Examples and Applications of Binary Search

CSCI2100 Tutorial 1
Shangqi Lu

Adapted from the slides of the previous offerings of the course
Outline

• We will first review the binary search algorithm through an example
• And then use the algorithm to solve a "two-sum" problem.
Binary Search Review

- Suppose we have the following sorted input set $S$, and are trying to find the value 13.
Binary Search Review

- Initializing $L$ to be 1 and $R$ to $n$ (in this case, 8)
Binary Search Review

• Since $L \leq R$
• Proceed by computing $M = (L + R)/2$
Binary Search Review

• Compare $v = 13$ and the value 8 indexed by $M$
• $v >$ the value indexed by $M$
• Means that the target is in the right half of the sorted sequence
Binary Search Review

• Look at the right half of the sorted sequence
• Set $L$ to be $M + 1$ (discard the left half)
• Recompute $M$
Binary Search Review

- Compare $v$ and the value 21 indexed by $M$
- $v < $ the value indexed by $M$
- Means that the target is in the left half of the sorted sequence
Binary Search Review

• Set \( R \) to be \( M - 1 \) (discard the right half)
• \( L, R, M = 5 \)
• \( v \) = the value indexed by \( M \), return “yes”
The Two-Sum Problem

• Problem Input:
  • A sequence of $n$ positive integers in strictly increasing order in memory at the cells numbered from 1 up to $n$
  • The value $n$ has been placed in Register 1
  • A positive integer $v$ has been placed in Register 2

• Goal:
  • Determine whether if there exist two different integers $x$ and $y$ in the sorted sequence such that $x + y = v$
Example

• A “yes”-input with $n = 12$, $v = 30$
Example

• A "no"-input with $n = 12$, $v = 29$
A First Attempt

- Naïve algorithm:
  - Enumerate all possible pairs in the sorted sequence
  - Check if they sum to $v$
  - There are $\binom{n}{2} = \frac{n(n-1)}{2}$ possible pairs
  - Worst-case time: at least $n(n - 1)/2$

- Can we do better than this?
  - Hint: Take advantage of the fact that the given sequence is sorted!
Binary Search the Answer

• Goal: Find a pair \((x, y)\) such that \(x + y = v\)
• Observe that given \(x\), \(y = v - x\), is determined
• Improve the naïve algorithm
  • Instead of enumerating all possible \(y\), we can find if there exits an integer \(v - x\) in the sequence
• Solution:
  • For each \(x\) in the sequence:
    • set \(y\) as \(v - x\)
    • Use binary search to see if \(y\) exists in the sequence
The Repeated Binary Search Algorithm

- Pseudocode:

1. Let $n$ be register 1 and $v$ be register 2
2. register $i \leftarrow 1$, register $one \leftarrow 1$
3. **while** $i \leq n$
4. read into register $x$ the memory cell at address $i$
5. $y \leftarrow v - x$
6. **if** $BinarySearch(y) = \text{"yes"}$
7. **return** \text{"yes"}
8. $i \leftarrow i + one$ (effectively increasing $i$ by 1)
9. **return** \text{"no"}
Worst-Case Running Time

• Worst case (when the output is “no”)
• This algorithm needs to run binary search \( n \) times
• Cost of each binary search: at most \( 10(1 + \log_2 n) \)
• Cost of the algorithm: at most \( 100n(1 + \log_2 n) \) (a loose upper bound)

• Can we do even better?
• Actually this problem can be solved in at most \( 100n \) time --- left for you to try outside the class.