NUS | Computing
National University of Singapore

Eric Han
eric_han@nus.edu.sg
https://eric-han.com
Computer Science

T03 – Week 4

Assignment 1 and 2 review

CS3230 – Design and Analysis of Algorithms

Admin

Assignment 1 scores with comments are published, can be found on Canvas.

- Comments have been given
- Any queries please approach me (after class or on telegram)

Will be marking Assignment 2 soon (by $+1$ week).

General comments

- Some proof is too succinct.
  - When in doubt, give at least a one line reasoning
- Glad to see many use limits to proof, which makes some easy!

Recap from T02

Practice, practice, practice...

Q1 Telescoping Series

Review and do more practice:

- Khan Academy Telescoping
- Libretext Telescoping

Q5 Substitution Method

Review and do more practice (search 'big o substitution method worksheet'):

- Brilliant Substitution Method
- The University of Auckland
- MIT PS1

Substitution Method can be very powerful: $T(n) = T(n - \sqrt{n}) + \sqrt{n}$; Intuition:

Recap from T02

Practice, practice, practice...

Q1 Telescoping Series

Review and do more practice:

- Khan Academy Telescoping
- Libretext Telescoping

Q5 Substitution Method

Review and do more practice (search 'big o substitution method worksheet'):

- Brilliant Substitution Method
- The University of Auckland
- MIT PS1

Substitution Method can be very powerful: $T(n) = T(n - \sqrt{n}) + \sqrt{n}$; Intuition:

- $\sqrt{n}$ is the minimum work needed to be done
- Try out a few steps to see if any pattern emerges.
- What can be substituted to get back the same thing?

Assignment 1 Q1a [P1]

Consider the following eight functions of n:

$$2^{3n}, \quad 3^{2n}, \quad n^{17}, \quad n^{17} - n^{16},$$

$$8^{\log_2 n}, \quad \log_{10} 2^{(n^{18})}, \quad n!, \quad \sqrt{n}$$

Order the above functions on the basis of nondecreasing order from smallest to largest, where $f(n)$ is considered smaller than $g(n)$ if $f(n) \in O(g(n))$ but $g(n) \notin O(f(n))$. If $f(n) \in \Theta(g(n))$, then either can come earlier in the order. Give proof/arguments on why your order is correct.

Answer

Simplified Functions and Their Growth

Function	Growth Type
$2^{3n} = 8^n$	Exponential
$3^{2n} = 9^n$	Exponential
n^{17}	Polynomial
$n^{17} - n^{16}$	Polynomial, $\forall n \geq 2 : 0.5n^{17} \leq n^{17} - n^{16} \leq n^{17} \implies \Theta(n^{17})$
$8^{\log_2 n} = n^3$	Polynomial
$\log_{10} 2^{(n^{18})} = \frac{n^{18}}{\log 10}$	Polynomial, $\forall n : \frac{1}{\log 10} n^{18} \leq \frac{n^{18}}{\log 10} \leq \frac{1}{\log 10} n^{18} \implies \Theta(n^{18})$
$n!$	Exponential
$\sqrt{n} = n^{0.5}$	Polynomial

Polynomial Growth

$$
\begin{aligned}
0 < i < j &\implies (\forall n \geq 2)\; n^i < n^j, \\
&\implies (\forall n \geq 2)\; n^{0.5} < n^3 < n^{17} < n^{18}.
\end{aligned}
$$

Hence, we have the current ordering:

$$\sqrt{n} \leq 8^{\log_2 n} \leq (n^{17} = n^{17} - n^{16}) \leq \log_{10} 2^{(n^{18})}$$

Exponential Growth

$(\forall n \geq 1)\ 8^n < 9^n$, and we have the ordering $8^n < 9^n$.

Comparing 9^n vs $n!$

Let $m = 2 \cdot 9^2$, so

$$
\begin{aligned}
m! &= 1 \cdot 2 \cdot \ldots \cdot m \\
&\geq \left(\frac{m}{2} + 1\right) \cdot \left(\frac{m}{2} + 2\right) \cdot \ldots \cdot m && \text{(Keep largest } \frac{m}{2} \text{ terms)} \\
&\geq \underbrace{(9^2) \cdot (9^2) \cdot \ldots \cdot (9^2)}_{m/2 = 9^2 \text{ terms}} && \text{(Each term is at least } 9^2\text{)} \\
&= (9^2)^{9^2} \\
&= 9^{2 \cdot 9^2} = 9^m. && \text{(Simplify)}
\end{aligned}
$$

Hence, $(\forall n \geq m)\ n! \geq 9^n$; Putting it all together:

$$8^n \leq 9^n \leq n!$$

Comparing Polynomial vs Exponential

Note that for

- Large enough k (ie. $\frac{1}{2^{\frac{1}{36}}-1}$), $\implies 1+\frac{1}{k} \leq 2^{1/36}$.
- $n \geq 35k \implies 35k-1 \leq n \implies 35k+n-1 \leq 2n \implies \frac{35k+n-1}{2} \leq n$.

$$n^{18} = \left[\underbrace{\frac{2}{1} \cdot \frac{3}{2} \cdot \ldots \cdot \frac{k+1}{k}}_{\text{First } k \text{ terms}} \cdot \underbrace{\frac{k+2}{k+1} \cdot \frac{k+3}{k+2} \cdot \ldots \cdot \frac{n}{n-1}}_{\text{Remaining } n-k-1 \text{ terms}} \right]^{18}$$

$$\leq \left[2^k \cdot 2^{(n-k-1)/36}\right]^{18} \qquad (1+\frac{1}{k} \leq 2^{1/36})$$

$$= \left[2^{(35k+n-1)/36}\right]^{18} = 2^{(35k+n-1)/2} \leq 2^n \qquad (\frac{35k+n-1}{2} \leq n).$$

Therefore:

$$\sqrt{n} \leq 8^{\log_2 n} \leq (n^{17} = n^{17} - n^{16}) \leq \log_{10} 2^{(n^{18})} \leq 8^n \leq 9^n \leq n!$$

Assignment 1 Q1b

Show that $\sum_{i=1}^{n} \frac{1}{i} \in \Theta(\ln n)$.

Answer

Note that:

- $\int_a^b \frac{1}{x}\,dx = \ln(x)\Big|_a^b = \ln(b) - \ln(a) = \ln\left(\frac{b}{a}\right)$
- For $i \geq 1$, also[1]: $\frac{1}{i+1} \leq \int_{x=i}^{i+1} \frac{1}{x}\,dx \leq \frac{1}{i} \implies \sum_{i=2}^{n} \frac{1}{i} = \sum_{i=1}^{n-1} \frac{1}{i+1} \leq \int_{x=1}^{n} \frac{1}{x}\,dx$

$$
\begin{aligned}
\ln n \leq \ln(n+1) = \ln(n+1) - \ln 1 &= \int_{x=1}^{n+1} \frac{1}{x} dx \\
&\leq \sum_{i=1}^{n} \frac{1}{i} && \left(\int_m^{n+1} f(x)\,dx \leq \sum_{k=m}^{n} f(k)\right) \\
&\leq 1 + \int_{x=1}^{n} \frac{dx}{x} = 1 + \ln n - \ln 1 = 1 + \ln n. && \left(\sum_{i=1}^{n} \frac{1}{i} \leq 1 + \int_{x=1}^{n} \frac{1}{x}\,dx\right)
\end{aligned}
$$

Thus $\sum_{i=1}^{n} \frac{1}{i} \in \Theta(\ln n) = \Theta(\log n)$.

---

[1] See Riemann Sum.

Assignment 1 Q2

Recall that the Fibonacci numbers are defined as follows: $F_0 = 0$, $F_1 = 1$, and for $n \geq 2$, $F_n = F_{n-1} + F_{n-2}$.

Prove that every non-negative integer m can be expressed as a sum of a finite set of Fibonacci numbers, no two of which are the same or consecutive. That is, every non-negative integer m can be written as $m = F_{i_1} + F_{i_2} + ... + F_{i_k}$, where

a. $i_1 < i_2 < \cdots < i_k$, and
b. for $1 \leq j < k$, $i_j + 1 < i_{j+1}$.

Answer

We prove the claim by induction on m.

Base Case: $m = 0$ and $m = 1$ hold as $m = F_0$ and $m = F_1$.

Induction step: Assume for all m with $0 \leq m < n$, m can be written as claimed.

Let r be the largest $F_r \leq n$:

- Case $F_r = n$: done.
- Case $F_r < n$: Let $n' = n - F_r$
  - Assume $n' \geq F_{r-1} \implies n = F_r + n' \geq F_r + F_{r-1} = F_{r+1} \implies$ contradiction.
  - So $n' < F_{r-1}$.
  - By induction $n' = F_{i_1} + ... + F_{i_k}$, where $i_1, \dots, i_k$ satisfy (a) and (b).
  - Notice largest term F_{i_k} is $F_{i_k} \leq n' < F_{r-1} \implies i_k < r - 1$.
  - Now, $n = F_{i_1} + ... + F_{i_k} + F_{i_{k+1}}$, where $i_{k+1} = r$.
  - Since $i_k < r - 1 \implies i_k + 1 < i_{k+1}$, done.

Thus, n can be expressed as required.

Assignment 1 Q3

Suppose that $f(n), g(n) : \mathbb{Z}_{\geq 0} \to \mathbb{Z}_{\geq 0}$ are increasing functions such that $f(n) \in O(g(n))$. Must it be true that $2^{f(n)} \in O(2^{g(n)})$?

Answer

No.

- For example, suppose $f(n) = 2n$ and $g(n) = n$.
- Then $f(n) \leq 2g(n) \in O(g(n))$.
- However, $2^{f(n)} = 2^{2n} = 4^n = 2^n \cdot 2^n$:
  - $(\forall c, n_0 > 0)(\forall n > \max(2 + c, n_0))\ 2^n \cdot 2^n > c \cdot 2^n$
  - $2^{2n} \notin O(2^n)$

Assignment 2 Q1 [P2]

Solve the following recurrence relations. Provide as tight a bound as possible. If you use the master theorem, state the case that applies, along with a short reasoning why the case applies. Otherwise, give a detailed proof (using any of the methods). Unless otherwise specified, you can assume base cases, $T(n)$ for $n \leq$ some constant, to be $\Theta(1)$. For ease of notation, floors/ceilings are omitted (as mentioned in class, asymptotically, they don't make much difference for the following questions). Thus, when using a term such as $6n/7$ below, assume it means $\lfloor \frac{6n}{7} \rfloor$.

a. $T(n) = 6T(n/3) + n^2$.
b. $T(n) = 9T(n/2) + 6n^3 + 4$.
c. $T(n) = T(n/7) + T(6n/7) + 5$. (Assume $7 \leq T(n) \leq 100$ for $n \leq 7$.)

Answer 1a

Using the **Master Theorem**, Case 3:

- $a = 6$, $b = 3$, $f(n) = n^2$
- $d = \log_3 6 < 1.7 = 2 - 0.3$ (thus, $\epsilon = 0.2$), and $f(n) \in \Omega\left(n^{d+\epsilon}\right)$.
- Furthermore, the **regularity condition** is satisfied, as:

$$6f(n/3) = 6(n/3)^2 = \frac{6n^2}{9} \leq \frac{6}{9}f(n) \implies c = \frac{6}{9} < 1$$

Thus, $T(n) \in \Theta(n^2)$.

Answer 1b

Using the **Master Theorem**, Case 1:

- $a = 9$, $b = 2$, $f(n) = 6n^3 + 4$
- $d = \log_2 9 > 3.1 = 3 + 0.1$ (thus, $\epsilon = 0.05$), and $f(n) \in O(n^{d-\epsilon})$.

Thus, $T(n) \in \Theta(n^{\log_2 9})$.

Answer 1c

Upper bound

Guess that $T(n) \leq 105n - 5$ for $n \geq 1$.
Base Cases for $1 \leq n \leq 7$:

$$T(n) \leq 100 \leq 105n - 5 \quad \text{for } 1 \leq n \leq 7.$$

Induction Step for $n > 7$:
Assume that the guess holds for $1 \leq n < m$, and prove for $n = m$.

$$T(m) \leq T(\lfloor m/7 \rfloor) + T(\lfloor 6m/7 \rfloor) + 5 \leq 105(m/7) - 5 + 105(6m/7) - 5 + 5 \leq 105m - 5.$$

In fact, it would work for all $T(n) \leq a \cdot n - 5$.

Lower bound
Guess that $T(n) \geq n$.
Base Cases for $1 \leq n \leq 7$:

$$T(n) \geq 7 \geq n \quad \text{for } 1 \leq n \leq 7.$$

Induction Step for $n > 7$:
Assume that the guess holds for $1 \leq n < m$, and prove for $n = m$.

$$T(m) \geq T(\lfloor m/7 \rfloor) + T(\lfloor 6m/7 \rfloor) + 5 \geq m/7 - 1 + 6m/7 - 1 + 5 \geq m + 3 \geq m.$$

Conclusion
From Upper and Lower bounds, it follows that $T(n) \in \Theta(n)$.

Assignment 2 Q2

Suppose that $f(1) = 0$ and $f(n) = 3f(n/3) + n$ when n is a power of 3. Show that $f(n) = n \log_3 n$ when n is a power of 3.

Answer

Base Case:

For $n = 1 = 3^0$,

$$f(n) = 0 = 3^0 \cdot \log_3 3^0.$$

Induction Step:

Suppose the hypothesis holds for $n = 3^m$.
Then, we show it for $n = 3^{m+1}$.

$$\begin{aligned} f(n) &= 3f(n/3) + n \\ &= 3f(3^m) + n \\ &= 3 \cdot 3^m \log_3 3^m + 3^{m+1} \\ &= 3^{m+1}(m+1) \\ &= (3^{m+1}) \log_3(3^{m+1}). \end{aligned}$$

Assignment 2 Q3 [P3]

Recall that the *greatest common divisor (GCD)* of two non-negative integers m, n is the greatest positive integer p such that p divides both m and n.

Algorithm 1: $GCD(m, n)$

if $m = 0$ **then**
  return n
else
  return $GCD(n \mod m, m)$;

Give as tight an upper bound as possible on the running time of the above algorithm in terms of n, the larger of the two inputs. You may assume that computing "$n \mod m$" takes constant time for this question.

Answer

Key Observation

- If $m \leq n/2$, then:

$$n \mod m < m \leq n/2.$$

- If $m > n/2$, then:

$$n \mod m \leq n - m < n/2.$$

Thus, in both cases, after two iterations, the maximum of the two values (on which GCD is taken) reduces by at least half.

Upper Bound

We can express the recurrence for the running time as:

$$T(n) \leq T(\lfloor n/2 \rfloor) + 2c,$$

where c is the bound on the runtime of one iteration. Solving this recurrence gives:

$$T(n) \in O(\log n).$$

Fibonacci Case's Lower Bound

Consider $GCD(F_m, F_{m+1})$, where F_i is the i-th Fibonacci number:

- In one iteration, the input numbers F_m, F_{m+1} become F_{m-1}, F_m.
- This requires $\Omega(m)$ steps.

Additionally, since $\log F_m \in \Theta(m)$ (recall[2] $F_m \geq 2^{m/2}$ and $F_m \leq 2^m$):

- For $n = F_m$, Euclid's algorithm takes time:

$$T(n) \in \Omega(\log n).$$

Conclusion

Thus, the running time of Euclid's algorithm is upper bounded by $T(n) \in O(\log n)$.

[2] Or Golden Ratio.