Análisis de algoritmos

Erick Gonzalez Parada ID: 178145
Matemáticas discretas, Universidad de las Américas Puebla, Puebla, México

September 2, 2023

Abstract

El análisis de estos algoritmos de búsqueda
permite conocer y empezar a dominar los conceptos del big-o notation

Keywords: búsqueda, algoritmo

1 Introducción

Primer algoritmo *linear search* se enfoca principalmente en buscar un elemento especifico en un array (arreglo) retornando en este caso verdadero si el elemento solicitado se encuentra dentro de este y retorna falso si el elemento solicitado no se encuentra en el.

El segundo algoritmo *binary search* también trata de buscar el elemento solicitado sin embargo función de manera diferente utilizando el concepto de divide y vencerás de tal manera que literal va dividiendo el espacio de búsqueda, sin embargo primero el array tiene que estar ordenado ya que lo siguiente después de dividir es preguntar si el elemento solicitado es mayor o menor que la posición en la que se encuentra, a partir de ahí básicamente volvemos a partir hasta que lleguemos a tener dos elementos y simplemente preguntaremos si es uno o el otro, si lo encontró regresamos positivo, sino regresamos negativo.

2 Explicación de los algoritmos

A continuación se presentaran los algoritmos con comentarios explicando el pseudocódigo paso a paso
Nota: en pseudocódigo los comentarios son denotas con un '%' y no pueden contener acentos.

linear search

```
% Parametros ocupados antes y para que la busqueda lineal funcione
Requiere: list a, size of list n, desired item x
% Empieza el primer for loop
% notar que el for loop esta 0 indexado
  for i=0 to n-1 do
  % si el elemento actual es el que estamos buscando
  % lo retornamos y ahi acaba todo
    if a[i]=x then
      return i
    end if
    % iteramos al siguiente elemento
    i=i+1
  % terminamos for loop
  end for
% aqui el pseudocodigo esta representando la ultima evaluacion
```

```
% donde nuestro iterador excede n-1 y con esto indicando que
% no se encontro el elemento deseado por lo que se retorna falso
if i=n then
  return false
end if
```

Binary search

```
% Parametros en esta ocacion son tener la lista ordenada
% de mayor a menor o de menor a mayor pero tiene que estar
% ordenada y pues tambien requerimos como parametro al
% elemento que deseamos obtener que lo retorna si lo ha encontrado
% notar que esta vez estamos indexando desde 1
binaryS(A[1...n],x)
  % variable entero izquierda apunta al primer elemento
  left = 1
  % variable entero derecha apunta al ultimo elemento
  right = n
    % el siguiente while loop es mientras
    % el valor del ultimo elemento sea mayor
    % o igual que nuestra primer apuntador
    while(right >= left)
      % un ultimo apuntador
      % que apunta en medio
      % del rango de los valores
      % right & left (floor)
      mid = left + [(right - left)\2]
      % si nuestro apuntador de
      % en medio ya apunta al
      % elemento deseado
      % retorna el elemento
      if x = A[mid]
        return mid
      % si no y x es mayor que
      % el elemnto que esta en el
      % medio de nuestro intervalo actual
      % entonces movemos nuestro apuntador de
      % la izquierda a lo que vale el apuntador
      % de en medio + 1
      if x > A[mid]
        left = mid + 1
      % si no movemos nuestro apuntador de la
      % derecha a lo que vale el apuntador de
      % en medio - 1
      if x < A[mid]
        right = mid - 1

    % si no encontramos nada hasta que el while loop da falso
    % retornamos null o bien pues que no encontramos el elemento deseado
    return NULL
```

3 Análisis del algoritmo

Utilizando conteo por bloques se van a analizar ambos algoritmos.

linear search

block	frequency	description
1	n + 1	the for loop
2	n	whats inside the for loop

Figura 1: linear search

Función característica:

$$2n + 1$$

Complejidad asintótica

$$O(n)$$

binary search

block	frequency	description
1	2	the left & right constants
2	log n + 1	whats inside the while loop (the space is divided every time

Figura 2: binary search

Función característica

$$2 + \log 2n + 1$$

Complejidad asintótica

$$O(k + log(n))$$

donde k es una constante.

comparando gráficas donde se quitan las constantes en ambas notaciones asintóticas

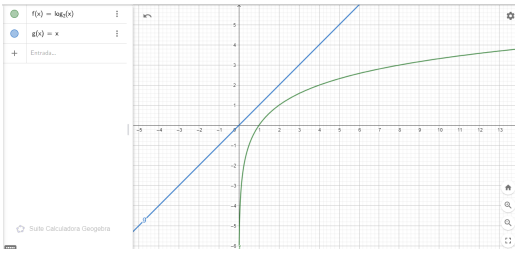

Figura 3: gráficas

4 Conclusiones

Concluyendo es importante reconocer que estas son los algoritmos principales para la búsqueda de algún elemento y llevándolos a la práctica de la vida real y en lo personal aplicaría linear search en los mínimos casos en donde se busca un elemento dentro de una lista (o array) pequeño mientras que la búsqueda binaria aplica excelente para cuando la lista (o array) es de proporciones muy grandes y en la vida real sirve para buscar entre millones de datos.

En otras palabras linear search llega a ser muy poco eficiente en contra de binary search y lo podemos comprobar nuevamente viendo la figura 3.