Comparison Sorts

Best [Average Cases]

The best sorts (in terms of average cases) have been $\Theta(N\log(N))$

Best [Average Cases]

The best sorts (in terms of average cases) have been $\Theta(N\log(N))$

Can a better sort exist? Can sorts have potentially linear Average Cases?

Best [Average Cases]

The best sorts (in terms of average cases) have been $\Theta(N\log(N))$

Can a better sort exist? Can sorts have potentially linear Average Cases?

Sort of, but these sorts are approaching the problem all wrong.

Best [Average Cases]

The best sorts (in terms of average cases) have been $\Theta(N\log(N))$

Can a better sort exist? Can sorts have potentially linear Average Cases?

Sort of, but these sorts are approaching the problem all wrong.

Comparison sorts are theoretically limited.

Input Space

Assume the input array is size N and all values are distinct.

Input Space

Assume the input array is size N and all values are distinct.

How many different possible inputs exist?

Input Space

Assume the input array is size N and all values are distinct.

How many different possible inputs exist?

A sorting method in a sense needs to determine which input is given in order to produced the sorted array.

Input Space

Assume the input array is size N and all values are distinct.

How many different possible inputs exist?

A sorting method in a sense needs to determine which input is given in order to produced the sorted array.

Reasoning if the result of the comparisons resulted in two potential arrangements, then a deterministic sort would be wrong on one of those two cases.

Input Space

Assume the input array is size N and all values are distinct.

How many different possible inputs exist?

A sorting method in a sense needs to determine which input is given in order to produced the sorted array.

Reasoning if the result of the comparisons resulted in two potential arrangements, then a deterministic sort would be wrong on one of those two cases.

Every comparison will split the input space into two pieces.

Input Space

Assume the input array is size N and all values are distinct.

How many different possible inputs exist?

A sorting method in a sense needs to determine which input is given in order to produced the sorted array.

Reasoning if the result of the comparisons resulted in two potential arrangements, then a deterministic sort would be wrong on one of those two cases.

Every comparison will split the input space into two pieces.

Best case is the split is even.

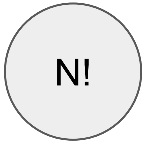
Minimum [Maximum Comparisons Needed]

We need at least $\log_2(N!)$ comparisons to make the input space 1 arrangement.

Minimum [Maximum Comparisons Needed]

We need at least $\log_2(N!)$ comparisons to make the input space 1 arrangement.

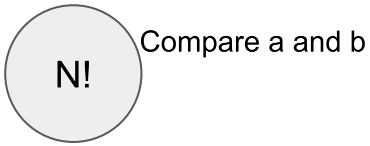
Pictorially



Minimum [Maximum Comparisons Needed]

We need at least $\log_2(N!)$ comparisons to make the input space 1 arrangement.

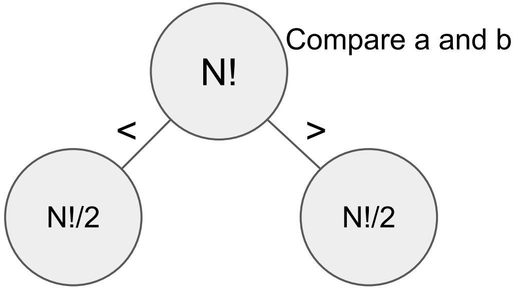
Pictorially



Minimum [Maximum Comparisons Needed]

We need at least $\log_2(N!)$ comparisons to make the input space 1 arrangement.

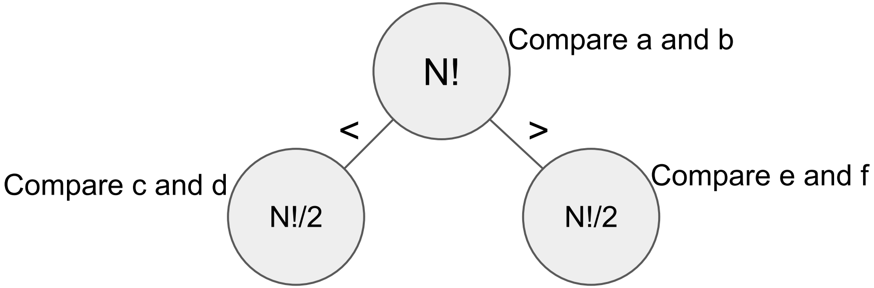
Pictorially



Minimum [Maximum Comparisons Needed]

We need at least $\log_2(N!)$ comparisons to make the input space 1 arrangement.

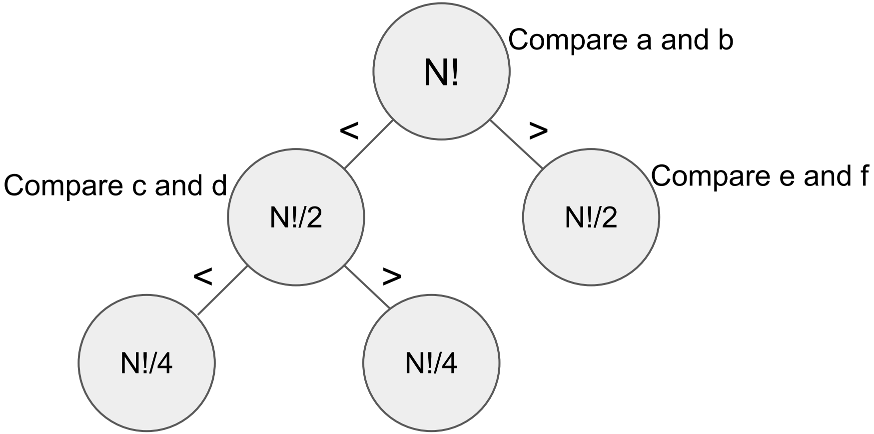
Pictorially



Minimum [Maximum Comparisons Needed]

We need at least $\log_2(N!)$ comparisons to make the input space 1 arrangement.

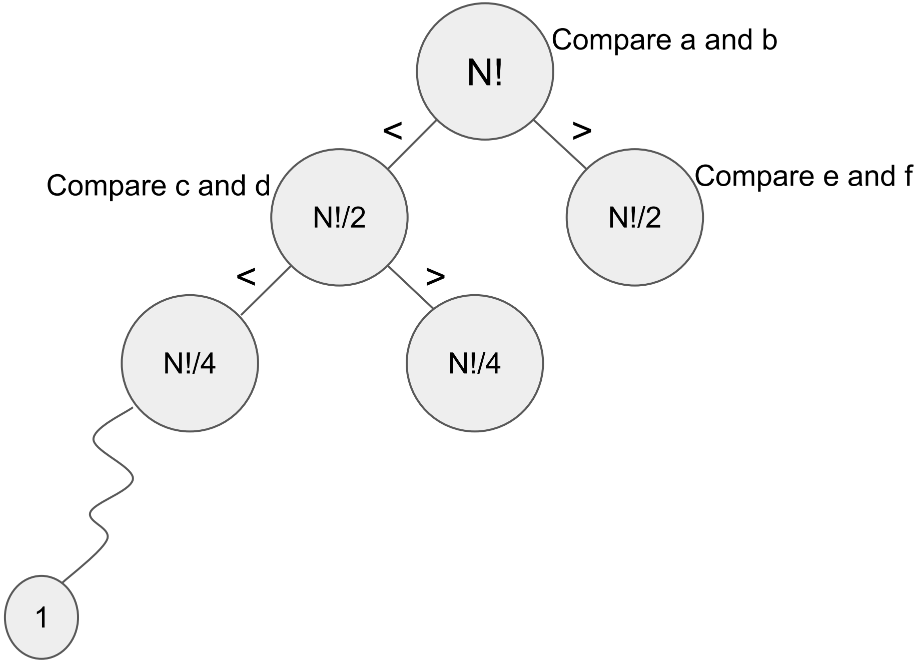
Pictorially



Minimum [Maximum Comparisons Needed]

We need at least $\log_2(N!)$ comparisons to make the input space 1 arrangement.

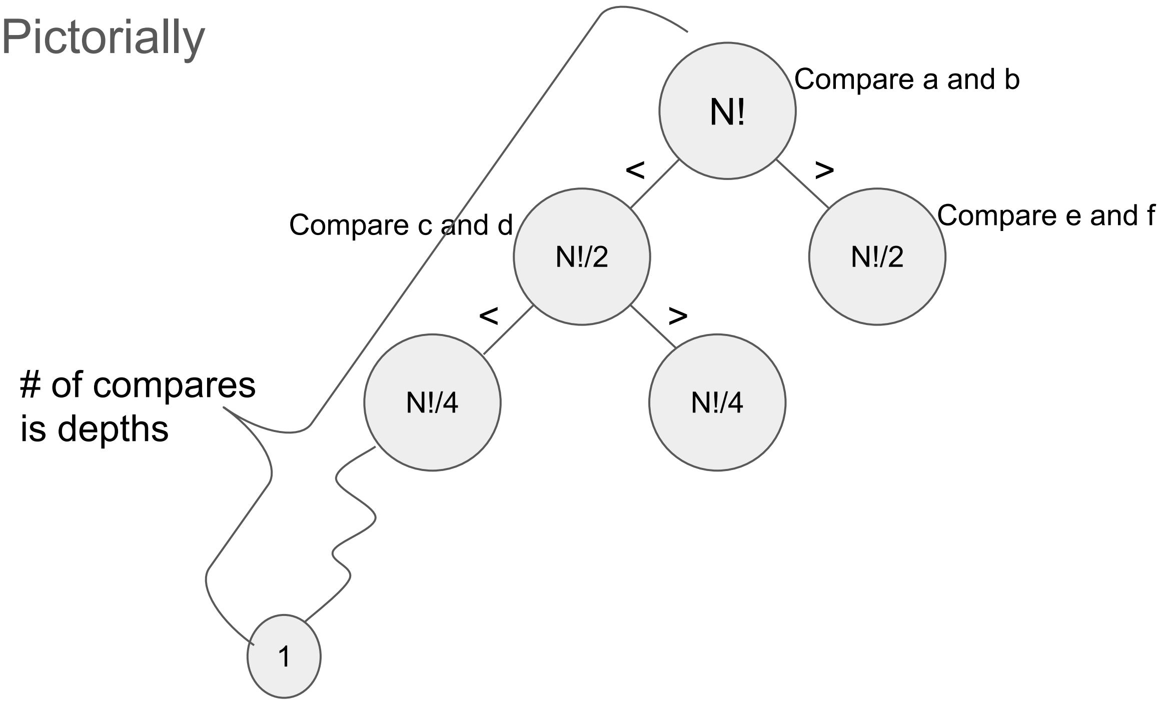
Pictorially



Minimum [Maximum Comparisons Needed]

We need at least $\log_2(N!)$ comparisons to make the input space 1 arrangement.

Pictorially



Back to Math

$\log_2(N!) = \log_2(N(N-1)(N-2)...(2)(1))$

Back to Math

$\log_2(N!) = \log_2(N(N-1)(N-2)...(2)(1))$

$\log_2(N!) = \log_2(N) + \log_2(N-1) + \log_2(N-2) + \ldots + \log_2(2) + \log_2(1)$

Back to Math

$\log_2(N!) = \log_2(N(N-1)(N-2)...(2)(1))$

$\log_2(N!) = \log_2(N) + \log_2(N-1) + \log_2(N-2) + \ldots + \log_2(2) + \log_2(1)$

Clearly less than $N \log_2(N)$

Back to Math

$\log_2(N!) = \log_2(N(N-1)(N-2)...(2)(1))$

$\log_2(N!) = \log_2(N) + \log_2(N-1) + \log_2(N-2) + \ldots + \log_2(2) + \log_2(1)$

Clearly less than N $\log_2(N)$

Half of the terms are greater than $\log_2(N/2) = \log_2(N)-1$

Back to Math

$\log_2(N!) = \log_2(N(N-1)(N-2)...(2)(1))$

$\log_2(N!) = \log_2(N) + \log_2(N-1) + \log_2(N-2) + \ldots + \log_2(2) + \log_2(1)$

Clearly less than $N \log_2(N)$

Half of the terms are greater than $\log_2(N/2) = \log_2(N)-1$

So a lower bound is $N/2(\log_2(N)-1) = N/2\log_2(N) - N/2 \in \Omega(N\log(N))$

Back to Math

$\log_2(N!) = \log_2(N(N-1)(N-2)...(2)(1))$

$\log_2(N!) = \log_2(N) + \log_2(N-1) + \log_2(N-2) + \ldots + \log_2(2) + \log_2(1)$

Clearly less than $N \log_2(N)$

Half of the terms are greater than $\log_2(N/2) = \log_2(N)-1$

So a lower bound is $N/2(\log_2(N)-1) = N/2\log_2(N) - N/2 \in \Omega(N\log(N))$

Thus, it's in $\Theta(N\log(N))$