Lecture 10

Thursday, September 24, 2020 5:00 PM

Reminder: * Hw4 & lab1 are due this Sunday

* Exam 1 is on Oct 8th

Binary Search: $T(n) = O(\log n)$

```
BS (a, s, e, key)        // a: array

mid = (s+e)/2
if s <= e
    if a[mid] > key
        return BS (a, s, mid, key)   // or BS (a[0:mid-1], key)
    else if a[mid] < key
        return BS(a, mid+1, e, key)
    else   // a[mid] == key
        return True
    end
end
return False
```

Example: A machine on the average can run a single line/step in 10^{-8} sec. What is the largest input size in which the code finishes in 1 sec?

total runtime/steps/lines

	T(n)	time to finish	
	1	10^{-8}	
BS →	$\log n$	$10^{-8} \log n = 1$	$\Rightarrow \log n = 10^8 \Leftrightarrow n = 2^{10^8} = (2^{10})^{10^7} \simeq (10^3)^{10^7} = 10^{30{,}000{,}000}$
LS →	n	$10^{-8} n = 1$	$\Rightarrow \boxed{n = 10^8}$
	n^2	$10^{-8} n^2 = 1$	$\Rightarrow n^2 = 10^8 \Rightarrow \boxed{n = 10^4 = 10{,}000}$
NP	2^n	$10^{-8} 2^n = 1$	$\Rightarrow 2^n = 10^8 \Rightarrow n = \log 10^8 = 8 \log 10 \simeq 8 \times 3.\sim \simeq \boxed{24}$

to have fun: $2^{10} = 1024 \simeq 1000 = 10^3$

f_{700} + ~~x+y~~

$\boxed{f_n = f_{n-1} + f_{n-2}}$

waste memory : Dynamic Programming

f_{200} — $x+y$, x, y — f_3, f_2, f_1, f_0

$= 8 \log 10 \simeq 8 \times 3.\sim \simeq \boxed{24}$

f_{199} $f_{198} \to f_{197}$ f_{196}

Similar to lab7.b

Example: You run an algorithm with the time complexity of $O(n^3)$ in 8sec. The input size was 10^4. How much time does it take to run on single step on your laptop?

$T(n) = O(n^3)$

1 line $\to$ X sec

2 lines $\to$ 2X

$\sim n^3 \to n^3 x = 8$ timer in code

$(10^4)^3 x = 8$

$10^{12} x = 8$

$\to \boxed{x = 8 \times 10^{-12} \text{ sec}}$

$\boxed{x = 8 \times 10^{-3} \text{ ns}}$

Example:

for $i = 1:n$

    BS(a, key) // $|a| = n^2 \log n$, key in not in a

    $\hookrightarrow |a| = n \Rightarrow O(\log n)$

end

e.g. $4^2 \log 4 = 16 \times 2 = 32$

$$T(n) = \sum_{i=1}^{n} T_{BS}(a, key) = \sum_{i=1}^{n} O(\log(|a|))$$

$$= \sum_{i=1}^{n} O(\log(n^2 \log n)) = O(n \log(n^2 \log n))$$

$$= O(n(\log n^2 + \cancel{\log \log n}))$$

$$= O(n \log n^2)$$

$= O(n \log n)$

Example:

for $i = 1 : n^3$ ← $i = 1, 2, 3, 4, 5 \text{ — } n, n+1, n+2 \text{ — } n^2, n^2+1, n^2+2 \text{ — } n^3$

if $i \leq n^2$ ← $i = 1\ 2\ 3 \text{ — } n^2$

BS(a, key) // $|a| = n$, key is the last element

else ← $n^2+1, n^2+2 + \sim, n^3$

Selection Sort(a) // $\Theta(n^2)$ (reversed: 5 4 3 2 1)

end

end

$$T(n) = \underbrace{\sum_{i=1}^{n^2} O(\log n)}_{\text{for + if}} + \sum_{i=n^2+1}^{n^3} \Theta(n^2)$$

Note: $\sum_{i=a}^{b} c = (b-a+1)c$

$$= O(n^2 \log n) + (n^3 - (n^2+1) + 1)\,\Theta(n^2)$$

$$= O(n^2 \log n) + (n^3 - n^2)\,\Theta(n^2)$$

$$= O(n^2 \log n) + O(n^2(n^3 - n^2))$$

$$= O(n^2 \log n) + O(n^5 \cancel{- n^4})$$

$$= O(n^2 \log n) + O(n^5)$$

$$\boxed{= O(n^5)}\ ✓$$

Selection-Sort:

$a = [5, \cancel{2}, \cancel{1}, 6, \cancel{4}]$, $|a| = n$

| 1 | 2 | 4 | 5 | 6 | ✓

$a = [1, 2, 4, 5, 6]$

| 1 | 2 | 4 | 5 | 6 |

$$T(n) = n + (n-1) + (n-2) + (n-3) + \sim\sim + 1$$

$$= \underbrace{\sum_{i=0}^{n-1} (n-i)}_{\text{looking at the summation from start to end}} = \underbrace{\sum_{i=1}^{n} i}_{\text{from the end to the start}} = \Theta\left(\frac{n(n+1)}{2}\right) = \boxed{\Theta(n^2)}$$