CMSC 131

Fall 2018

Choosing Algorithms

- What is an algorithm?

- What factors do we consider when choosing?
  - Familiar design
  - Ease of coding
  - Efficiency
    - Runtime
    - Memory

Algorithmic Complexity

- Fact: The more data we process the longer it takes

- Question: Is the processing time always *proportional* to the size of the data set?

- How can we classify algorithms with regard to this?

- Let’s draw some graphs!

Linear Search

- Demonstration: Linear Search

- What shape will the runtime graph be?

- Roughly speaking, what happens to the runtime as the size of the dataset doubles?

- We classify *all* algorithms with this shape as “linear” algorithms.

Binary Search

- Demonstration: Binary Search

- What shape will the runtime graph be?

- What happens to the runtime as the size of the dataset doubles?

- We classify all algorithms with this shape as “logarithmic” algorithms.

Example: “Nervous Search”

1. Look in the first box.
2. If not there, start from the beginning and look through the first two boxes.
3. If not there, start from the beginning and look through the first three boxes.
4. Etc.

- Is this a good way to do a search? ☺️
- What shape will the runtime graph be?
- What happens to the runtime as the size of the dataset doubles?
- We classify all algorithms with this shape as “Quadratic” algorithms.

Comparing Algorithms

Suppose that on a fixed set of data:

- Person 1 will run a *linear* algorithm (A)
- Person 2 will run a *quadratic* algorithm (B)

- Can we say which one will run faster?

- What *can* we say with certainty about the performance of A vs. B?

- Let’s do the same analysis but comparing linear with logarithmic.

Intuition for Big-O Notation

We say an algorithm is…

- O(n) if it is linear (or faster)
- O(log n) if it is logarithmic (or faster)
- O(if it is quadratic (or faster)

Examples of Big-O "Categories"

$O(1)$
$O(\log n)$
$O(n)$
$O(n \log n)$
$O(n^2)$
$O(n^3)$
…
$O(n^{1000})$
…
$O(2^n)$
$O(72^n)$
…
$O(n!)$
$O(n^n)$

Observations:

- There are always categories "between"
  e.g.: What's between O(1) and O(log n)?
- O(1) is the fastest, but there is no slowest
- Recall: When comparing performance of two algorithms from different categories, what can we say about performance?
- Why is the division between green/red in that spot?
- Why is it called "asymptotic" complexity?

Examples

What is the asymptotic complexity (Big-O) for these?

- Linear Search
- Binary Search
- Coloring in every square of a Flag of height n
- Count enemies remaining on n by n battlefield
- Find closest enemy within radius r on an n by n battlefield

More Big-O Examples

What is the asymptotic complexity (Big-O) for these?

- Find center of gravity for n by n image of geographic region
  - With finite “sampling”
- Inserting at the front of an array
- Inserting at the front (head) of a linked list
- Access the nth element of an array
- Access the nth element of a linked list