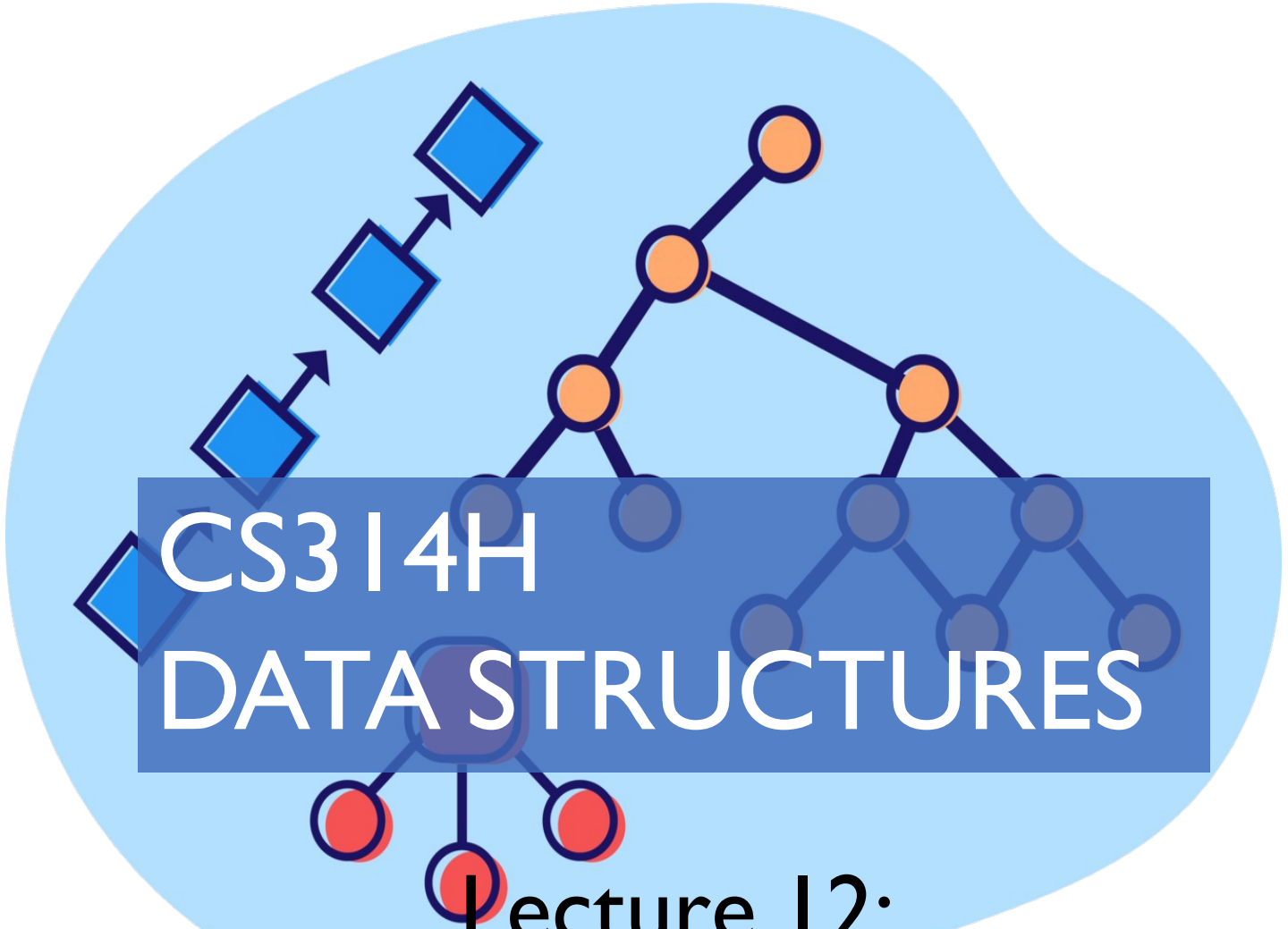


CS314H
DATA STRUCTURES

Lecture 12:
Algorithm Analysis 2 – Sorting Lower Bound

Mikyung Han

Please, interrupt and ask questions AT ANY TIME !

Reminders

- Read Ch 16 Ch 7 (except 7.4 and 7.6) if you haven't done so

Outline

1. Administrative
2. Hangman

What is Hangman?

- Start with set of words in a dictionary
  - `{ally beta cool deal else flew good hope ibex}`
- Computer starts by choosing a word to guess. For example,
  - `{else}`
- Each round user guesses an alphabet in the word and computer outputs the result
  - Round one: your guess? e
  - Round one result: Yes, there are two e's [e - - e]
  - Round two: your guess? a
  - Round two result: No, there is no a [e - - e]
  - Round three: your guess? s
  - Round three result:Yes, there is one s [e - s e]
  - …

Hangman has a brother

… who is evil

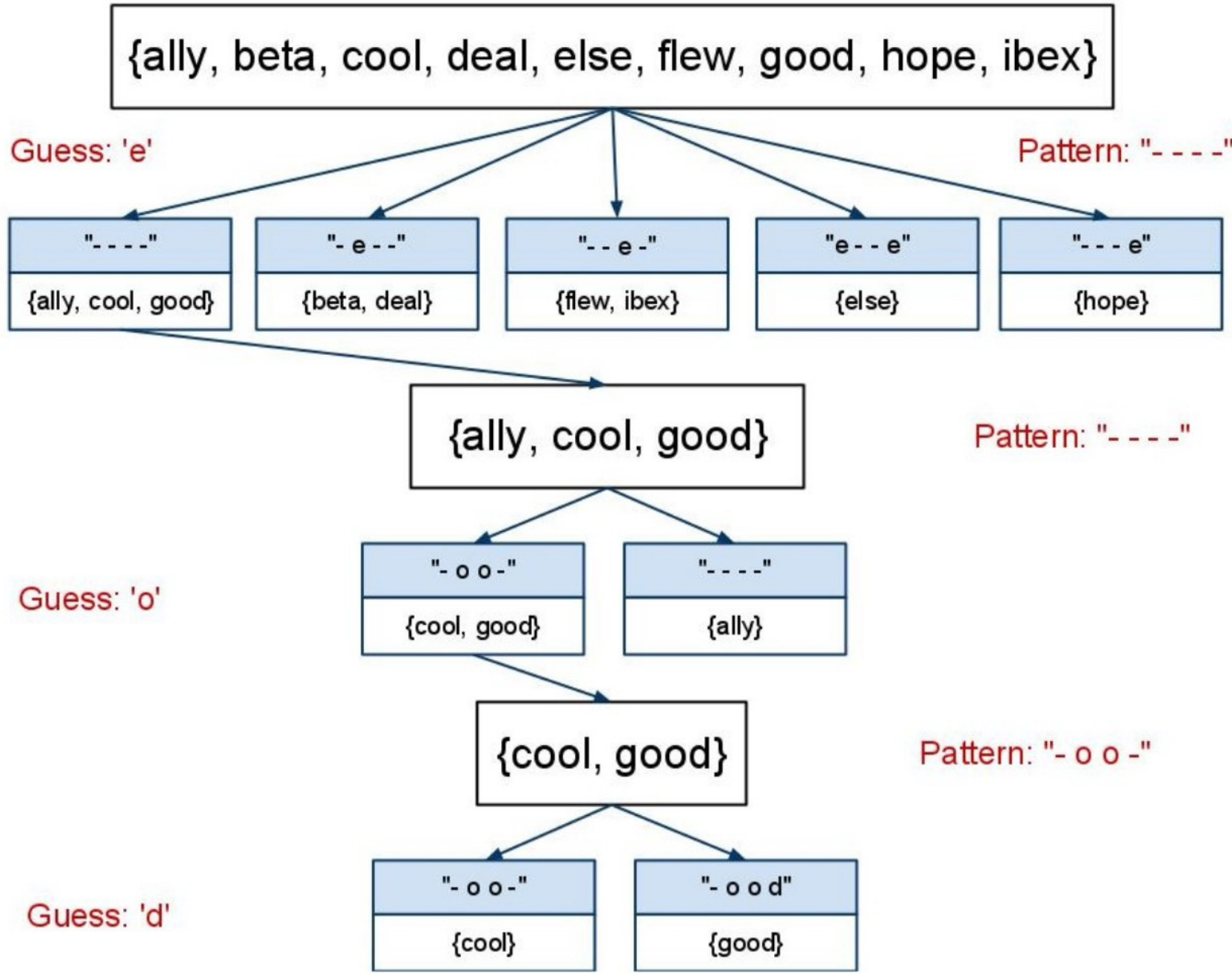
What is Evil Hangman? See UW assignment spec

- Start with set of words in a dictionary
  - `{ally beta cool deal else flew good hope ibex}`
- Computer never commits to a word but tries to delay user guessing
  - `{ally beta cool deal else flew good hope ibex}`
- Each round user guesses an alphabet in the word and computer outputs the result
  - Round one: your guess? e
- Which answer would reveal less about the word?
  - if no, `{ally cool good}`
  - if yes, `{else} or {beta deal} or {flew ibex} or {hope}`
- Algorithm pick the "largest" set
  - Round one result: No, there is no e [- - - -]

What is Evil Hangman?

- **Current possible words**
  - `{ally cool good}`
  - Round two: your guess? o
- **Which answer would reveal less about the word?**
  - if yes, `{cool good}`
  - if no, `{ally}`
- **Algorithm pick the "largest" set**
  - Round two result: Yes, there are two o's [- o o -]

With a larger dictionary, it takes much longer to commit


{ally, beta, cool, deal, else, flew, good, hope, ibex}
Guess: 'e'
Pattern: "- - - -"
"- - - -"
{ally, cool, good}
"- e - -"
{beta, deal}
"- - e -"
{flew, ibex}
"e - - e"
{else}
"- - - e"
{hope}
{ally, cool, good}
Pattern: "- - - -"
Guess: 'o'
"- o o -"
{cool, good}
"- - - -"
{ally}
{cool, good}
Pattern: "- o o -"
Guess: 'd'
"- o o -"
{cool}
"- o o d"
{good}

Outline

1. Administrative
2. Evil Hangman
3. Lower bound of sorting

Theorem: Any deterministic comparison-based sorting algorithm in the worst case must perform $\Omega(n \log n)$ comparisons to sort n elements

- What does deterministic mean?
  - Given the same input algorithm always outputs the same result
- What is Ω?
  - Lower bound: need at least n log n comparisons

Why just count the number of comparison?

Sorting algorithm solves the problem by doing a sequence of comparing two elements "Is $a_i < a_j$?"

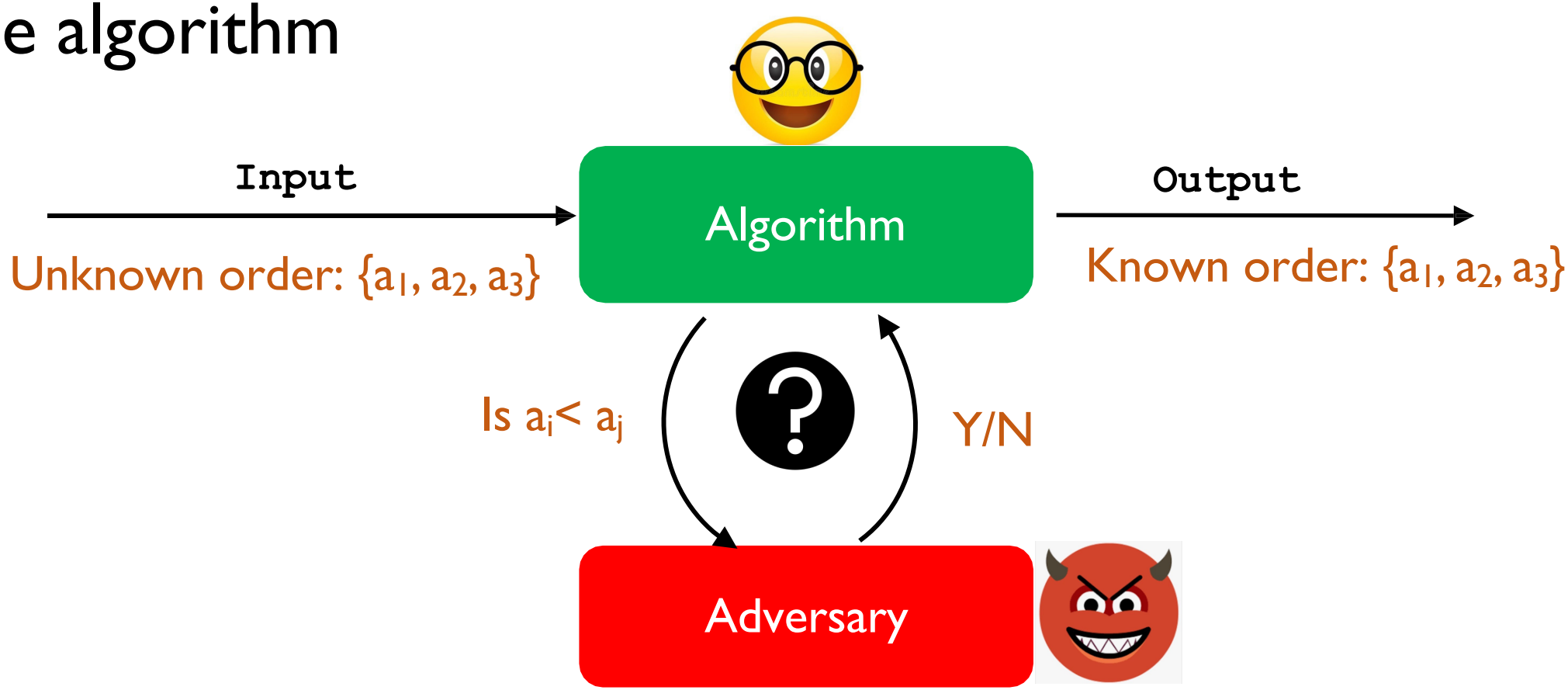
- What is essential operation in comparison-based sorting algorithm?
- Can algorithm perform other actions than comparisons?
- Since we are interested in the lower bound only
  it's ok to abstract sorting algorithm to such sequence of comparisons

Counting num of comparisons provides a valid lower bound

Theorem: Any deterministic comparison-based sorting algorithm in the worst case must perform $\Omega(n \log n)$ comparisons to sort n elements

To prove above, let's play a game

The algorithm



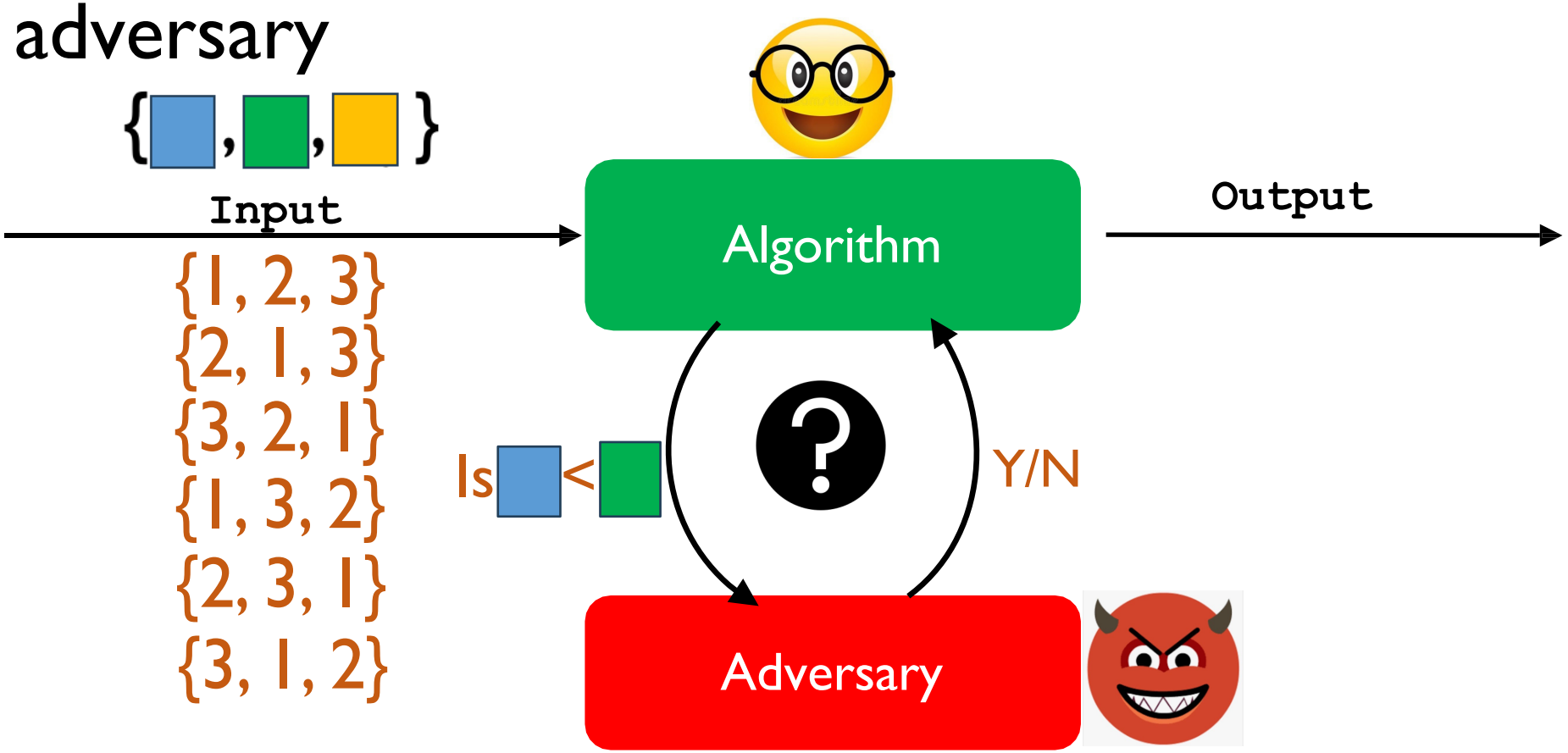
Algorithm
- initially doesn't know anything about (a_1, a_2, a_3)
- picks two elements to question as smart as possible to reduce the number of rounds
- Only after asking the adversary, it knows which one is smaller

For example, we have 3 elements { a_1 , a_2 , a_3 }

- Algorithm doesn't know which ones are bigger or smaller

- First, algorithm asks $a_1 < a_2$?
- Say, the answer is "Yes"
- Next, algorithm asks $a_2 < a_3$?
- What should you answer if you are the adversary?
  - If yes, then algorithm knows about all with just 2 comparisons
  - If no, then algorithm still has to ask one more time $a_1 < a_3$?

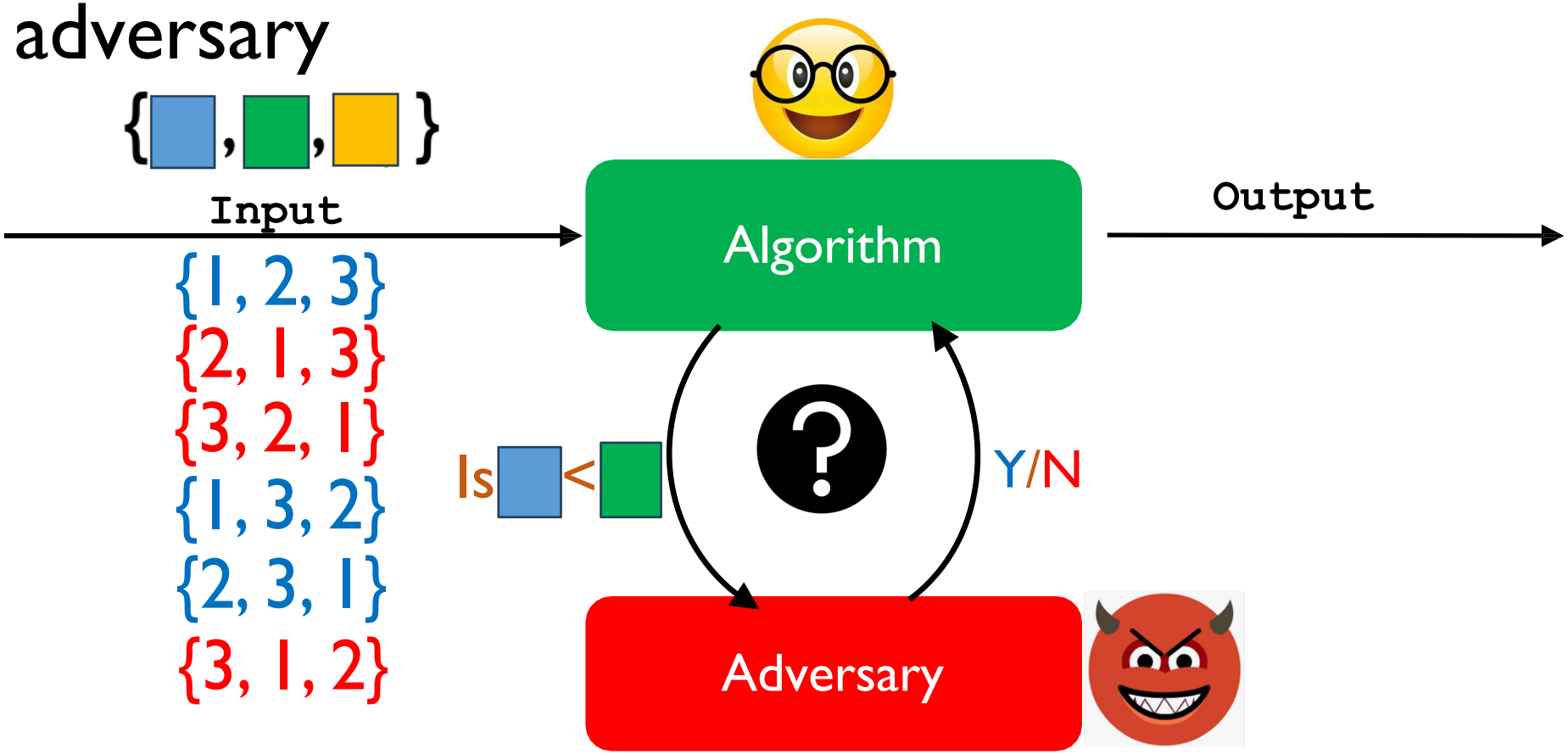
The adversary



Adversary

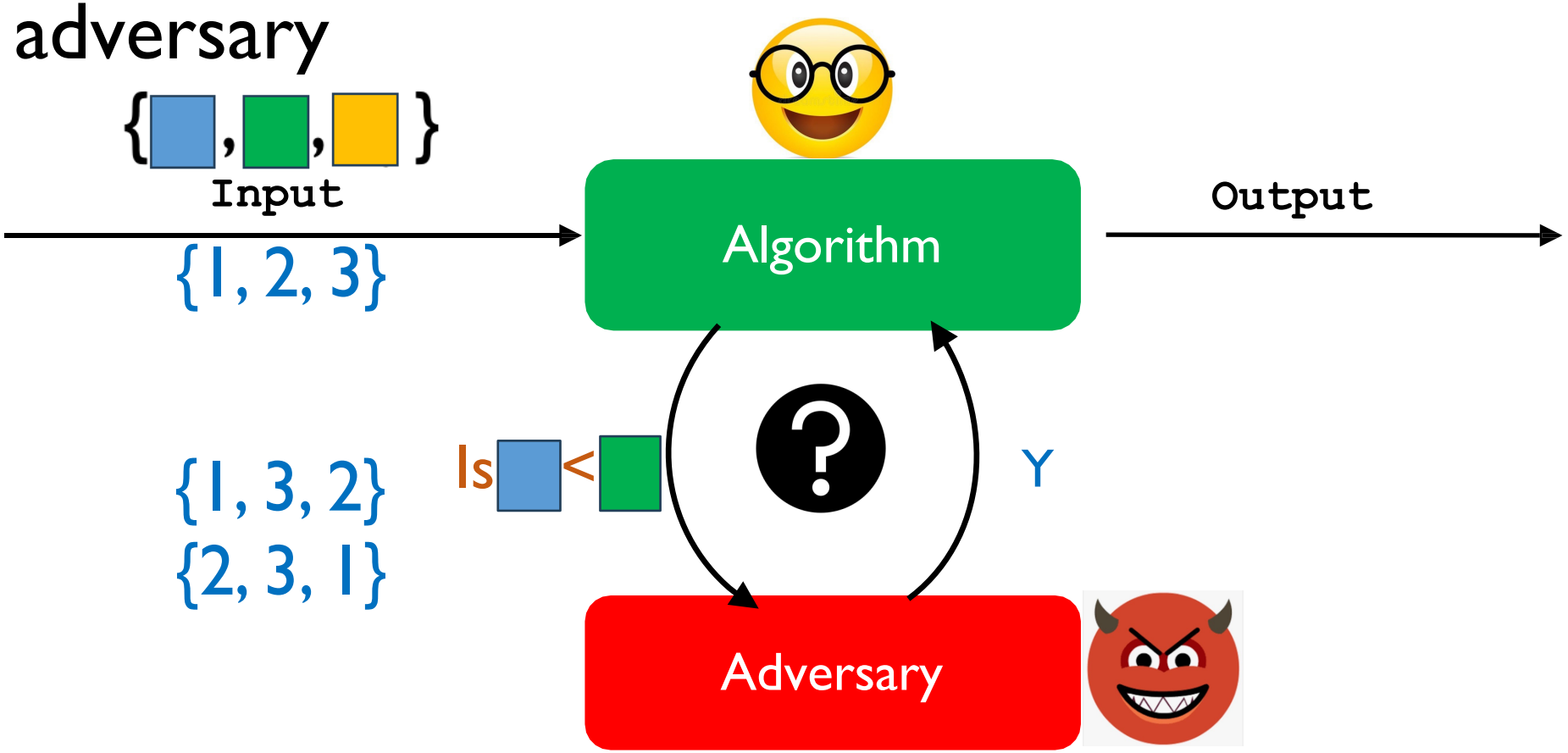
- Starts with all possible combination of elements as its pool
- Based the question, it answers in the most "evil" way, giving the least info as possible (i.e., the answer with the largest pool)

The adversary



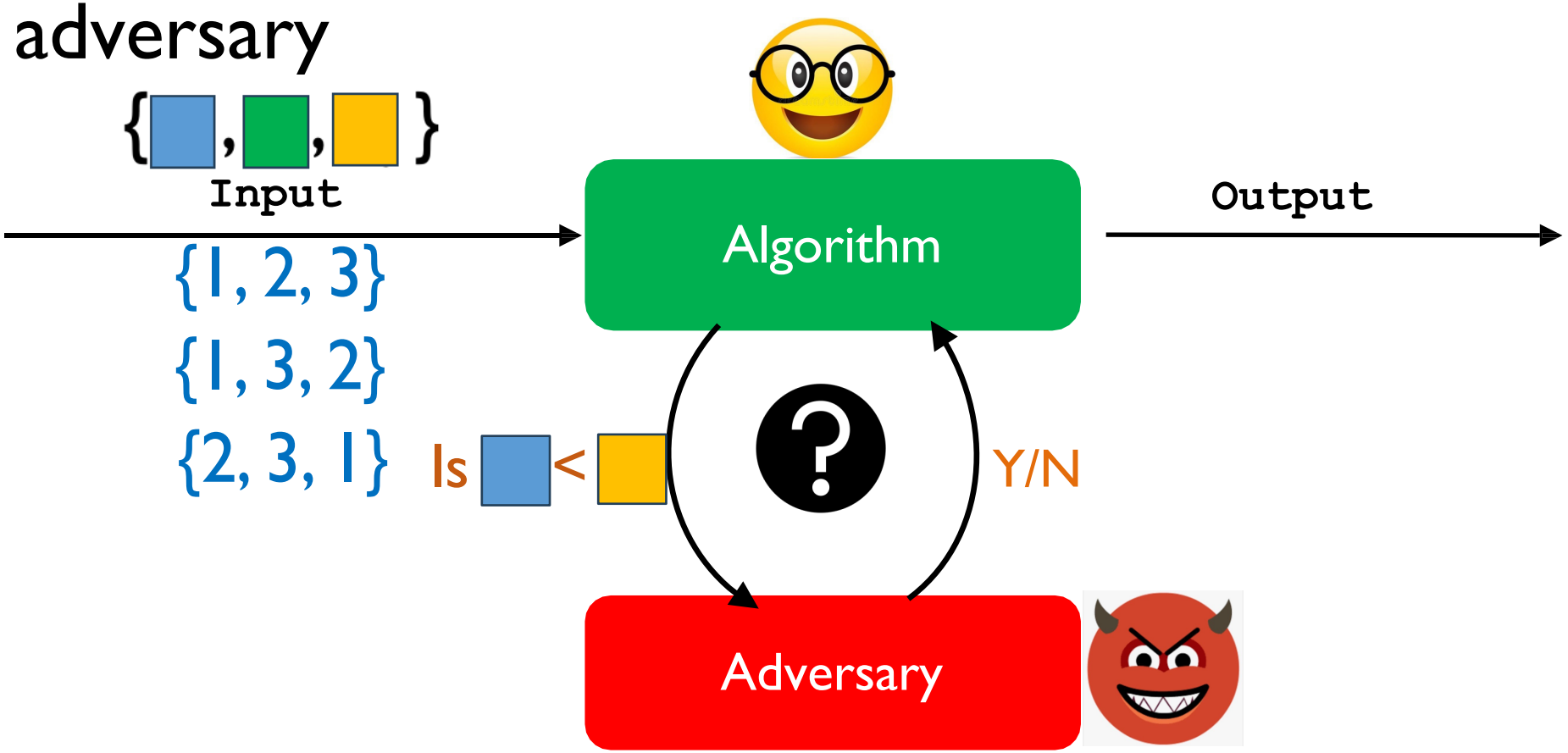
Adversary answers either Y or N as neither reveals less

The adversary



Say, Adversary choose to answer Y

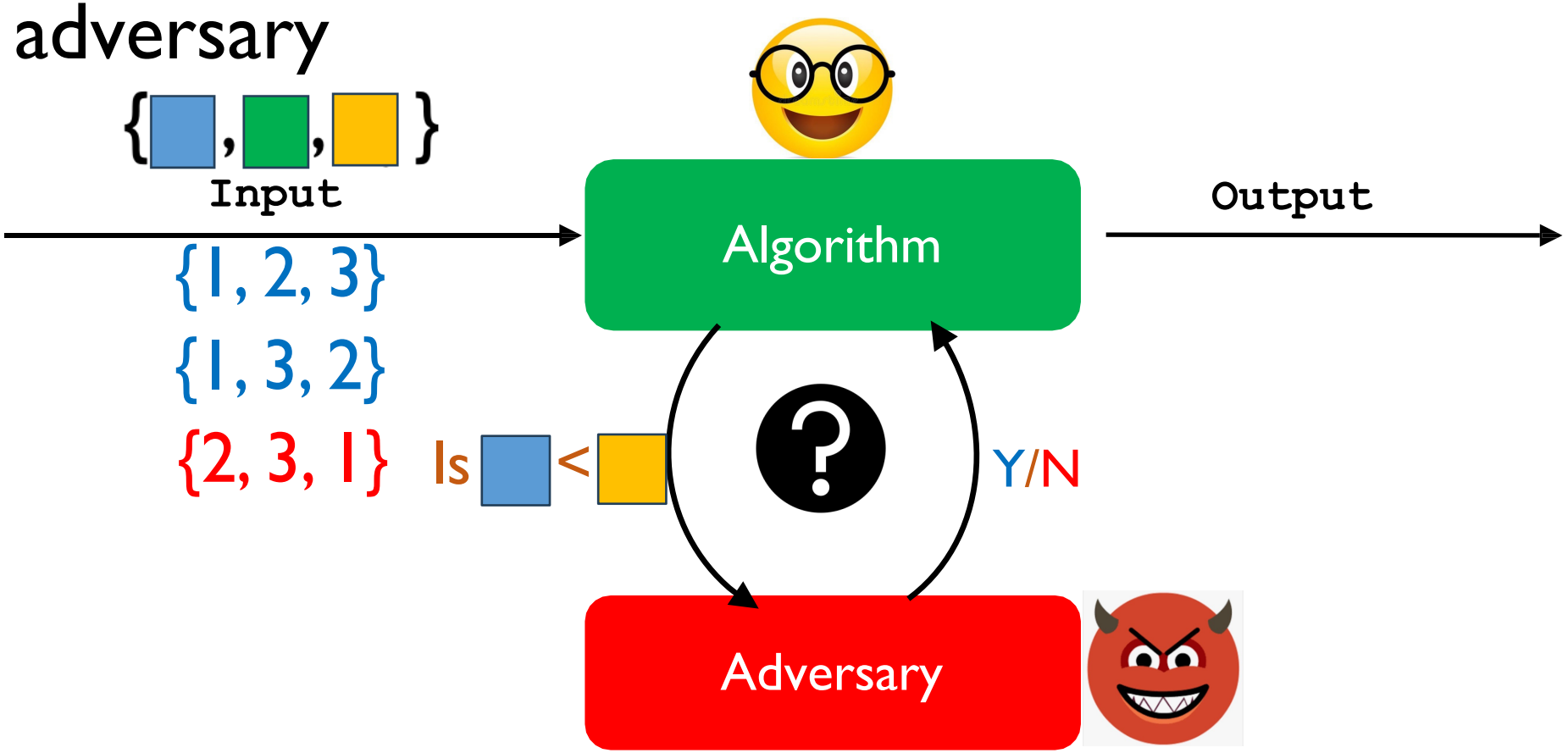
The adversary



Algorithm asks "Is $a_1 < a_3$?"

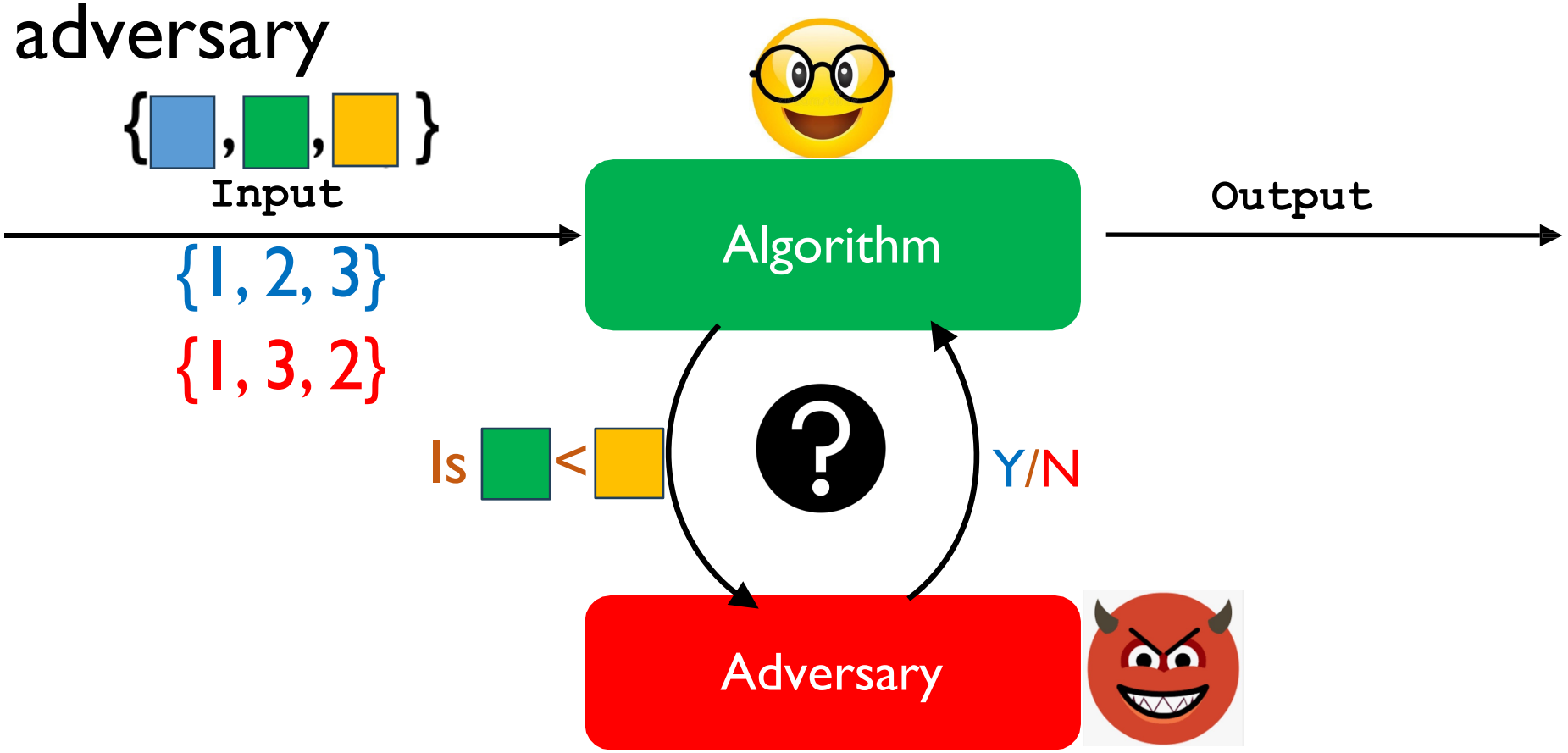
What should the adversary answer?

The adversary



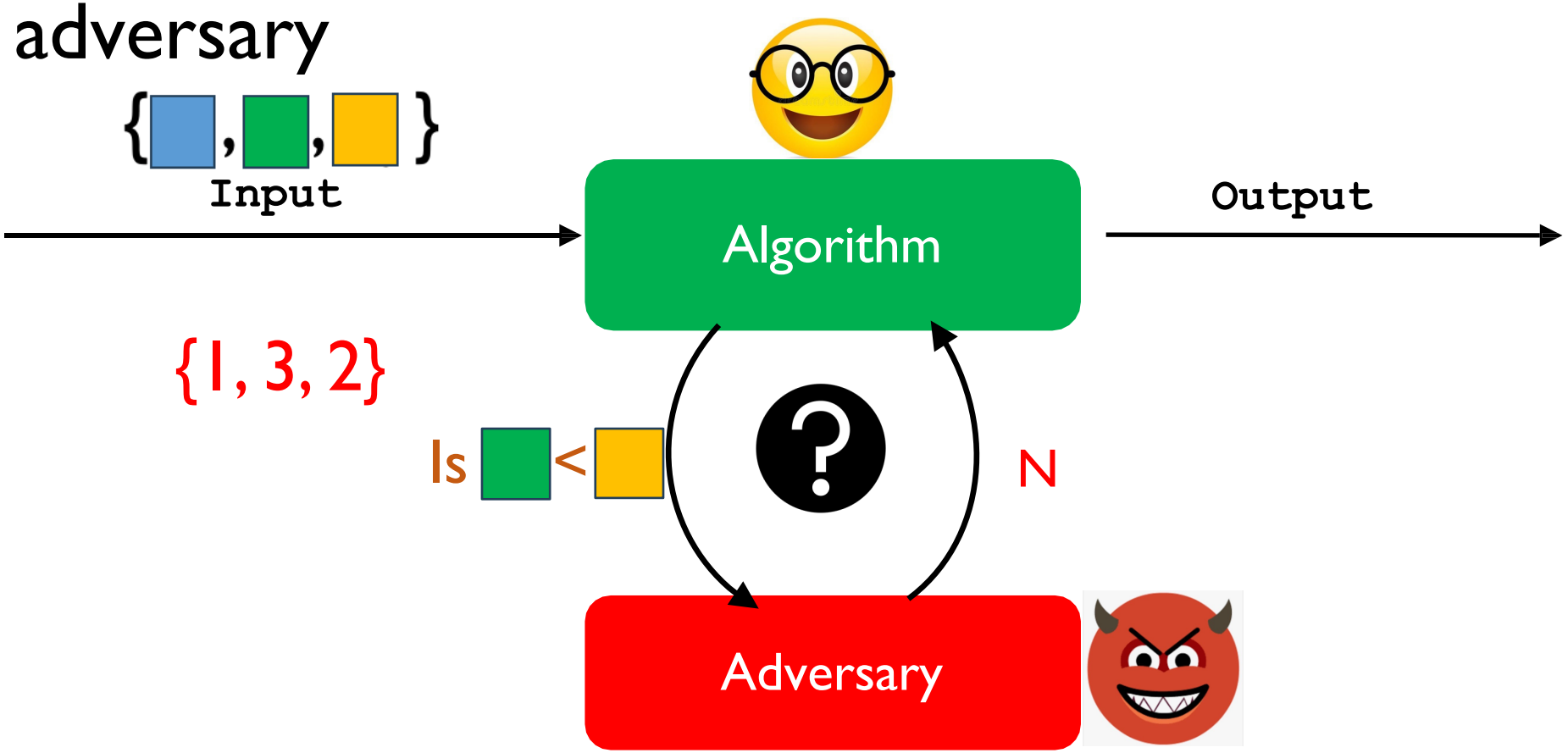
Adversary answers Y as it leads to one additional round

The adversary



Algorithm asks the final question. Adversary can say either Y or N

The adversary



Say, Adversary choose to answer N. Algorithm finds the ordering!

Given the sequence of questions, adversary finds a worst possible input

- Namely "Is $a_1 < a_2$?","Is $a_1 < a_3$?", "Is $a_2 < a_3$?"
- {1,3, 2} turns out to be the worst case input
- Could there be other worst case inputs?

If algorithm asked with a different sequence, worst case input would have changed

Why algorithm has to be as smart as possible?

Theorem: Any deterministic comparison-based sorting algorithm in the worst case must perform $\Omega(n \log n)$ comparisons to sort n elements

The smartest will provide the lower bound for ANY algorithm

Why adversary has to be as evil as possible?

Theorem: Any deterministic comparison-based sorting algorithm in the worst case must perform $\Omega(n \log n)$ comparisons to sort n elements

Being as evil as possible == hand-crafting the very WORST case input

Now let's prove

Theorem: Any deterministic comparison-based sorting algorithm in the worst case must perform $\Omega(n \log n)$ comparisons to sort n elements

Where does n and log n come from?

Outline

1. Administrative
2. Evil Hangman
3. Lower bound of sorting as a game
4. Proof of n log n

What is the initial pool size with n elements?

With n elements, there are n! possible permutations

Each round the pool size shrinks as the adversary answers Y/N to the comparison question

If S is the current pool and |S| is the pool size, how many will remain in the pool after a new comparison in the worst case (for algorithm)?

Half of |S|. Why?

Suppose NOT.
So, after the comparison, less than half remained

When would this be possible?

When the adversary is NOT evil enough

If adversary was most evil it would have picked the other choice leaving more options for itself. Contradiction!

In terms of "set": consider two items and pool S

- Half the sets in S will have smaller item placed before larger item
- The other half will have larger item placed before smaller item

Each round, half will be eliminated based on adversary's answer

Given X, how many times do you have to divide by half so that it becomes 1?

$\log_2 X$

In this case X = n!
$\log_2$ n! gives the total number of rounds

Use "log (p x q) = log p + log q"

$\log n! = \log n + \log (n-1) + \quad \ldots \quad + \log(1)$

$= \log n + \log (n-1) + \ldots + \log(\ ) + \ \ldots \quad + \ \log(1)$

$\geq \log n + \log (n-1) + \ldots + \log(\ )$ //drop the smaller half

$\geq \log (\ ) + \log(\ ) + \ldots + \log(\ )$ //replace all terms with smallest one

$\geq (\ ) \ \log (\ ) =$

# of comparisons is $\Omega(n \log n)$

Outline

1. Administrative
2. Evil Hangman
3. Lower bound of sorting as a game
4. Proof of n log n
5. Conclusion

Smartest algorithm under the harshest condition has $\Omega(n \log n)$ bound

Harshest condition gives the worst-case input:
Some other input may require lesser num of comparisons

Smartest algorithm gives the lower bound for all:
non-smart algorithm could do more num of comparisons