CSCB63 – Design and Analysis of Data Structures

Anya Tafliovich[1]

[1]based on notes by Anna Bretscher and Albert Lai

how long do things take

Remember this algorithm?

```
i = 1
while i < len(A):
  v = A[i]
  j = i
  while j > 0 and A[j-1] > v:
    A[j] = A[j-1]
    j = j - 1
  A[j] = v
  i = i + 1
```

What do we count? Does it matter?

how long do things take

Let's try counting this way:

- get/set variables: 1 step
- function call: 1 + steps to evaluate each argument + steps to execute function
- return statement: 1 + steps to evaluate return value
- if/while condition: 1 + steps to evaluate the boolean expression
- assignment statement: 1 + steps to evaluate each side
- arithmetic/comparison/boolean operators: 1 + steps to evaluate each operand
- array access: 1 + steps to evaluate array index
- constants: free!

how long do things take

```
def InsertionSort (A):                       STEPS
    i = 1                                      2
    while i < len(A):                          5
      v = A[i]                                 5
      j = i                                    3
      while j > 0 and A[j-1] > v:              10 or 3
        A[j] = A[j-1]                          8
        j = j - 1                              4
      A[j] = v                                 5
      i = i + 1                                4
```

What assumptions did we make? Are they realistic?

So, what's the total number of steps?

how long do things take

In the **worst case**:

- line 1: once : 2 steps

For $n \geq 1$:

- line 2: $n-1$ times (true) + 1 time (false) : $5n$ steps
- lines 3, 4, 8, 9: $n-1$ times : $(5+3+5+4)(n-1) = 17n-17$ steps
- line 5: for each i: i times (true) + 1 time (false) : $10i+3$ steps
- lines 6, 7: for each i: i times : $(8+4)i = 12i$ steps

$$
\begin{aligned}
&2 + 5n + 17n - 17 + \sum_{i=1}^{n-1}(10i + 3 + 12i) \\
=&22n - 15 + \sum_{i=1}^{n-1}(22i + 3) \\
=&22n - 15 + 22\frac{(n-1)n}{2} + 3(n-1) \\
=&11n^2 + 14n - 18
\end{aligned}
$$

how long do things take

In the **best case**:

- line 1: once : 2 steps

For $n \geq 1$:

- line 2: $n-1$ times (true) + 1 time (false) : $5n$ steps
- lines 3, 4, 8, 9: $n-1$ times :
  $(5+3+5+4)(n-1) = 17n - 17$ steps
- line 5: for each i: 1 time (false) : 10 steps
- lines 6, 7: for each i: 0 times : 0 steps

$$\begin{aligned} 2 + 5n + 17n - 17 + \sum_{i=1}^{n-1} 10 &= 22n - 15 + (n-1)10 \\ &= 32n - 25 \end{aligned}$$

how long do things take

What if we write the same algorithm differently?

```
0 def InsertionSort (A):
1     n = len(A)
2     for (i = 1; i < n; i++):
3         for (j = i; j > 0 and A[j] < A[j-1]; j--):
4             swap A[j], A[j-1]
```

- line 1: once, 4 steps

For $n \geq 1$:

- line 2: 1 step (once) + 2 steps (once) + 3 steps (n times) + 2 steps ($n-1$ times)
- line 3: for each i: 1 step (once) + 3 steps (once) + 11 steps (i times) + 2 steps (once) + 2 steps (i times)
- line 4: for each i: 9 steps (i times)

how long do things take

- line 1: once, 4 steps

For $n \geq 1$:

- line 2: 1 step (once) + 2 steps (once) + 3 steps (n times) + 2 steps ($n-1$ times)
- line 3: for each i: 1 step (once) + 3 steps (once) + 11 steps (i times) + 2 steps (once) + 2 steps (i times)
- line 4: for each i: 9 steps (i times)

$$4 + 1 + 2 + 3n + 2(n-1) + \sum_{i=1}^{n-1}(1 + 3 + 11i + 2 + 2i + 9i)$$

$$= 5n + 5 + \sum_{i=1}^{n-1}(22i + 6)$$

$$= 5n + 5 + 22\frac{(n-1)n}{2} + 6(n-1)$$

$$= 11n^2 - 1$$

Is this the same running time? In what sense?

how long do things take

Q. What if I now run this algorithm on a machine that is slower to perform variable look up and write?

Q. Should the complexity change?

Q. How important are those constants as the input size n gets large?

Q. How are our two results $11n^2 + 14n - 18$ and $11n^2 - 1$ similar?

Q. They are both quadratic polynomials.

how long do things take

We say...

- that a quadratic polynomial is of order n^2,
- that a cubic polynomial is of order n^3,
- that $4n\lg(n) + 2n + 10$ is of order $n\lg(n)$.

Why can we say this? With a little mathemagic:

$$11n^2 + 14n - 18 \leq 11n^2 + 14n \leq 11n^2 + 14n^2 = 25n^2$$

Another example:

$$11n^2 - 21n + 19 \leq 11n^2 + 19 \leq 11n^2 + n \leq 11n^2 + n^2 = 12n^2$$

for all natural $n \geq 19$

how long do things take — formally

For all natural $n \geq 19$:

$$11n^2 - 21n + 19 \leq 12n^2$$

There exists an $n_0 \in \mathbb{N}$ such that, for all natural $n \geq n_0$,

$$11n^2 - 21n + 19 \leq 12n^2$$

We can take this even further and say, there exists real $c > 0$ and natural n_0 such that, for all natural $n \geq n_0$,

$$11n^2 - 21n + 19 \leq c \cdot n^2$$

which is exactly the definition of "Big-Oh"!

Big-Oh — Asymptotic Upper Bound

We denote:

- $\mathbb{N}$: the set of natural numbers
- $\mathbb{R}^+$: the set of positive real numbers
- $\mathcal{F}$: the set of functions $f : \mathbb{N} \to \mathbb{R}^+$

Let $g \in \mathcal{F}$. Define $\mathcal{O}(g)$ to be the set of functions $f \in \mathcal{F}$ such that

$$\exists c \in \mathbb{R}^+, \exists n_0 \in \mathbb{N}, \forall n \in \mathbb{N}, n \geq n_0 \Rightarrow f(n) \leq c \cdot g(n)$$