CSC165 winter 2013

Mathematical expression

Danny Heap
heap@cs.toronto.edu
BA4270 (behind elevators)
http://www.cdf.toronto.edu/~heap/165/W13/
416-978-5899

Course notes, chapter 4

Outline

more asymptotics

notes

worst case

denote the worst-case complexity for program P with input $x \in I$, where the input size of x is n as $W_P(n) = \max\{t_P(x) \mid x \in I \wedge \text{size}(x) = n\}$

The upper bound $W_P \in \mathcal{O}(U)$ means

$$\exists c \in \mathbb{R}^+, \exists B \in \mathbb{N}, \forall n \in \mathbb{N}, n \geq B$$
$$\Rightarrow \max\{t_P(x) \mid x \in I \wedge \text{size}(x) = n\} \leq cU(n)$$

That is: $\exists c \in \mathbb{R}^+, \exists B \in \mathbb{N}, \forall x \in I, \text{size}(x) \geq B$

$$\Rightarrow t_P(x) \leq cU(\text{size}(x))$$

The lower bound $W_P \in \Omega(L)$ means

$$\exists c \in \mathbb{R}^+, \exists B \in \mathbb{N}, \forall n \in \mathbb{N}, n \geq B$$
$$\Rightarrow \max\{t_P(x) \mid x \in I \wedge \text{size}(x) = n\} \geq cL(n)$$

That is: $\exists c \in \mathbb{R}^+, \exists B \in \mathbb{N}, \forall n \in \mathbb{N}, n \geq B$

$$\Rightarrow \exists x \in I, \text{size}(x) = n \wedge t_P(x) \geq cL(n)$$

bounding a sort

```
    def IS(A) :
        """ IS(A) sorts the elements of A in non-decreasing or
1.      i = 1
2.      while i < len(A) :
3.          t = A[i]
4.          j = i
5.          while j > 0 and A[j-1] > t :
6.              A[j] = A[j-1] # shift up
7.              j = j-1
8.          A[j] = t
9.          i = i+1
```

$i = 1, \ldots, n-1$

$j = i, \ldots, 1$

5, 6, 7, $3 \cdot i$ "steps" + 1 for loop

I want to prove that $W_{\text{IS}} \in \mathcal{O}(n^2)$.

big-oh of n^2

We know, or have heard, that all quadratic functions grow at "roughly" the same speed. Here's how we make "roughly" explicit.

$$\mathcal{O}(n^2) = \{f : \mathbb{N} \mapsto \mathbb{R}^{\geq 0} \mid \exists c \in \mathbb{R}^+, \exists B \in \mathbb{N}, \forall n \in \mathbb{N}, n \geq B \Rightarrow f(n) \leq cn^2\}$$

Those are a lot of symbols to process. They say that $\mathcal{O}(n^2)$ is a set of functions that take natural numbers as input and produce non-negative real numbers as output. An additional property of these functions is that for each of them you can find a multiplier c, and a breakpoint B, so that if you go far enough to the right (beyond B) the function is bounded above by cn^2.

In terms of limits, this says that as n approaches infinity, $f(n)$ is no bigger than cn^2 (once you find the appropriate c).

prove $W_{\text{IS}} \in \mathcal{O}(n^2)$

prove $W_{IS} \in \Omega(n^2)$ $\exists c \in \mathbb{R}^+, \exists B \in \mathbb{N}, \forall n \in \mathbb{N}, n \geq B \Rightarrow \exists A, |A| = n, W_{IS}(A) \geq cn^2$

Pick $c =$ 1/2. Then $c \in \mathbb{R}^+$

Pick $B =$ 3. Then $B \in \mathbb{N}$

Assume $n \in \mathbb{N}$. # in order to intro $\forall$.

Assume $n \geq B$ # to intro $\Rightarrow$

Pick $A = [n, \ldots, 1]$ # $|A| = n$, sorted backwards.

lines 5, 6, 7 contribute $3i + 1$ steps for each i in $i = 1, 2, \ldots, n-1$, contributing ? steps. $\frac{(3n+2)(n-1)}{2}$

$\sum_{i=1}^{n-1} 3i+1$

can be written $3(\underline{1+2+\cdots+n-1}) + (n-1)$ $n-1$

$= 3\left(\frac{1+2+\cdots+n-1 \atop n-1+\cdots+2+1}{\frac{n+\cdots+n+n}{2}}\right) + (n-1) = \frac{3(n(n-1))}{2} + n-1$

$= \frac{3n^2 - 3n + 2n - 2}{2} =$

$= \frac{n^2 + (n^2 - 3n) + 2n + (n^2 - 2)}{2}$

$\geq \frac{n^2}{2}$ # since $n \geq B \geq 3$

$= cn^2$ # $c = 1/2$

prove $W_{\text{IS}} \in \Omega(n^2)$

Then $W_{IS}(A) \geq cn^2$

Then $\exists A, |A| = n, W_{IS}(A) \geq cn^2$ # intro $\exists$

Then $n \geq B \Rightarrow \exists A, |A| = n, W_{IS}(A) \geq cn^2$ # intro $\Rightarrow$

Then $\forall n \in \mathbb{N}$, # intro $\forall$

Then $\exists c \in \mathbb{R}^+, \exists B \in \mathbb{N},$ # intro # $\exists$ # 2x

maximum slice

```
def max_sum(L) :
  """maximum sum over slices of L"""
  max = 0
  i = 0
  while i < len(L) :
    j = i + 1
    while j <= len(L) :
      sum = 0
      k = i
      while k < j :
        sum = sum + L[k]
        k = k + 1
      if sum > max :
        max = sum
      j = j + 1
    i = i + 1
  return max
```

array pos

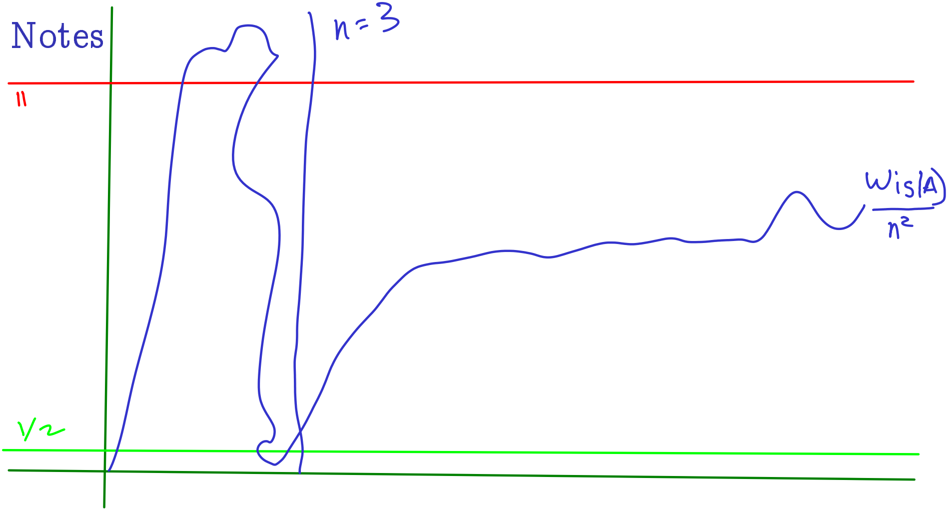
L = [-2, -5, -7]

[i : j]

$W_{ms}(n) \in \Theta(n^3)$

" " $\Omega(n^3)$

Notes



Notes