Bubble Sort (HQ 2)

Shubham Agrawal. MT22129

Ayush Agrawal MT22095

(1) Recursive algo

// sorts A using Bubble Sort

Bubblesort(A):

1. n=|A|; BS(n)

// Puts the largest #J elements in their // correct posn. Assume that A is global.

BS(j):

1. If j=0: return.
2. BS(j-1)
3. j_0 = j // assuming j_0 is global.
4. Pushbig(1)

// assume that the (j-1) largest elements have been found out & put in their correct place. the following function (Pushbig(k)) puts the J^{th} largest element in its correct position.

```
Pushbig (k)                    // k is used to iterate
{                                 from 1 to n-j+1
1.   if (k == n-j+1)                          correct
       return;                  where n-j+1 is the position
2.                              of the jth. largest element.
3.
4.   if (a[k] > a[k+1])
5.     swap(a[k], a[k+1])
6.
7.   Pushbig (k+1)
```

Proof of Correctness.

① Pushbig(k)

Induction Claim: To prove that pushbig(k) is correct, ie it places the $(j_0+1)^{th}$ largest element in its correct place given that the j_0 largest elements are at their correct places.

Base case: $(j_0 == n)$

this is trivially true coz this means that $(n-1)$ largest elements are at their correct places, hence the n^{th} largest element has to be at its correct place

without doing anything.

Induction Hypothesis: Assume that pushbig(k) places the j_0^{th} largest ~~largest~~ element at its correct place given the largest $(j_0 - 1)$ elements are correctly placed

proof:-

1) the $(j_0+1)^{th}$ largest element must be b/w indices (1 to $n-j_0$) coz the j largest elements occupy indices $n-j_0+1$ to n.

2) for any 2 consecutive elements $A[k-1], A[k]$ for $k \leq (n-j_0)$, line 1 is no-op (as $n-j_0+1 > n-j_0$). Two cases arise while executing line 4.

a) $a[k] > a[k+1]$, in that case we swap the two elements. Now index $(k+1)$ holds the greater of the two elements

(b) $a[k+1] \geq a[k]$. In that case,

we dont swap the elements. even then the $(k+1)^{th}$ element is the larger one Hence after line 5, $(k+1)^{th}$ inden necessarily holds the larger elemnt

3) By induction of k from 1 to $n-j_0$, we can show that $A[n-j_0]$ will be the largest among elements from $A[1]$ to $A[n-j]$. Since the j_0 largest elements have already been placed correctly $A[n-j_0]$ now holds the $(j_0+1)^{th}$ largest element.

proof of correctness of BS(j)

Base case: $j = 1$

after line 2, BS(0) returns doing nothing and after line 4 pushbig(1) also returns as $n == 1$ & $j_0 == 1$, doing nothing. This is the

correct behavior as single element is always sorted

Induction Hypothesis: Assume that

BS(j) correctly sorts j elements

Induction Claim: BS(j+1) correctly sorts j+1 elements

(a) line 1 is no-op for $j > 0$.
(b) line 2 correctly sorts j elements as stated by induction hypothesis.
(c) line 4 correctly places the (j_0+1) th largest element (Refer the proof & definition of pushbig())

Hence BS(j) is correct.

as BS(n) correctly sorts n elements & BubbleSort(A) just calls BS(n)

hence Bubblesort(A) sorts all the n elements of A correctly

(3) Time complexity analysis

(a) Time complexity of Pushbig(k):

In pushbig(k), k is an iterator which always runs from 1 to $n-j_0+1$. At every step, a constant amount of work is being done.

Hence time complexity ($Pb(n)$)

$\Rightarrow$ $\max_{j_0 = 1 \text{ to } n}(c \times (n-j_0+1))$

$j_0 = 1 \quad c \times (n) \longrightarrow$ Max

$j_0 = 2 \quad c \times (n-1)$

$\vdots$

$j_0 = n \quad c \times 1$

$$Pb(n) \leq cn$$

Time complexity of function BS (B(n))

$\Rightarrow B(n) \leq B(n-1) + Pb(n) + C_1$

(Please refer definition of function BS on page 1)

$\Rightarrow B(n) \leq B(n-1) + cn + c_1$.

Solving by substitution :

$B(n) \leq \left(B(n-2) + c(n-1) + c_1\right) + cn + c_1$

$\leq B(n-2) + 2cn + c_2$

$\leq B(n-3) + c(n-2) + c_1 + 2cn + c_2$

$\leq B(n-3) + 3cn + c_3$

$\vdots$

$\leq B(0) + ncxn + cn$.

now $B(0)$ is constant — { Base condition }

$B(n) \leq cxn^2 + c_k$.

$B(n) = O(n^2)$

Time complenity of function Bubble sort is asymptotically equal to that of function BS { please refer to definition of Bubblesort on page 1 } as Bubblesort does nothing but calls BS(j) once.

Hence Bubblesort is $O(n^2)$