Survey of Binary Search

Lokesh Podipipireddy

April 2017

1 Abstract

The purpose of this document is to explore binary search in various shapes that are set out to accomplish different tasks. We also analyze the different variations of binary search in relation to the general scheme of binary search.

2 Introduction

In this paper, we discuss the general schema of binary search. From there we examine the bisection method and exponential search. Bisection method is for finding the root of a function while exponential search is used for finding an element in a sorted list. We try to analyze the difference and similarties between these variations and the general schema

3 Binary Search

Binary search is a search algorithm that finds the position of a target value inside of a sorted array. It starts off by comparing the target value with the middle element of the sorted array. If the value is unequal it elminates one half side of the array and compares the target value to the middle element of the array that isn't eliminated. It continues to do this until the middle element is found.

The way it determines which half to eliminate is by comparing the target value to the middle element of the array. If the middle element is less than the target value, it eliminates all the element from the beginning of the array to the middle element. If the middle element is greater than target value, it eliminates all the elements from the element right after the middle of the array to the end of the array.

4 Binary Search General Schema

```
{a < b}
x, y := a, b
```

```
do x + 1 < y ->
    m := (x + y) / 2
    if arr[m] < key -> x := m
    [] arr[m] > key -> y := m //[] is the guard
    fi
od
```

As you can see, given an index where is x is the start and y is the end (x¡y) we and a given key to compare against mid-pont. We can determine which half of the array to eliminate.

5 Binary Search in Bisection Method

The Bisection is a classic method in numerical computation that is used to find the roots of a given function. Let's say you have a given function f that we want to the roots. We know that in order to the root of a function, we find the x-intercept of the function. The way we do this by starting out with two guesses for the x-intercept. Let's call this a and b. We then take the value of f(a) and f(b) and multiply them together. If this value is greater than 0 than we change a:=m. If this value is less than 0, we change b:=m and then repeat the process until we have zeroed in on the root of the function.

```
{a < b}
x, y := a, b
do x + 1 < y ->
    m := (x + y) / 2
    if f(a)*f(b) > 0 -> x:=m
    [] f(a)*f(b) < 0 -> y := m //[] is the guard
    fi
od
```

6 Similarties and differences from binary search

Bisection method is definately an instance of binary search. The calculation of midpoint and assigning the start and end index to midpoint is the same. The difference arises in the gaurd statements and instead of an array, it using a continous function. From gaurd statements you can tell instead of checking against some midpoint, it is checking to see if product of f(a) and f(b) is greater than 0. this is because this method assumes the key to be f(x) = 0 and it trying find that x value.

7 Binary Search in Exponential Search

Exponential Search is another searching algorithm. It works on an unbounded, infinite list. What it does is, it determines the approximate range that a given key would resides in and performs binary search in that range. Assuming that the list is sorted in ascending order, the algorithm looks for the first exponent, j, where the value of the exponent is greater than the search key. It then performs binary search on the subarray where the beginning index is exponent of j-1 and the ending index is exponent of j.

```
{a < b}
bound = 1
do bound < size && arr[bound] < key ->
    bound = bound*2
od

x, y := bound/2, bound
do x + 1 < y ->
    m := (x + y) / 2
    if arr[m] < key -> x := m
    [] arr[m] > key -> y := m //[] is the guard
    fi
od
```

8 Similarties and differences from binary search

There really isn't much of a difference in the binary search portion of it but there is an addition which is determining the range to operate within.

9 Conclusion

The Binary Search Algorithm can be found in many new algorithms with variations on the operands and the subarray elimination gaurds as well. It may not be as easy to view these algorithms through the lens of binary search because they operate under different circumstances. For example bisection method is not a comparison of a key to the midpoint. We treat f(a)=0 to be the midpoint and 0 is the key that is being compared. The bisection method operates on a continous function which means it can be reprented by a continous graph and easier to visualize the process of the bisection through this lens. At the end, the general priniciple of binary search exists in bisection method and exponential method.

10 References

Bentley, Jon L.; Yao, Andrew C. (1976). "An almost optimal algorithm for unbounded searching". Information Processing Letters.

Willams, Jr., Louis F. (1975). A modification to the half-interval search (binary search) method. Proceedings of the 14th ACM Southeast Conference. pp. 95–101.