How can we know when our algorithm is optimal (asymptotically)?

Is there a sorting algorithm with asymptotic time complexity (worst-case or expected case) better than $O(n \log n)$?

We can't prove any limitation (lower bound) without basing it on some model of computation.

Model of Computation

Comparison-based model: you can only learn about relative order of elements through a comparison.

Observation:
For a comparison-based alg.
time complexity $\geq$ # of comparisons

Can I prove a statement of form: Any comparison-based alg to sort n elements makes $\geq F(n)$ comparisons?

Adversary Lower Bound

View as a game with 2 players

Define adversary strategy

Must describe how to reply to **whatever question** the alg asks
In a way that all answers are consistent with some "input" ← # adv picked

Goal of adv: max # rounds (one question answer/round)

Alg: tries to minimize the # of rounds

What's a good adv. strategy for 20 questions.

Adv can make a List L with all possible #s

{~~1~~, ~~2~~, ~~3~~, 4, 5, 6, 7, 8, ~~9~~, ~~10~~}

Alg Is # < 9? yes means 1-8 "alive"
no means 9-10 "alive"

Is # < 4? ~~yes~~ 1, 2, 3 alive
no 4, 5, 6, 7, 8 alive

Correct Alg cannot be done until $|L| = 1$.

Why not? Whatever the alg says the answer is, the adv. can report that he had a different answer all along.

This answer is an input that causes the # comparisons to be # of rounds in the game

Goal:

For any comparison-based alg to solve problem P there exist an input of size n for which computation time $\geq F(n)$

Worst-case

Analyze adv. strategy we gave for "20 questions" where # adv picks 1, …, n

Initially $|L| = n$

Question: How many rounds <u>must</u> occur before $|L|$ could be 1. (maybe more rounds are needed)

initially $|L| = n$

If L_i is # of elements in $|L|$ after round i $(L_0 = n)$

$$L_{i+1} \geq \lceil L_i/2 \rceil$$

rounds until $|L| = 1$ $\geq \lceil \log_2 n \rceil$

Ex

$$n = 16 \xrightarrow{Q1} 8 \xrightarrow{Q2} 4 \xrightarrow{Q3} 2 \xrightarrow{Q4} 1$$

$\log_2 16 = 4$

$n = 17$

$$17 \xrightarrow{Q1} 9 \xrightarrow{Q2} 5 \xrightarrow{Q3} 3 \xrightarrow{Q4} 2 \xrightarrow{Q5} 1$$