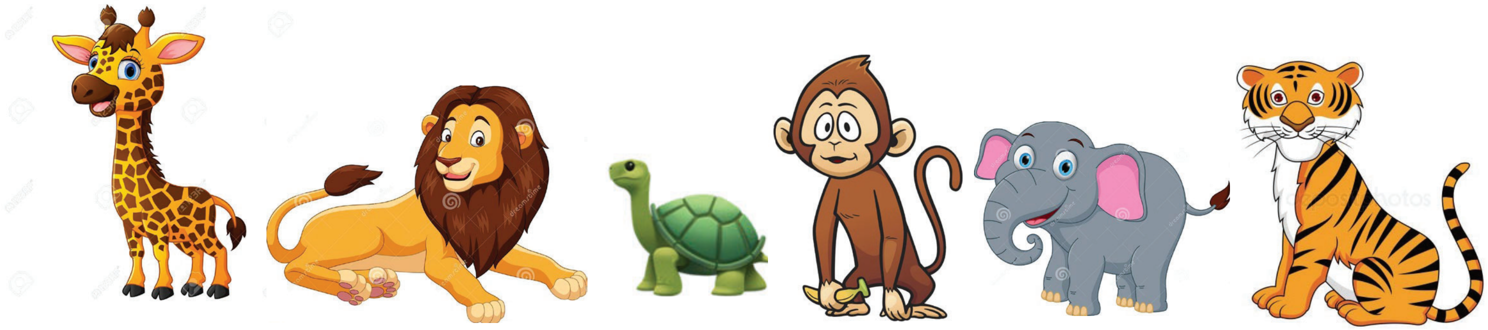
r = 1 rounds:

Upper bound:

Lower bound:

Finding the maximum

Input:

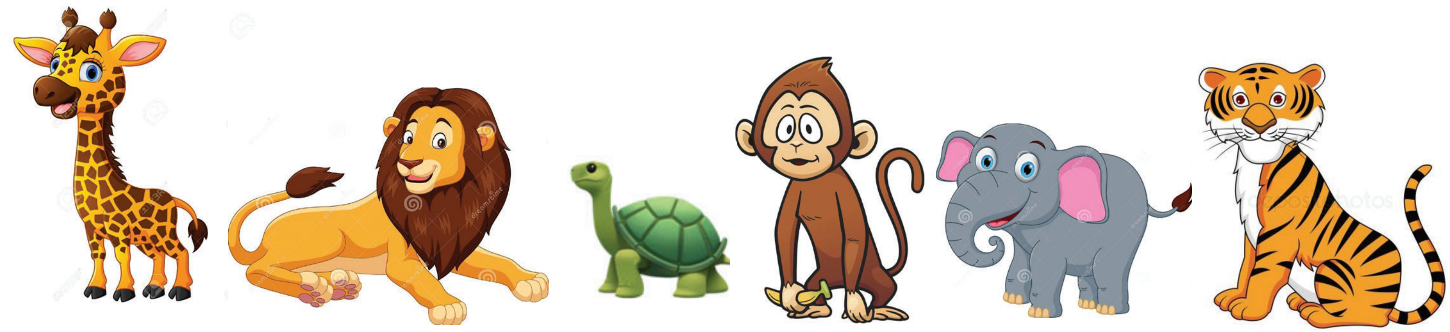


r = 1 rounds:

Upper bound: Ask for all pairwise comparisons $\binom{n}{2}$ and output the maximum after receiving the answers.

Lower bound:

Finding the maximum

Input: 

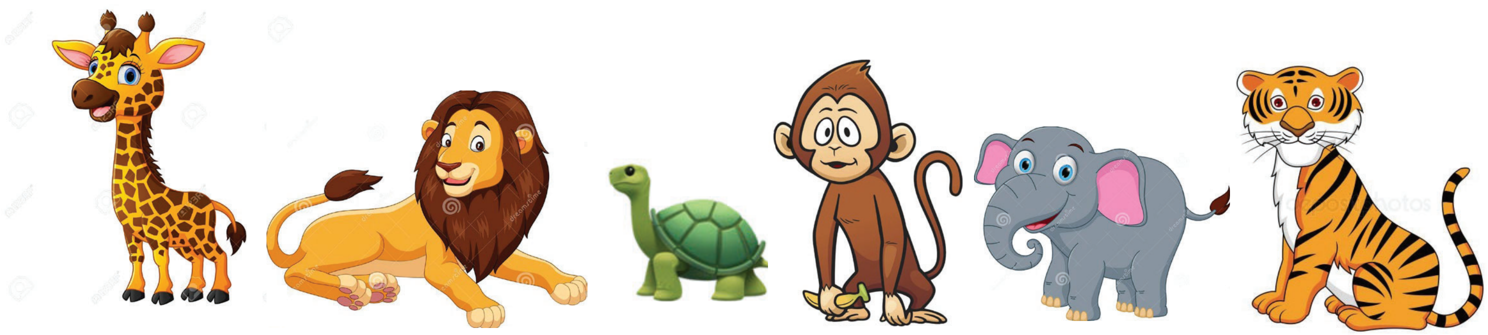
r = 1 rounds:

Upper bound: Ask for all pairwise comparisons $\binom{n}{2}$ and output the maximum after receiving the answers.

Lower bound: If even one comparison is missing, that can be enough to return the wrong maximum:

- Suppose there is a comparison missing between elements x_i and x_j in the input vector x.
- Then adversary can answer the queries so that x_i and x_j are greater than all the other elements -> not enough info to determine the max.

Finding the maximum

Input: 

r = 1 rounds: $\Theta(n^2)$.

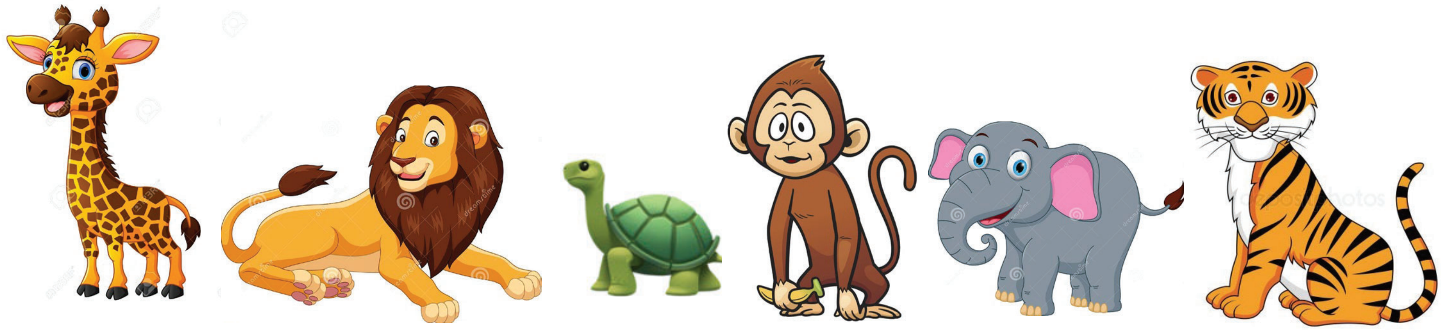
Upper bound: Ask for all pairwise comparisons $\binom{n}{2}$ and output the maximum after receiving the answers.

Lower bound: If even one comparison is missing, that can be enough to return the wrong maximum:

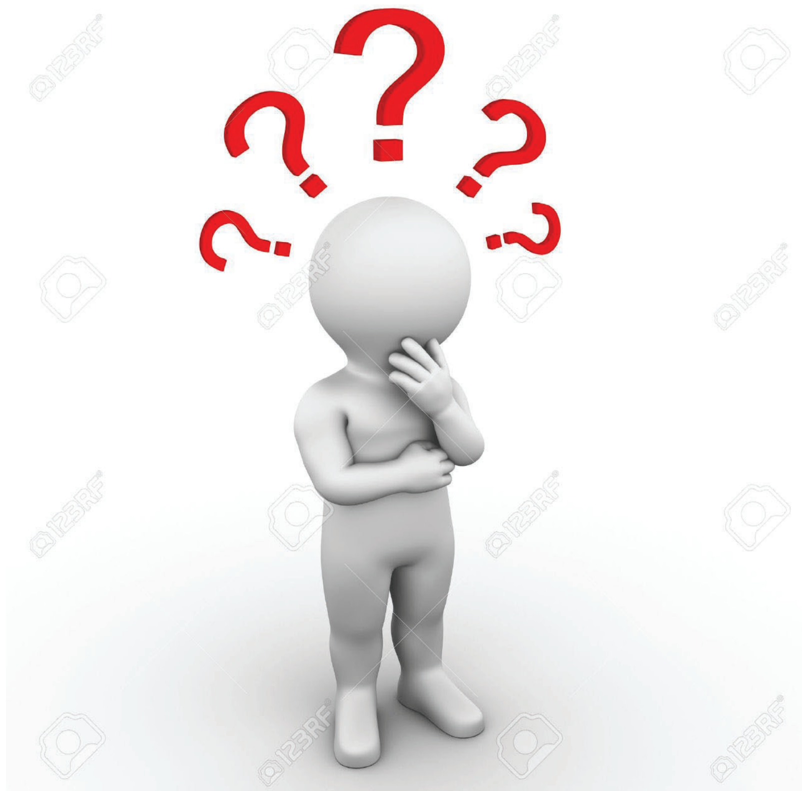
- Suppose there is a comparison missing between elements x_i and x_j in the input vector x.
- Then adversary can answer the queries so that x_i and x_j are greater than all the other elements -> not enough info to determine the max.

Finding the maximum

Input:



r = 2 rounds: How many comparisons?



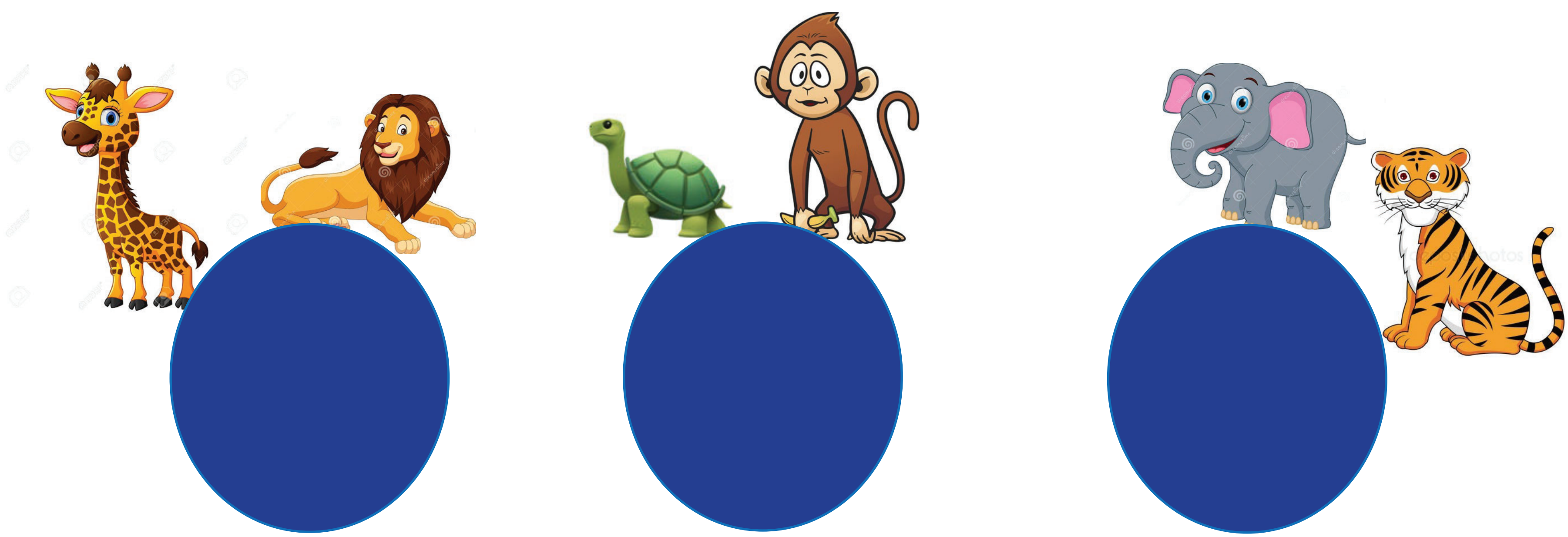
Finding the maximum

r = 2 rounds. ***Upper bound:***

Finding the maximum

r = 2 rounds. ***Upper bound:***

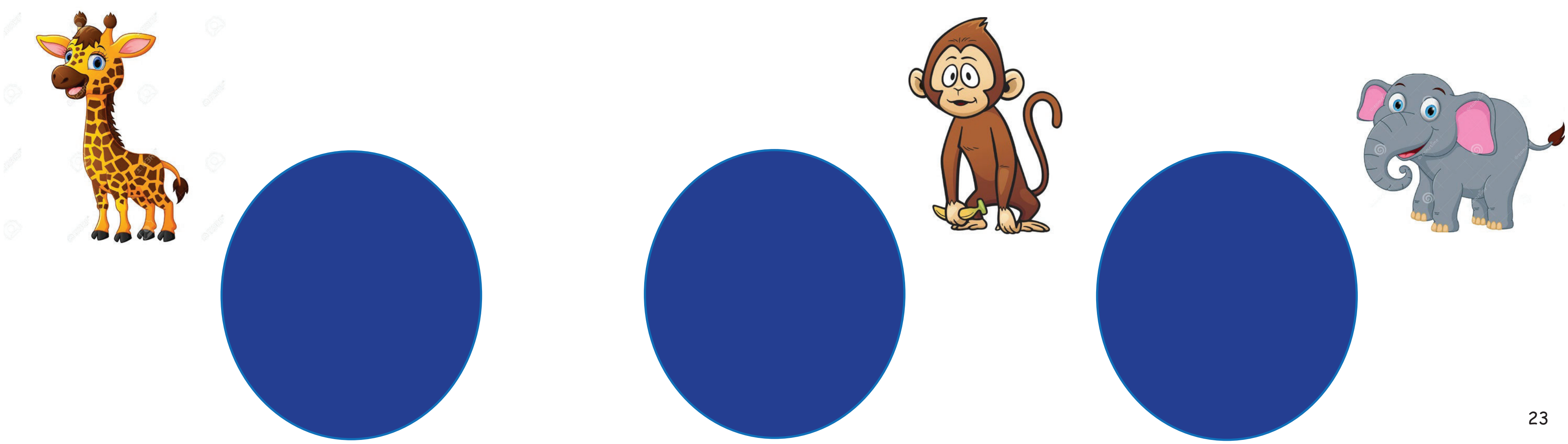
Round 1. Divide the items into k groups of size n/k and select the max from each group => $\binom{n/k}{2}$ comparisons inside each group => at most $k * \frac{n^2}{k^2}$ comparisons overall.



Finding the maximum

r = 2 rounds. ***Upper bound:***

***Round 1*.** Divide the items into k groups of size n/k and select the max from each group => $\binom{n/k}{2}$ comparisons inside each group => at most $k * \frac{n^2}{k^2}$ comparisons overall.

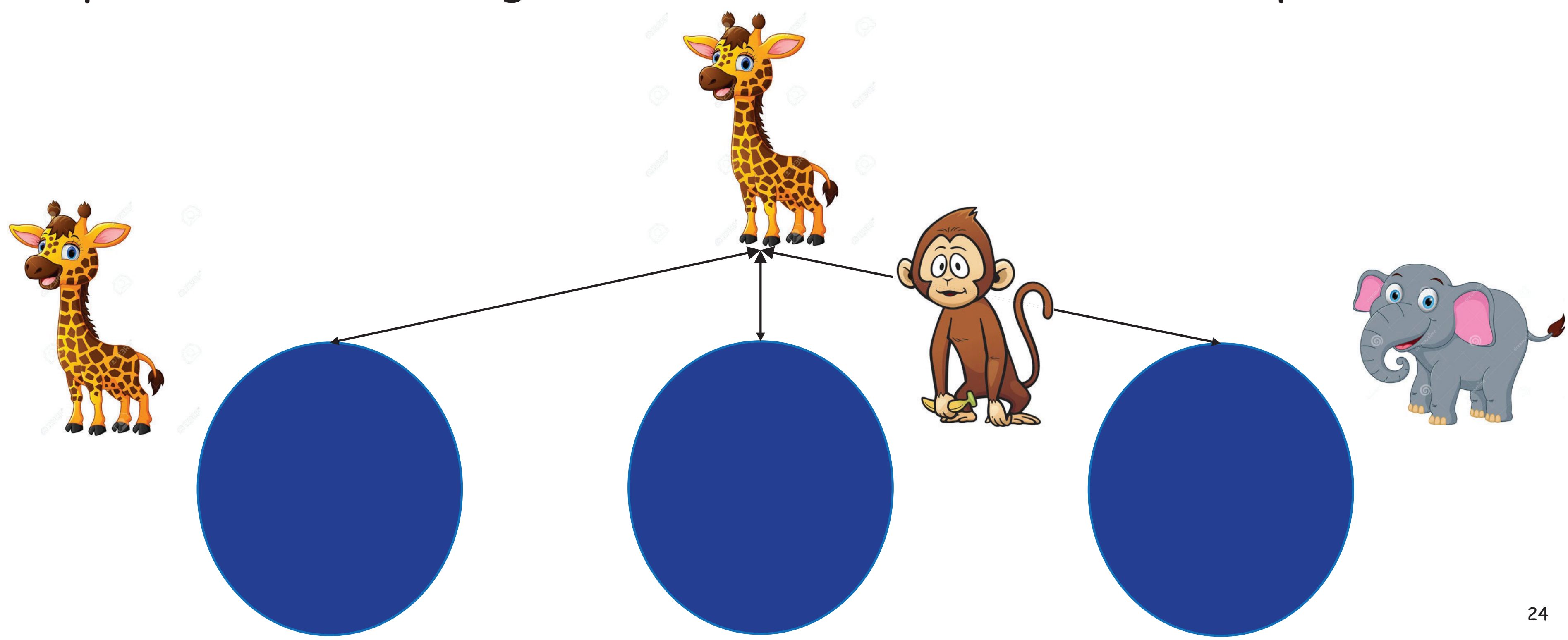


Finding the maximum

r = 2 rounds. ***Upper bound:***

Round 1. Divide the items into k groups of size n/k and select the max from each group => $\binom{n/k}{2}$ comparisons inside each group => at most $k * \frac{n^2}{k^2}$ comparisons overall.

Round 2. Select the maximum among the group maxima => round 1 protocol for finding the max of k elements => k^2 comparisons



Finding the maximum

r = 2 rounds. ***Upper bound:***

Round 1. Divide the items into k groups of size n/k and select the max from each group => $\binom{n/k}{2}$ comparisons inside each group => at most $k * \frac{n^2}{k^2}$ comparisons overall.

Round 2. Select the maximum among the group maxima => round 1 protocol for finding the max of k elements => k^2 comparisons.

The total number of comparisons is at most $k * \frac{n^2}{k^2} + k^2$.

Set k to equalize the work for rounds 1 and 2:

$$k * \frac{n^2}{k^2} = \mathrm{k}^2 \Rightarrow \mathrm{k} = \mathrm{n}^{2/3}.$$

Total number of comparisons is $2 \cdot k^2 = 2 \cdot n^{4/3}$.

Finding the maximum

r = 2 rounds. ***Upper bound:*** $O(n^{\frac{4}{3}})$

Round 1. Divide the items into k groups of size n/k and select the max from each group => $\binom{n/k}{2}$ comparisons inside each group => at most $k * \frac{n^2}{k^2}$ comparisons overall.

Round 2. Select the maximum among the group maxima => round 1 protocol for finding the max of k elements => k^2 comparisons.

The total number of comparisons is at most $k * \frac{n^2}{k^2} + k^2$.

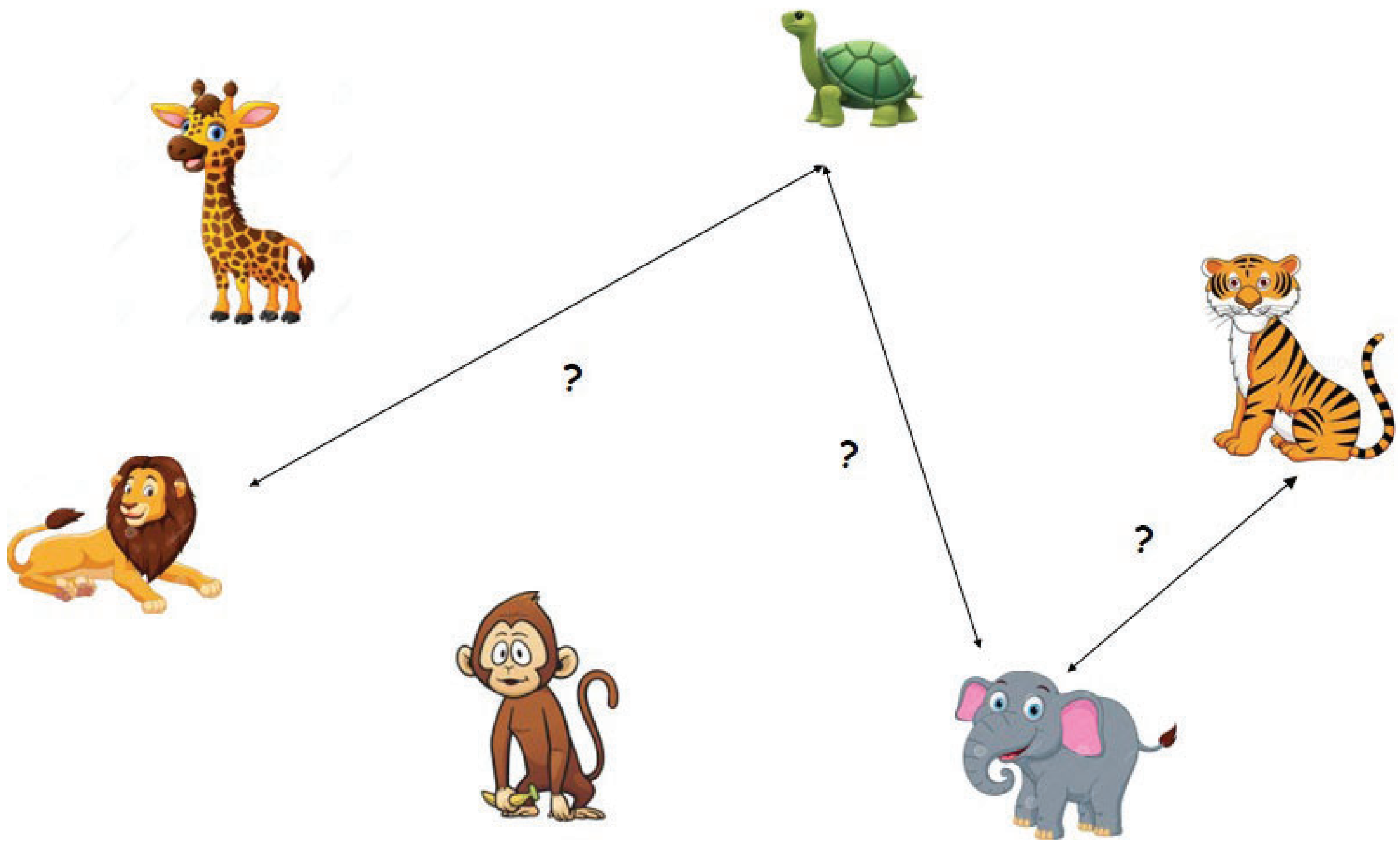
Set k to equalize the work for rounds 1 and 2:

$$k * \frac{n^2}{k^2} = \mathrm{k}^2 \Rightarrow \mathrm{k} = \mathrm{n}^{2/3}.$$

Total number of comparisons is $2 \cdot k^2 = 2 \cdot n^{4/3}$.

Finding the max: lower bound in 2 rounds

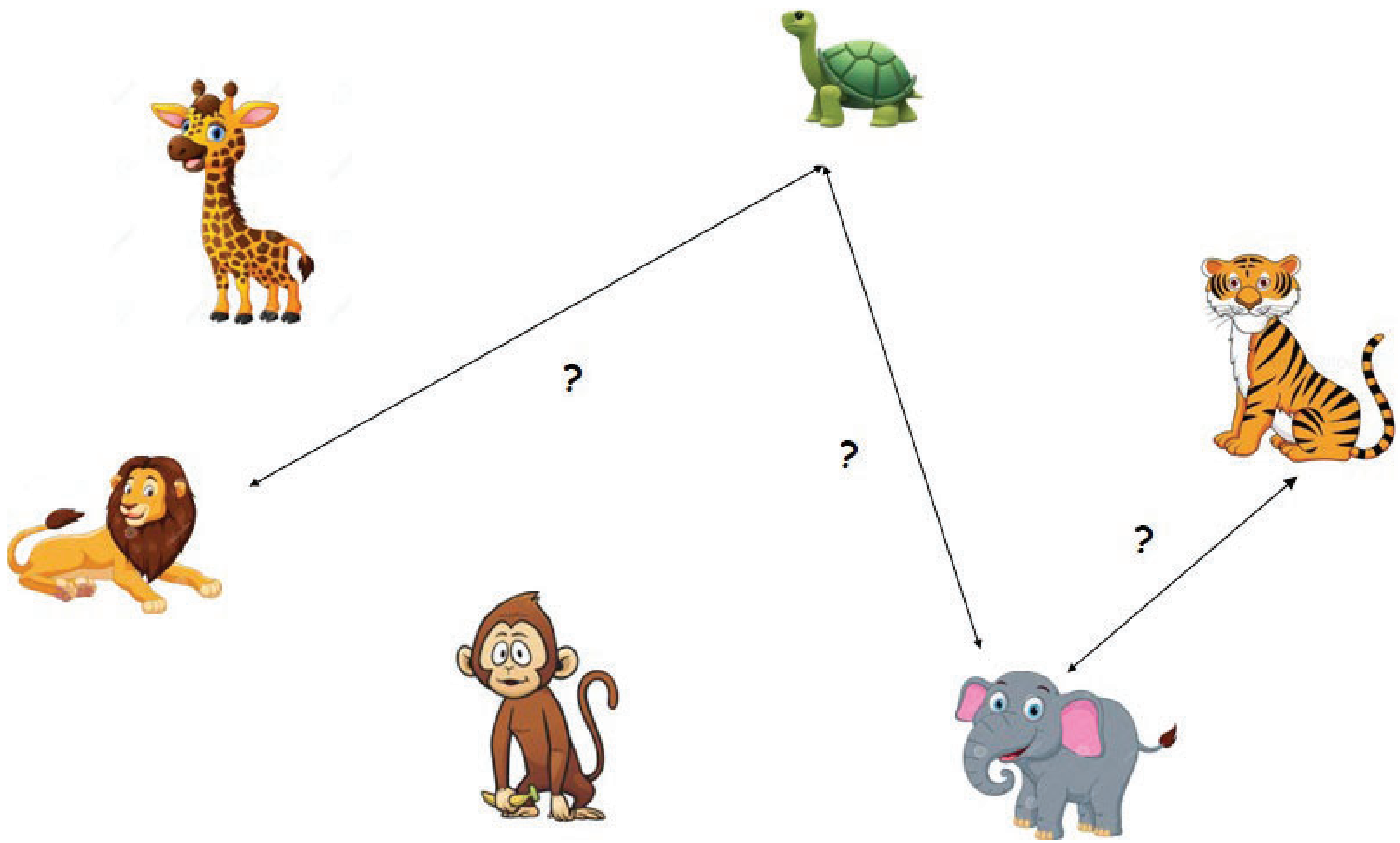
Proof: Consider graph on n vertices, one for each element. Draw an edge if a comparison query is asked b.w. those elements in round 1.



Finding the max: lower bound in 2 rounds

Proof: Consider graph on n vertices, one for each element. Draw an edge if a comparison query is asked b.w. those elements in round 1.

Case 1: If the graph has more than $n^{\frac{4}{3}}$ edges, we are done.

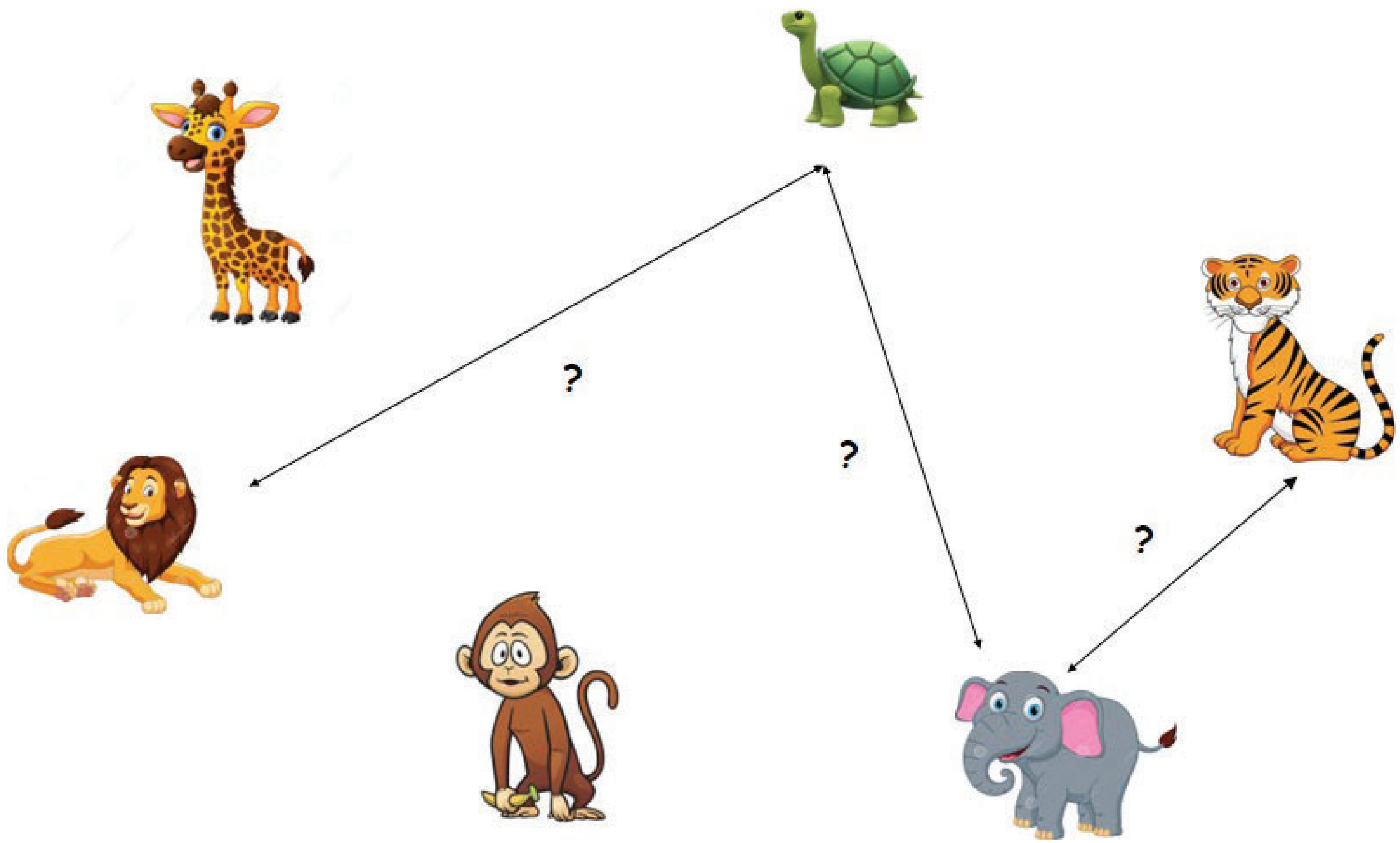


Finding the max: lower bound in 2 rounds

Proof: Consider graph on n vertices, one for each element. Draw an edge if a comparison query is asked b.w. those elements in round 1.

Case 1: If the graph has more than $n^{\frac{4}{3}}$ edges, we are done.

Case 2: Otherwise, suppose it has < $n^{\frac{4}{3}}$ edges.



Finding the max: lower bound in 2 rounds

Proof: Consider graph on n vertices, one for each element. Draw an edge if a comparison query is asked b.w. those elements in round 1.

Case 1: If the graph has more than $n^{\frac{4}{3}}$ edges, we are done.

Case 2: Otherwise, suppose it has < $n^{\frac{4}{3}}$ edges.

Lemma 1: ***A graph with n nodes and fewer than s edges has an independent set of size at least $\frac{n^2}{8s}$.***

Finding the max: lower bound in 2 rounds

Proof: Consider graph on n vertices, one for each element. Draw an edge if a comparison query is asked b.w. those elements in round 1.

Case 1: If the graph has more than $n^{\frac{4}{3}}$ edges, we are done.

Case 2: Otherwise, suppose it has < $n^{\frac{4}{3}}$ edges.

Lemma 1: ***A graph with n nodes and fewer than s edges has an independent set of size at least $\frac{n^2}{8s}$.***

- The graph has < $n^{\frac{4}{3}}$ edges, by **Lemma 1**, it has an independent set U of size k such that

$$k \geq \frac{n^2}{8 * n^{\frac{4}{3}}} = \frac{n^{\frac{2}{3}}}{8} \text{ (*)}$$

Finding the max: lower bound in 2 rounds

Proof: Consider graph on n vertices, one for each element. Draw an edge if a comparison query is asked b.w. those elements in round 1.

Case 1: If the graph has more than $n^{\frac{4}{3}}$ edges, we are done.

Case 2: Otherwise, suppose it has < $n^{\frac{4}{3}}$ edges.

Lemma 1: ***A graph with n nodes and fewer than s edges has an independent set of size at least $\frac{n^2}{8s}$.***

- The graph has < $n^{\frac{4}{3}}$ edges, by **Lemma 1**, it has an independent set U of size k such that

$$k \geq \frac{n^2}{8 * n^{\frac{4}{3}}} = \frac{n^{\frac{2}{3}}}{8} \text{ (*)}$$

- Make the items in U win all pairwise comparisons.

Finding the max: lower bound in 2 rounds

Proof: Consider graph on n vertices, one for each element. Draw an edge if a comparison query is asked b.w. those elements in round 1.

Case 1: If the graph has more than $\mathrm{n}^{\frac{4}{3}}$ edges, we are done.

Case 2: Otherwise, suppose it has < $\mathrm{n}^{\frac{4}{3}}$ edges.

Lemma 1: ***A graph with n nodes and fewer than s edges has an independent set of size at least $\frac{n^2}{8s}$.***

- The graph has < $\mathrm{n}^{\frac{4}{3}}$ edges, by **Lemma 1**, it has an independent set U of size k such that
$$k \geq \frac{n^2}{8 * n^{\frac{4}{3}}} = \frac{n^{\frac{2}{3}}}{8} \ (*)$$
- Make the items in U win all pairwise comparisons. These items must be compared in round 2 => at least $\binom{k}{2}$ comparisons in round 2.

Finding the max: lower bound in 2 rounds

Proof: Consider graph on n vertices, one for each element. Draw an edge if a comparison query is asked b.w. those elements in round 1.

Case 1: If the graph has more than $n^{\frac{4}{3}}$ edges, we are done.

Case 2: Otherwise, suppose it has < $n^{\frac{4}{3}}$ edges.

Lemma 1: ***A graph with n nodes and fewer than s edges has an independent set of size at least $\frac{n^2}{8s}$.***

- The graph has < $n^{\frac{4}{3}}$ edges, by **Lemma 1**, it has an independent set U of size k such that

$$k \geq \frac{n^2}{8 * n^{\frac{4}{3}}} = \frac{n^{\frac{2}{3}}}{8} \text{ (*)}$$

- Make the items in U win all pairwise comparisons. These items must be compared in round 2 => at least $\binom{k}{2}$ comparisons in round 2. Since $k \geq \frac{1}{8} * n^{\frac{2}{3}}$ by (*), round 2 makes $\Omega(n^{\frac{4}{3}})$ comparisons.

Finding the max: lower bound in 2 rounds

Proof of Lemma 1: ***A graph with n nodes and fewer than s edges has an independent set of size at least $\frac{n^2}{8s}$.***

Finding the max: lower bound in 2 rounds

Proof of Lemma 1: ***A graph with n nodes and fewer than s edges has an independent set of size at least $\frac{n^2}{8s}$.***

Since the number of edges is $\leq s$ $\Rightarrow$ the sum of degrees is $\leq 2 \cdot s$.

Finding the max: lower bound in 2 rounds

Proof of Lemma 1: ***A graph with n nodes and fewer than s edges has an independent set of size at least $\frac{n^2}{8s}$.***

Since the number of edges is $\leq s$ $\Rightarrow$ the sum of degrees is $\leq 2 \cdot s$.

Thus: the **average** degree is $\leq \frac{2 \cdot s}{n}$.

Finding the max: lower bound in 2 rounds

Proof of Lemma 1: ***A graph with n nodes and fewer than s edges has an independent set of size at least $\frac{n^2}{8s}$.***

Since the number of edges is $\leq s$ $\Rightarrow$ the sum of degrees is $\leq 2 \cdot s$.

Thus: the **average** degree is $\leq \frac{2 \cdot s}{n}$. ★

Then not that many nodes can exceed this value by a lot.

Finding the max: lower bound in 2 rounds

Proof of Lemma 1: ***A graph with n nodes and fewer than s edges has an independent set of size at least $\frac{n^2}{8s}$.***

Since the number of edges is $\leq s$ $\Rightarrow$ the sum of degrees is $\leq 2 \cdot s$.

Thus: the **average** degree is $\leq \frac{2 \cdot s}{n}$. ★

Then not that many nodes can exceed this value by a lot.

Let q be the number of nodes with degree $\geq 4 \cdot \frac{s}{\mathrm{n}}$.

Finding the max: lower bound in 2 rounds

Proof of Lemma 1: ***A graph with n nodes and fewer than s edges has an independent set of size at least $\frac{n^2}{8s}$.***

Since the number of edges is $\leq s \Rightarrow$ the sum of degrees is $\leq 2 \cdot s$.

Thus: the average degree is $\leq \frac{2 \cdot s}{n}$. ★

Then not that many nodes can exceed this value by a lot.

Let q be the number of nodes with degree $\geq 4 \cdot \frac{s}{\mathrm{n}}$.

Upper bounding the sum of degrees of the q nodes by the upper bound on the sum of degrees, we get

$$\mathrm{q} \cdot \left(4 \cdot \frac{s}{n}\right) \leq 2 \cdot s \Rightarrow q \leq \frac{n}{2}$$

Finding the max: lower bound in 2 rounds

Proof of Lemma 1: ***A graph with n nodes and fewer than s edges has an independent set of size at least $\frac{n^2}{8s}$.***

Since the number of edges is $\leq s$ $\Rightarrow$ the sum of degrees is $\leq 2 \cdot s$.

Thus: the average degree is $\leq \frac{2 \cdot s}{n}$. ★

Then not that many nodes can exceed this value by a lot.

Let q be the number of nodes with degree $\geq 4 \cdot \frac{s}{\mathrm{n}}$.

Upper bounding the sum of degrees of the q nodes by the upper bound on the sum of degrees, we get

$$\mathrm{q} \cdot \left(4 \cdot \frac{s}{n}\right) \leq 2 \cdot s \Rightarrow q \leq \frac{n}{2}$$

Let W = set of nodes with degree $\leq 4 \cdot \frac{s}{n}$.

Finding the max: lower bound in 2 rounds

Proof of Lemma 1: ***A graph with n nodes and fewer than s edges has an independent set of size at least $\frac{n^2}{8s}$.***

Since the number of edges is $\leq s \Rightarrow$ the sum of degrees is $\leq 2 \cdot s$.

Thus: the average degree is $\leq \frac{2 \cdot s}{n}$. ★

Then not that many nodes can exceed this value by a lot.

Let q be the number of nodes with degree $\geq 4 \cdot \frac{s}{\mathrm{n}}$.

Upper bounding the sum of degrees of the q nodes by the upper bound on the sum of degrees, we get

$$\mathrm{q} \cdot \left(4 \cdot \frac{s}{n}\right) \leq 2 \cdot s \Rightarrow q \leq \frac{n}{2}$$

Let W = set of nodes with degree $\leq 4 \cdot \frac{s}{n}$. We get $|\mathrm{W}| \geq \frac{\mathrm{n}}{2}$.

Finding the max: lower bound in 2 rounds

Proof of Lemma 1 (cont): ***A graph with n nodes and fewer than s edges has an independent set of size at least $\frac{n^2}{8s}$.***

Recap: W = set of nodes with degree $\leq 4 \cdot \frac{s}{n}$. Also $|\mathrm{W}| \geq \frac{\mathrm{n}}{2}$.

Select independent set from W greedily: select a node, eliminate all its neighbours, repeat.

What is an upper bound on how many nodes each removal eliminates?

Finding the max: lower bound in 2 rounds

Proof of Lemma 1 (cont): ***A graph with n nodes and fewer than s edges has an independent set of size at least $\frac{n^2}{8s}$.***

Recap: W = set of nodes with degree $\leq 4 \cdot \frac{s}{n}$. Also $|\mathrm{W}| \geq \frac{\mathrm{n}}{2}$.

Select independent set from W greedily: select a node, eliminate all its neighbours, repeat.

Each removal eliminates $\leq 4 \cdot \frac{s}{n}$ vertices, but W has at least $\frac{\mathrm{n}}{2}$ nodes

Finding the max: lower bound in 2 rounds

Proof of Lemma 1 (cont): ***A graph with n nodes and fewer than s edges has an independent set of size at least $\frac{n^2}{8s}$.***

Recap: W = set of nodes with degree $\leq 4 \cdot \frac{s}{n}$. Also $|W| \geq \frac{n}{2}$.

Select independent set from W greedily: select a node, eliminate all its neighbours, repeat.

Each removal eliminates $\leq 4 \cdot \frac{s}{n}$ vertices, but W has at least $\frac{n}{2}$ nodes

=> the number of iterations is at least…?

Finding the max: lower bound in 2 rounds

Proof of Lemma 1 (cont): ***A graph with n nodes and fewer than s edges has an independent set of size at least $\frac{n^2}{8s}$.***

Recap: W = set of nodes with degree $\leq 4 \cdot \frac{s}{n}$. Also $|W| \geq \frac{n}{2}$.

Select independent set from W greedily: select a node, eliminate all its neighbours, repeat.

Each removal eliminates $\leq 4 \cdot \frac{s}{n}$ vertices, but W has at least $\frac{n}{2}$ nodes

=> the number of iterations is at least:

$$\frac{|W|}{\text{max \# vertices removed per round}} = \frac{n/2}{4 \cdot \frac{s}{n}} = \frac{n^2}{8s}.$$

Finding the max: lower bound in 2 rounds

Proof of Lemma 1 (cont): ***A graph with n nodes and fewer than s edges has an independent set of size at least $\frac{n^2}{8s}$.***

Recap: W = set of nodes with degree $\leq 4 \cdot \frac{s}{n}$. Also $|\mathrm{W}| \geq \frac{\mathrm{n}}{2}$.

Select independent set from W greedily: select a node, eliminate all its neighbours, repeat.

Each removal eliminates $\leq 4 \cdot \frac{s}{n}$ vertices, but W has at least $\frac{\mathrm{n}}{2}$ nodes => the number of iterations is at least:

$$\frac{|\mathrm{W}|}{\text{max \# vertices removed per round}} = \frac{n/2}{4 \cdot \frac{s}{n}} = \frac{n^2}{8s}.$$

Thus the graph has an independent set of size at least $\frac{n^2}{8s}$. This completes the proof of the lemma.

Finding the maximum in 3 rounds

Exercise.

Finding the maximum in 3 rounds

Upper bound:

- Divide the elements into groups of k [TBD] and find the maximum element in each group

Finding the maximum in 3 rounds

Upper bound:

- Divide the elements into groups of k [TBD] and find the maximum element in each group -> in round 1 can obtain the maximum $\mathrm{y_i}$ for each group i. We get $k * \frac{n^2}{k^2} = \frac{n^2}{k}$ comparisons in round 1.

Finding the maximum in 3 rounds

Upper bound:

- Divide the elements into groups of k [TBD] and find the maximum element in each group -> in round 1 can obtain the maximum $\mathrm{y_i}$ for each group i. We get $k * \frac{n^2}{k^2} = \frac{n^2}{k}$ comparisons in round 1.
- By 2 round protocol, the maximum of the $\mathrm{y_i}$'s can be obtained using $2 \cdot k^{\frac{4}{3}}$ comparisons in the remaining two rounds.

Finding the maximum in 3 rounds

Upper bound:

- Divide the elements into groups of k [TBD] and find the maximum element in each group -> in round 1 can obtain the maximum y_i for each group i. We get $k * \frac{n^2}{k^2} = \frac{n^2}{k}$ comparisons in round 1.
- By 2 round protocol, the maximum of the y_i's can be obtained using $2 \cdot k^{\frac{4}{3}}$ comparisons in the remaining two rounds.
- The total number of comparisons is $\leq \frac{n^2}{k} + 2 \cdot k^{\frac{4}{3}}$.

Finding the maximum in 3 rounds

Upper bound:

- Divide the elements into groups of k [TBD] and find the maximum element in each group -> in round 1 can obtain the maximum y_i for each group i. We get $k * \frac{n^2}{k^2} = \frac{n^2}{k}$ comparisons in round 1.
- By 2 round protocol, the maximum of the y_i's can be obtained using $2 \cdot k^{\frac{4}{3}}$ comparisons in the remaining two rounds.
- The total number of comparisons is $\leq \frac{n^2}{k} + 2 \cdot k^{\frac{4}{3}}$.
- By setting $k = n^{\frac{6}{7}}$, we get at most $\frac{n^2}{k} + k^{\frac{4}{3}} = 3 \cdot n^{\frac{8}{7}} \in O(n^{\frac{8}{7}})$ comparisons.

Finding the maximum in 3 rounds

Lower bound:

Finding the maximum in 3 rounds

Lower bound:

- Use same technique as for r=2 rounds: assume the first round uses at most k comparisons (if it uses less we are done).

Finding the maximum in 3 rounds

Lower bound:

- Use same technique as for r=2 rounds: assume the first round uses at most k comparisons (if it uses less we are done).
- Find large independent set S (By Lemma 1, can find of size $\geq \frac{n^2}{8 \cdot k}$)

Recall Lemma 1: ***A graph with n nodes and fewer than s edges has an independent set of size at least $\frac{n^2}{8s}$.***

Finding the maximum in 3 rounds

Lower bound:

- Use same technique as for r=2 rounds: assume the first round uses at most k comparisons (if it uses less we are done).
- Find large independent set S (By Lemma 1, can find of size $\geq \frac{n^2}{8 \cdot k}$)
- By applying the lower bound for r=2 rounds, it follows we need at least $\left(\frac{n^2}{8k}\right)^{4/3}$ comparisons to find the max in S in 2 rounds.

Finding the maximum in 3 rounds

Lower bound:

- Use same technique as for r=2 rounds: assume the first round uses at most k comparisons (if it uses less we are done).
- Find large independent set S (By Lemma 1, can find of size $\geq \frac{n^2}{8 \cdot k}$)
- By applying the lower bound for r=2 rounds, it follows we need at least $\left(\frac{n^2}{8k}\right)^{4/3}$ comparisons to find the max in S in 2 rounds.
- Thus any 3 round protocol uses either $\geq \mathrm{k}$ comparisons in the first round or $\geq \left(\frac{n^2}{8k}\right)^{4/3}$ in the next 2 rounds.

Finding the maximum in 3 rounds

Lower bound:

- Use same technique as for r=2 rounds: assume the first round uses at most k comparisons (if it uses less we are done).
- Find large independent set S (By Lemma 1, can find of size $\geq \frac{n^2}{8 \cdot k}$)
- By applying the lower bound for r=2 rounds, it follows we need at least $\left(\frac{n^2}{8k}\right)^{4/3}$ comparisons to find the max in S in 2 rounds.
- Thus any 3 round protocol uses either $\geq \mathrm{k}$ comparisons in the first round or $\geq \left(\frac{n^2}{8k}\right)^{4/3}$ in the next 2 rounds.
- By setting $\mathrm{k} = n^{8/7}$, we get a lower bound of $\Omega(n^{8/7})$.