Lecture 23: Sorting and Complexity

CSCI 101

Spring 2018

Today

- Announcements
  - Thursday: Bring computer for evaluations
  - Final exam: self-scheduled; 2 sheets of notes allowed
- Computational Complexity
  - Big-O notation to describe # of operations
  - Last week: complexity of search algorithms
- Sorting Algorithms
  - Elementary methods are $O(n^2)$
  - Divide-and-Conquer methods are $O(n \log n)$

Big-O notation

- Use a function to describe number of basic operations in terms of input size
- The function includes only the dominant terms, ignoring constants
- Example: list traversal is $O(n)$ for a list of n values

Binary Search

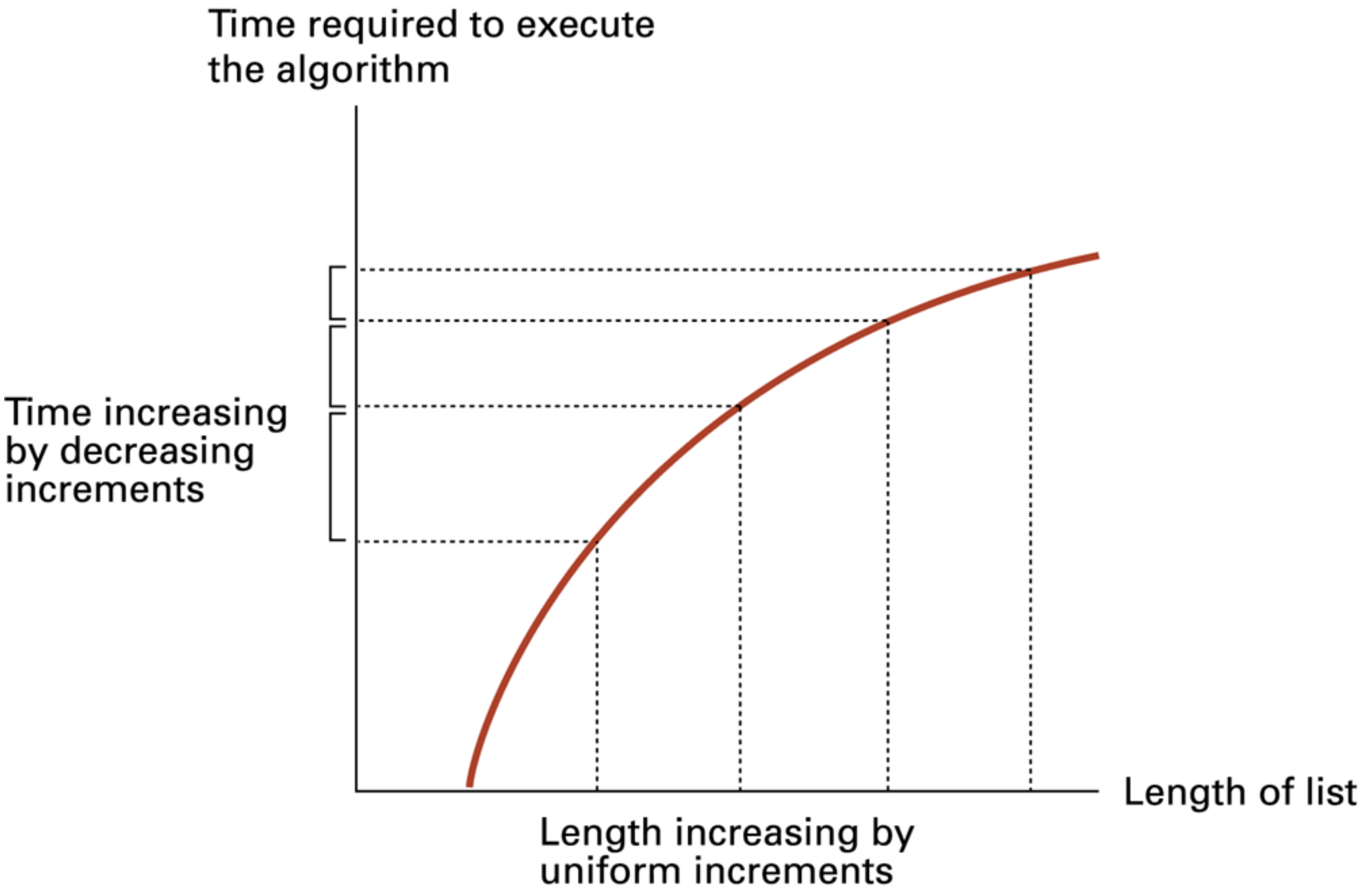
We can search faster if the list is sorted

⟶ Compare middle element to the target, then refine search to one half of list

2	5	8	11	15	16	21	24	29	41	45	58	71	85	92	95

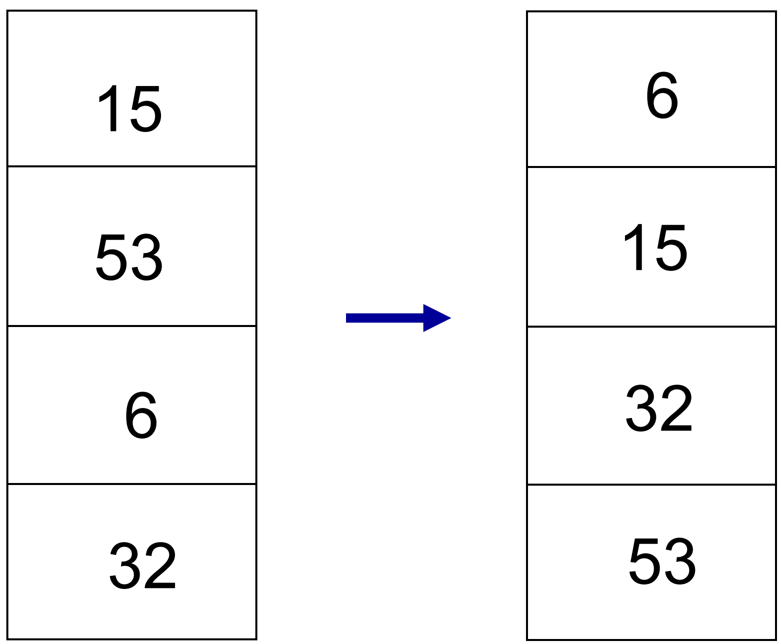
Number of operations for a list of n elements: $O(\log_2 n)$ or $O(\log n)$

Binary Search: O(log n)



Sorting

How to sort n values into increasing order?



Sorting Algorithms

Selection Sort → $O(n^2)$

Find smallest value and swap into first position. Repeatedly select the smallest n-1 times.

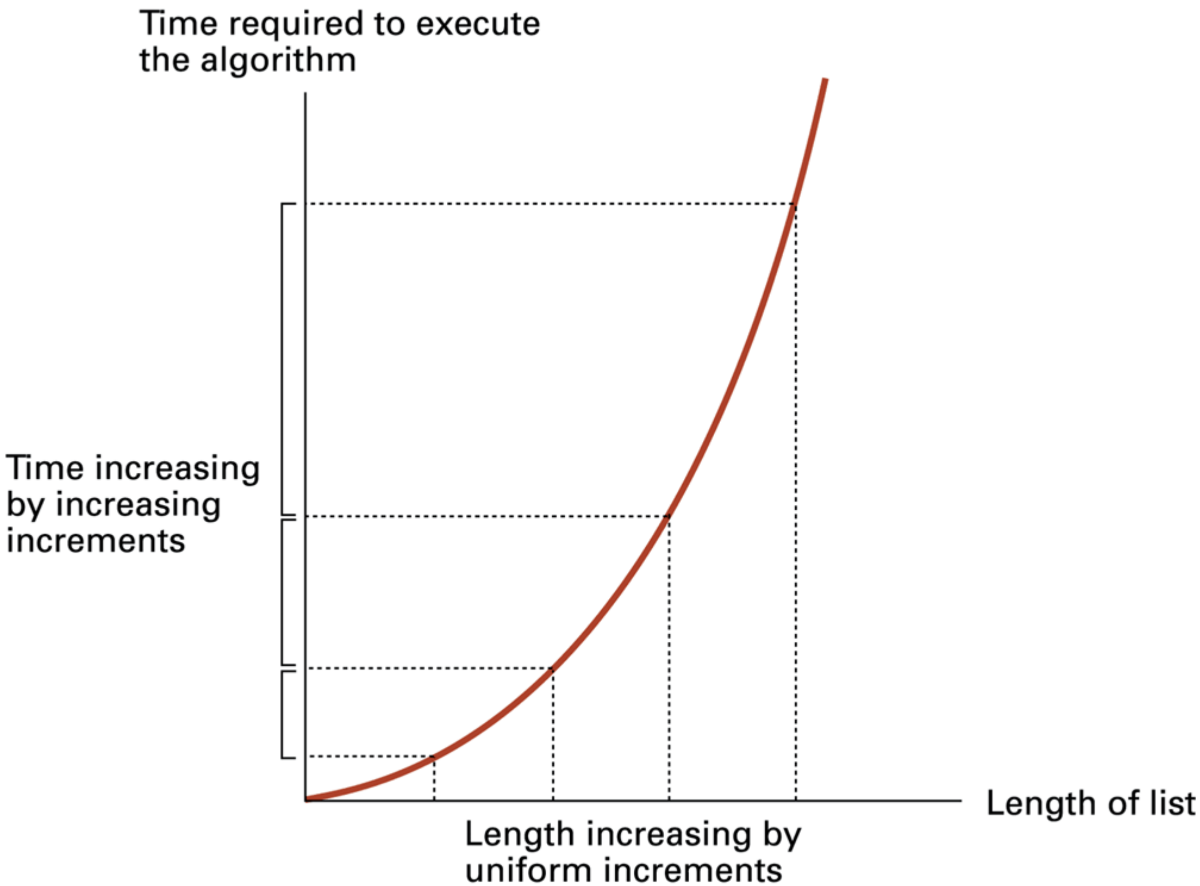
Insertion Sort → $O(n^2)$

Examine each element in turn, inserting it into its proper position among the already sorted values to the left.

Bubble Sort → $O(n^2)$

Swap neighboring values if out of order (largest bubbles to end). Do this n-1 times.

Insertion Sort is $O(n^2)$

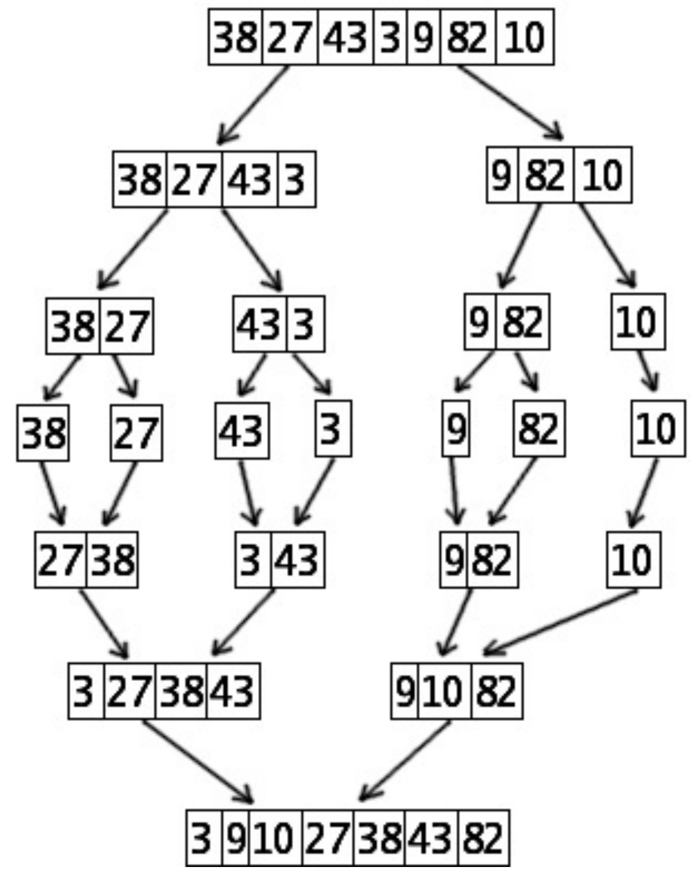


Merge Sort

If a list has only 1 item, then we're done

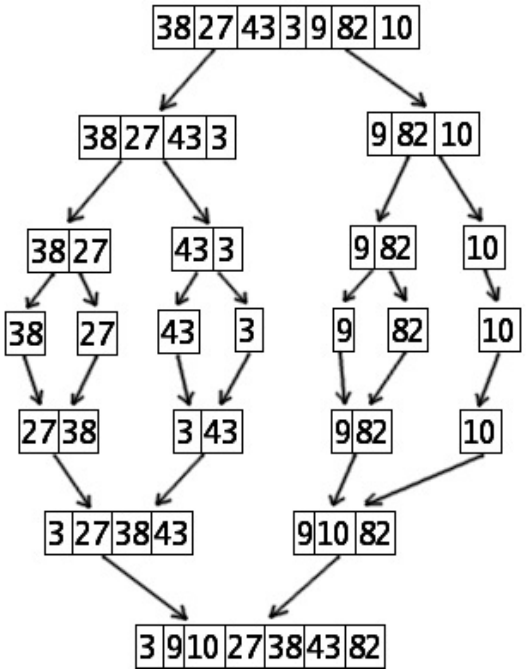
Else:

1. Split the list in half
2. Recursive sort each half
3. Merge the two sorted halves together (how?)



Merge Sort Complexity

- How many levels of splits?
- How many basic operations at each level?
  - Hint: how many basic operations to merge two lists of size n/2?
  - Hint: how many basic operations to merge four lists of size n/4?
- Merge sort complexity is O(log n) x O(n) = O(n log n)



Summary

- Computational complexity describes how the time needed to execute an algorithm increases as its input size increases
- Big-O notation summarizes an algorithm's complexity in terms of its input size
  - Linear search: O(n) Binary search: O(log n)
  - Selection sort: O(n^2) Insertion sort: O(n^2)
  - Bubble sort: O(n^2) Merge sort: O(n log n)