Binary Search Algorithm

Instructor: Alice Wang

awang@capilanou.ca

Department of Computing Science

Office Hours: 10 - 11am on Tues., Fri. (FR533)

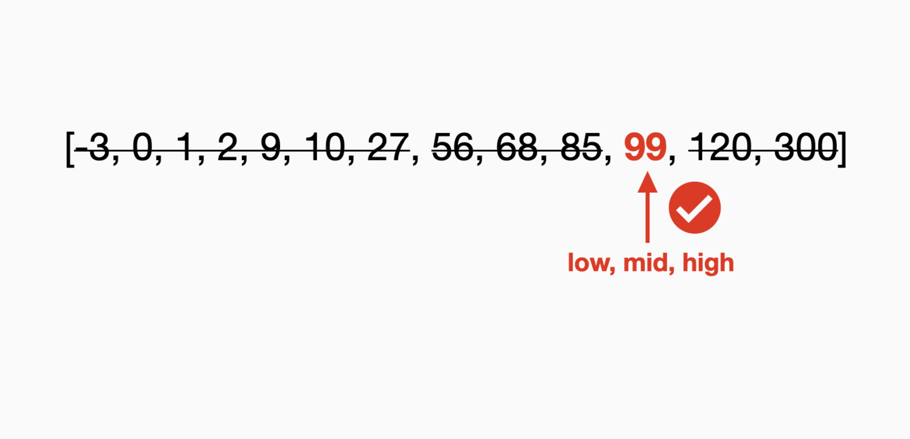
Overview

1. Understand binary search (example)
2. Design the algorithm (pseudocode)
3. Algorithm analysis (time complexity)
4. Binary search's applications
5. Binary search's limitations and alternatives

What is binary search?

A **super** efficient algorithm for finding target value within a **sorted** array.

Binary search example



[~~-3, 0, 1, 2, 9, 10, 27~~, ~~56, 68, 85~~, 99, ~~120, 300~~]

Design the binary search algorithm (pseudocode)

```
Start searching the entire array using lowest and highest indices.
Repeat the following steps until the search range is empty:
    Find the middle position of the current search range.
    If the middle value is less than the target value:
        Narrow the search range to the higher half.
    Else if the middle value is greater than the target value:
        Narrow the search range to the lower half.
    Else (the middle value equals the target):
        Find the target and return its index.
The search range becomes empty, return -1 indicating the target is not found.
```

Note: Algorithm implementations posted on the course GitHub.

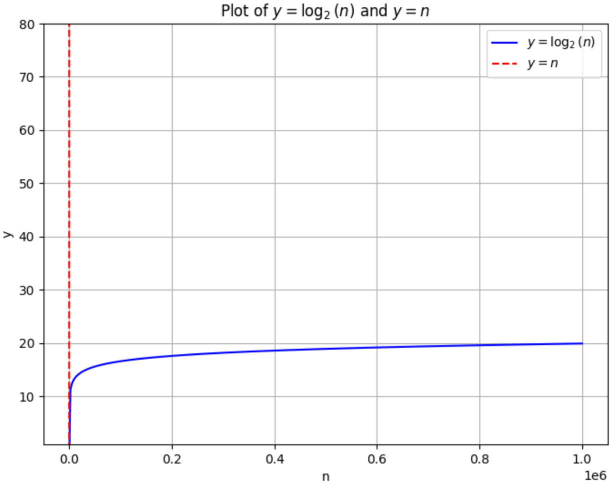
Algorithm analysis (time complexity - worst case)

$$\begin{aligned}
T(n) &= T(n/2) + 1 \\
&= T(n/4) + 1 + 1 \\
&= T(n/8) + 1 + 1 + 1 \\
&\vdots \\
&= T(2 \text{ or } 3) + 1 + \cdots + 1 \\
&= T(1) + 1 + 1 + \cdots + 1 \\
&= 1 + \log_2 n \\
&= \mathcal{O}(\log_2 n)
\end{aligned}$$

Feel the time complexity - King’s Chessboard

1	2	4	8	16	32	64	128
256	512	1024	2048	4096	8192	16384	32768
66k	131k	262k	524k	1.0m	2.1m	4.2m	8.4m
16.8m	33.6m	67.1m	134m	268m	537m	1.1b	2.1b
1.1t							
							9.2×10^{18}

Feel the time complexity



Binary search's applications

- **Guess game**
- **Search in a dictionary**
- **Find a nearest target** (LeetCode practices)
- **Root search of a function**
- **git bisect**
- **Interpolation search** (data roughly uniformly distributed)
- **Exponential search** (data unbounded)
- **Data structure: Binary Search Tree, B Tree**

Binary search's limitations and alternatives

Limitations:

sorted and **array**.

Alternatives:

1. **Linear search**
2. **Interpolation search**
3. **Exponential search**
4. **Hash table**: when order is not required.
5. **Binary search tree (self-balanced)**: AVL tree, red black tree.

Q & A

References

- Website:
  ”https://learn.zybooks.com/zybook/CAPILANOUCOMP120Fall2021”

- Thomas H. Cormen, et al. (2009)
  Introduction to Algorithms
  3rd Edition, MIT Press