Order Of Growth 1

Tom Bohbot

September 2020

1 Introduction

The goal of this assignment is to understand how to prove runtimes and to understand the formal definitions of Big-O, Ω, and Θ.

NOTE 1: iff means if and only if.
NOTE 2: s.t. means such that.

2 Pick Your Constants: Big-O

2.1

Prove by selecting the appropriate constants and the definition used in lecture that f(n) is O(n^2)
Given: f(n) = 32n^2 + 17n + 1
Claim: f(n) is O(n^2)
Formal Definition of Big-O: T(n) = O(f(n)) iff there exists positive constants, c and n_0 s.t. T(n) $\leq$ c x f(n) for all n $\geq$ n $_0$

Proof:
f(n) has a O(n^2) runtime if, and only if, f(n) $\leq$ c*n^2.
1) 32n^2 + 17n + 1 $\leq c * n^2$
2) let c = 50 and n = 1.
3) 32n^2 + 17n + 1 $\leq 50 * n^2$, plug in 50 for c
4) $\frac{32n^2+17n+1}{n^2} \leq 21$
5)$\frac{32(1)^2+17(1)+1}{(1)^2}\leq 50$, plug in 1 for n
6) 50 $\leq$ 50
7)$Ifanynisinsertedlargerthanone,theresultwillbecomeincreasinglysmaller,andwillalwaysbelessthan21.Thi$
n_0 cannot disprove this, and that f(n) has a runtime of O(n^2).

2.2

Conversely: show, using counter-examples, that f(n)is neither O(n) nor O(nlogn)
Given: f(n) = 32n^2 + 17n + 1

General Claim: f(n) is not O(n) not O(nlogn)
Formal Definition of Big-O: T(n) = O(f(n)) iff there exists positive constants, c and n_0 s.t. T(n) $\leq$ c x f(n) for all n $\geq$ n $_0$
Claim 1: f(n) is not O(n).
Proof:
Prove by contradiction.
Assume f(n) is O(n).
If f(n) is O(n) then there is some equation that satisfies f(n) $\leq c * n$
1)$32n^2$ + 17n + 1 $\leq c * n$
2)$\frac{32n^2+17n+1}{n} \leq c$
3)$32n + 17 + \frac{1}{n} \leq$ c
4) Since the first term in the fraction was reduced to 32n, this can never be less than some constant factor. Since n can be any number multiplied by 32, it does not satisfy a number less than c. Therefore, this function cannot have a runtime of O(n).

Claim 2: f(n) is not O(nlogn).
Proof:
Prove by contradiction.
Assume f(n) is O(nlogn).
If f(n) is O(nlogn) then there is some equation that satisfies f(n) $\leq c * nlogn$
1)$32n^2$ + 17n + 1 $\leq$ c*nlogn
2) $\frac{32n^2+17n+1}{nlogn} \leq$ c
3) $\frac{32n}{logn} + \frac{17}{logn} + \frac{1}{nlogn} \leq$ c
4) Since the first term in the fraction was reduced to $\frac{32n}{logn}$, this can never be less than some constant factor. Since n can be any number multiplied by 32, it does not satisfy a number less than c. Therefore, this function cannot have a runtime of O(nlogn).

It has now been proved that f(n) is neither O(n) nor O(nlogn).

3 Pick Your Constants: Ω

3.1

Prove by selecting the appropriate constants and the definition used in lecture that f(n) is both $\Omega(n^2)$ and $\Omega(n)$.
Given: f(n) = $32n^2$ + 17n + 1
General Claim: f(n) is both $\Omega(n^2)$ and $\Omega(n)$
Formal Definition of Ω runtime: f(n) = Ω(g(n)) iff there exists positive constants c and n_0 s.t. f(n) $\geq$ c * g(n) for all n $\geq$ n_0
Proof: f(n) has an $\Omega(n^2)$ and $\Omega(n)$ runtime iff there exist positive consants c and n_0 s.t. T(n) $\geq$ c x f(n) for all n $\geq$ n_0.

Proving f(n) has an $\Omega(n^2)$ runtime: 1) f(n) $\geq$ c x n^2

2) $32n^2 + 17n + 1 \geq c \times n^2$
3) let c = 50 and n = 1
4) $32n^2 + 17n + 1 \geq 50 \times n^2$, plug in 50 for c
5) $32(n)^2 + 17(n) + 1 \frac{}{(n)^2} \geq 50$
6) 5) $32(1)^2 + 17(1) + 1 \frac{}{(1)^2} \leq 21$, plug in 1 for n
7) $50 \geq 50$
8) If any n is inserted larger than one, the result will become increasingly smaller, and will always be less than 50. This proves that any $n \geq n_0$ cannot disprove this, and that f(n) has a runtime of $\Omega(n^2)$.

Proving f(n) has an Ω(n) runtime:
1) $f(n) \geq c \times n$
2) $32n^2 + 17n + 1 \geq c \times n$
3) let c = 50 and n = 1
4) $32n^2 + 17n + 1 \geq 50 \times n$, plug in 50 for c
5) $32(n)^2 + 17(n) + 1 \frac{}{n} \geq 50$
6) 5) $32(1)^2 + 17(1) + 1 \frac{}{(1)} \leq 21$, plug in 1 for n
7) $50 \geq 50$
8) If any n is inserted larger than one, the result will become increasingly smaller, and will always be less than 50. This proves that any $n \geq n_0$ cannot disprove this, and that f(n) has a runtime of $\Omega(n)$.

Since f(n) has both a $\Omega(n)$ and $\Omega(n^2)$ runtime, the original claim above has now been proved. The original claim being that f(n) is both$\Omega(n)$ and $\Omega(n^2)$.

3.2

Conversely, show using counter-examples, that f(n) is not $\Omega(n^3)$.
Given: $f(n) = 32n^2 + 17n + 1$
Claim: f(n) is not $\Omega(n^3)$.
Formal Definition of Ω runtime: $f(n) = \Omega(g(n))$ iff there exists positive constants c and n_0 s.t. $f(n) \geq c * g(n)$ for all $n \geq n_0$
Proof:
Prove through contradiction.
Assume f(n) has a $\Omega(n^3)$ runtime.
If this is true then there is some equation that satisfies $f(n) \geq c * n^3$
1) $f(n) \geq c \times n^3$
2) $32n^2 + 17n + 1 \geq c \times n^3$
3) $\frac{32(n)^2+17(n)+1}{n^3} \geq c$
4) $\frac{32}{n} + \frac{17}{n^2} + \frac{1}{n^3} \geq c$
5) Since there are terms that have n as a denominator, f(n) can never be greater than some constant factor. Since n can be any number, namely infinity, it does not satisfy a number greater than c, as any number divided by infinity is zero. Therefore, this function cannot have a runtime of $\Omega(n^3)$.

4 Pick Your Constants: Θ

Claim: f(n) is $\Theta(n^2)$.
Formal Definition of Θ runtime: $f(n) = \Theta(g(n))$ iff there exist positive constants c_1, c_2, and n_0 s.t. c_1 x $g(n) \leq f(n) \leq c_2$ x $g(n)$ for all $n \geq n_0$
Given: $f(n) = 32n^2 + 17n + 1$
Proof:
Proving c_1 x $g(n) \leq f(n)$:
let $g(n) = n^2$, since that is our claim.
1) c_1 x $g(n) \leq f(n)$
2) c_1 x $n^2 \leq f(n)$
3) c_1 x $n^2 \leq 32n^2 + 17n + 1$
4) let c = 50 and n = 1
5) 50 x $(1)^2 \leq 32(1)^2 + 17(1) + 1$
6) $50 \leq 50$
7) So far we have determined that c_1 x $g(n) \leq f(n)$ since we have found some constant and n_0 that satisfy the above equation.

Proving $f(n) \leq c_1$ x $g(n)$:
let $g(n) = n^2$, since that is our claim.
1) $f(n) \leq c_1$ x $g(n)$
2) c_1 x $n^2 \geq f(n)$
3)c_1 x $n^2 \geq 32n^2 + 17n + 1$
4) let c = 50 and n = 1
5) 50 x $(1)^2 \geq 32(1)^2 + 17(1) + 1$
6) $50 \geq 50$
7) So far we have determined that c_1 x $g(n) \geq f(n) since we have found some constant and n_0$ that satisfy the above equation.

Since we have determined that both c_1 x $g(n) \leq f(n)$ and $f(n) \leq c_1$ x $g(n)$, this means that c_1 x $g(n) \leq f(n) \leq c_2$ x $g(n)$ is true which satisfies the original definition of Θ runtime, which means that this function is $\Theta(n^2)$.

4.1

Conversely, show using counter-examples, that f(n) is neither $\Theta(n)$ nor $\Theta(n^3)$.
Given: $f(n) = 32n^2 + 17n + 1$
General Claim: f(n) is neither $\Theta(n)$ nor $\Theta(n^3)$.
Formal Definition of Θ runtime: $f(n) = \Theta(g(n))$ iff there exist positive constants c_1, c_2, and n_0 s.t. c_1 x $g(n) \leq f(n) \leq c_2$ x $g(n)$ for all $n \geq n_0$
Proof:
Prove through contradiction:

Proving f(n) is not $\Theta(n)$.
Assume f(n) is $\Theta(n)$ s.t. c_1 x $g(n) \leq f(n) \leq c_2$ x $g(n)$ for all $n \geq n_0$
Proving $f(n) \leq c_2$ x $g(n)$: 1) $f(n) \leq c_2$*n, let g(n) be n as this is the claim.

2) $32n^2 + 17n + 1 \leq c_2$*n
3) $\frac{32n^2+17n+1}{n} \leq c_2$
4) 32n + 17 $\frac{1}{n} \leq c_2$
5) Since the first term is still in terms of n, this equation can never be true. If n were to be substituted with infinity it would surely be larger than c_2, meaning that it is impossible for f(n) to be Θ(n).

Proving f(n) is not $\Theta(n^3)$.
Assume f(n) is Θ(n) s.t. c_1 x g(n) $\leq$ f(n) $\leq$ c_2 x g(n) for all n $\geq$ n_0
Proving f(n) $\geq$ c_2 x g(n): 1) f(n) $\geq c_2$*n^3, let g(n) be n^3 as this is the claim.
2) $32n^2 + 17n + 1 \geq c_2$*n^3
3) $\frac{32n^2+17n+1}{n^3} \geq c_2$
4) $\frac{32}{n} + \frac{7}{n^2} + \frac{1}{n^3} \geq c_2$
5) Since the first term, and the other terms, are still in terms of n, this equation can never be true. If n were to be substituted with infinity it would surely be smaller than c_2. It would be smaller because anything divided by infinity is zero. This means that it is impossible for f(n) to be $\Theta(n^3)$.