آنالیز الگوریتم:

(بخش دوم)

"بدترین"، "میانگین" "بهترین"

در پست قبلی بحث کردیم که چگونه تحلیل مجانب بر مشکل تحلیل الگوریتم ها غلبه می کند.

در این پست نمونه ای از جستجوی خطی را می گیریم و آن را با استفاده از تحلیل مجانبی تحلیل می کنیم.

برای تحلیل یک الگوریتم می توانیم سه حالت داشته باشیم:

1) بدترین حالت

2) حالت میانگین

3) بهترین حالت

اجازه دهید پیاده سازی زیر از جستجوی خطی را در نظر بگیریم.

C++

Linear Search
main

Line...arch 〉 My Mac Running Linear Search : Linear Search

C⁺ main

Linear Search 〉 Linear Search 〉 C⁺ main 〉 No Selection

```cpp
//  Linear Search
//
//  Created by Shayan Aryania

#include <iostream>
using namespace std;

// Linearly search x in arr[]. If x is present then return the index, otherwise return -1
int search(int arr[], int n, int x)
{
    int i;
    for (i = 0; i < n; i++) {
        if (arr[i] == x)
            return i;
    }
    return -1;
}

// Driver Code
int main()
{
    int arr[] = { 1, 10, 30, 15 };
    int x = 30;
    int n = sizeof(arr) / sizeof(arr[0]);
    cout << x << " is present at index "
        << search(arr, n, x);

    getchar();
    return 0;
}
```

Linear Search Line: 8 Col: 32

```
30 is present at index 2
```

All Output Filter

برای دانلود فایل کامل کدها به لینک
گیت هاب
که در هایلایت درج شده مراجعه کنید.

oo+oooooooo>>

Python

LinearSearch.py — Desktop

LinearSearch.py

```python
# Linear Search
#Shayan Aryania

def search(arr, x):
    for index, value in enumerate(arr):
        if value == x:
            return index
    return -1


# Driver Code
arr = [1, 10, 30, 15]
x = 30
print(x, "is present at index",
    search(arr, x))
```

PROBLEMS TERMINAL ··· Python Debug Console

```
/bin/python3.10 /Users/shayan/.vscode/ex
tensions/ms-python.python-2022.2.1924087
327/pythonFiles/lib/python/debugpy/launc
her 50135 -- /Users/shayan/Desktop/Linea
rSearch.py
30 is present at index 2
shayan@Shayans-MBP Desktop %
```

main* 0 0 Ln 16, Col 1 Spaces: 4 UTF-8 LF {} Python

برای دانلود فایل کامل کدها به لینک
گیت هاب
که در هایلایت درج شده مراجعه کنید.

@ShayanAryania

ooo|oooooo>>

تجزیه و تحلیل

بدترین حالت

در تحلیل بدترین حالت، ما حد بالایی را در زمان اجرای یک الگوریتم محاسبه می‌کنیم.

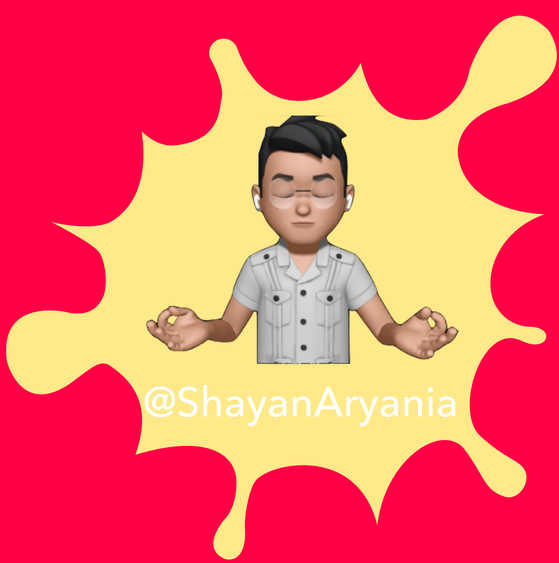
ما باید موردی را بدانیم که باعث می شود حداکثر تعداد عملیات اجرا شود.
برای جستجوی خطی، بدترین حالت زمانی اتفاق می‌افتد که عنصر مورد جستجو در آرایه وجود نداشته باشد.

وقتی x وجود ندارد، تابع search() آن را با تمام عناصر arr[] یکی یکی مقایسه می کند.

بنابراین، بدترین حالت پیچیدگی زمانی جستجوی خطی $\Theta(n)$ خواهد بود.

WORST CASE SCENARIO

این تحلیل عموما برای تمام الگوریتم ها در نظر گرفته میشود و معیار تحلیل زمانی قرار میگیرد.



تجزیه و تحلیل

حالت میانگین

در تحلیل حالت میانگین، همه ورودی‌های ممکن را می‌گیریم و زمان را برای همه ورودی‌ها محاسبه می‌کنیم.

تمام مقادیر محاسبه شده را جمع کنید و مجموع آن را بر تعداد کل ورودی ها تقسیم کنید.

ما باید توزیع موارد را بدانیم (یا پیش بینی کنیم).
برای مسئله جستجوی خطی، فرض کنیم همه موارد به طور یکنواخت توزیع شده اند .

بنابراین همه موارد را جمع می کنیم و حاصل را بر (n+1) تقسیم می کنیم. در زیر مقدار پیچیدگی زمانی متوسط مورد آمده است.

$$\sum_{i=1}^{n} \frac{\Theta(i)}{(n+1)} = \frac{\Theta((n+1) * \frac{(n+2)}{2}}{(n+1)}$$

تجزیه و تحلیل

بهترین حالت ممکن

در بهترین حالت، ما کران پایینی زمان اجرای یک الگوریتم را محاسبه می کنیم.

ما باید موردی را بدانیم که باعث می شود حداقل تعداد عملیات اجرا شود. در مسئله جستجوی خطی، بهترین حالت زمانی رخ می دهد که عنصر مورد نظر در مکان اول حضور داشته باشد.

تعداد عملیات در بهترین حالت ثابت است (به n وابسته نیست).
بنابراین پیچیدگی زمانی در بهترین حالت $\Theta(1)$ خواهد بود.

BEST CASE SCENARIO

اغلب اوقات، ما تحلیل بدترین حالت را برای تجزیه و تحلیل الگوریتم ها انحام می دهیم. در بدترین تجزیه و تحلیل، ما یک کران بالای زمان اجرای یک الگوریتم را تضمین می کنیم که اطلاعات خوبی است.

تجزیه و تحلیل حالت میانگین در بیشتر موارد عملی آسان نیست و به ندرت انحام می شود. در تحلیل موردی متوسط، باید توزیع ریاضی همه ورودی‌های ممکن را بدانیم (یا پیش‌بینی کنیم).

تحلیل بهترین حالت جعلی است. تضمین کران پایین در یک الگوریتم هیچ اطلاعاتی را ارائه نمی دهد، زیرا در بدترین حالت، اجرای یک الگوریتم ممکن است سال ها طول بکشد.

تجزیه و تحلیل

بهترین حالت ممکن

برای برخی از الگوریتم ها، همه موارد به طور مجانبی یکسان هستند،
یعنی هیچ بدترین و بهترین حالت وجود ندارد.

مثلا Merge Sort در همه حالت ها،
عملیات را در $\Theta(n\log n)$ را در انحام می دهد.

اما اکثر الگوریتم‌های مرتب‌سازی دیگر
بدترین و بهترین حالت‌ها را دارند.

به عنوان مثال،

QuickSort()
مرتب سازی سریع

بدترین حالت = آرایه ورودی از قبل مرتب شده باشد
بهترین حالت = عناصر محوری همیشه آرایه را به دو نیمه تقسیم کنند.

مرتب‌سازی درجی
InsertionSort()

بدترین حالت = آرایه مرتب‌سازی معکوس شود
بهترین حالت = آرایه به همان ترتیب خروجی مرتب شود.

Follow

www.instagram.com/shayanaryania

www.linkedin.com/in/shayanaryania

Like & Share

Save