Best, Worst, and Average Case Analysis Overview

This video explains time complexity analysis for algorithms using linear search and binary search tree (BST) examples, focusing on best case (minimum time), worst case (maximum time), and average case times, measured by number of comparisons.

Key principle: Cases represent different input scenarios; asymptotic notations (**Big O**, **Omega**, **Theta**) can apply to any case.

---

1. Linear Search

Linear search scans a list of **n elements** from left to right, checking each element until the **key** is found or the list ends.

Examples:

- Successful search for **7** (at index 5): 6 comparisons.
- Unsuccessful search for **20**: n comparisons (checks entire list).

Best Case

- **Scenario**: Key at **first index** (e.g., searching for **8**).
- **Time**: **1 comparison** (constant time).
- **Notation**: Theta(1), O(1), or Omega(1)

Worst Case

- **Scenario**: Key at **last index** or not present (e.g., **18** or **20**).
- **Time**: **n comparisons**.
- **Notation**: Theta(n), O(n), or Omega(n)

Average Case

- **Definition**: Total time across all possible cases ÷ number of cases.
- **Cases**: Key at position 1 (1 comp), 2 (2 comp),..., n (n comp).

- **Formula**: frac{1 + 2 + dots + n}{n} = frac{n+1}{2}
- **Notation**: Theta(n) (linear, similar to worst case).
- **Note**: Rarely computed due to difficulty; worst case often preferred.

Linear Search Summary Table

Case	Scenario	Comparisons	Asymptotic Notation
Best	Key at index 1	1	Theta(1)
Worst	Key at index n/not found	n	Theta(n)
Avg	All positions equal likely	frac{n+1}{2}	Theta(n)

---

2. Binary Search Tree (BST) Search

BST: Elements organized with **smaller values** on left, **larger** on right of each node. Search follows this rule.

Example: Search for **15** in balanced BST (root=20): 3 comparisons (height = log n).

Time depends on tree height (h): comparisons = h + 1.

Best Case

- **Scenario**: Key at **root** (e.g., **20**).
- **Time**: **1 comparison**.
- **Notation**: $$\Theta(1)$$

Worst Case

- **Scenario**: Key at **leaf** node.
- **Time**: Depends on height **h**:

Tree Type	Height (h)	Time
Balanced	log n	O(log n)
Skewed	n	O(n)

- **Minimum worst case**: O(log n) (balanced tree).

- **Maximum worst case**: O(n) (left/right skewed tree).

Diagram of BST Heights

Key Takeaways for Revision

🔑 **Core Concepts**:

1. **Best case** = minimum time scenario (constant for both algorithms).
2. **Worst case** = maximum time (n for linear; h for BST where $$h \in [\log n, n]$$).
3. **Average case** = expected time ($$\frac{n+1}{2}$$ for linear).
4. **Any asymptotic notation** works for any case (not fixed: O/Ω/Θ flexible).