CS5800: Algorithms — Virgil Pavlu

Homework 1

Name: Mumuksha Pant
Collaborators:

Instructions:

- Make sure to put your name on the first page. If you are using the LaTeX template we provided, then you can make sure it appears by filling in the `yourname` command.
- Please review the grading policy outlined in the course information page.
- You must also write down with whom you worked on the assignment. If this changes from problem to problem, then you should write down this information separately with each problem.
- Problem numbers (like Exercise 3.1-1) are corresponding to CLRS 3^{rd} edition. While the 2^{nd} edition has similar problems with similar numbers, the actual exercises and their solutions are different, so make sure you are using the 3^{rd} edition.

IF curr1.data == curr2.data => *Return curr1.data*
curr1=curr1.next , curr2=curr2.next

RETURN count
end

(b) Write the actual code in a programming language (C/C++, Java, Python etc) of your choice and run it on a made-up test pair of two lists. A good idea is to use pointers to represent the list linkage.

Solution:
Reference taken from Geeks for Geeks

---

2. (10 points) Exercise 3.1-1

Solution:

We will prove that function $f(n) => max(f(n), g(n)$ lies within upper and lower bound conditions. If it satisfies upper and lower bounds then it will satisfy θ

Part I - Prove Lower Bound (ω)
max(f(n) , g(n)) ≥ f(n) –assuming maximum is g(n)
max(f(n) , g(n))≥ g(n) – assuming maximum is f(n)

summing the 2 statements above -
2 (max (f(n) , g(n)) ≥ f(n) + g(n)

(max (f(n) , g(n)) ≥ $\frac{1}{2}$ (f(n) + g(n)) -1

where f(n) represents (max (f(n) , g(n))
c1 represents $\frac{1}{2}$
g(n) represents (f(n) + g(n))

Part II - Prove Upper Bound (O)
f(n) + g(n) ≤ f(n) –assuming f(n) is greater
f(n) + g(n) ≤ g(n) –assuming g(n) is greater

Taking max of f(n) and g(n) and combining the above 2 statements -
(max (f(n) , g(n)) ≤ 1* (f(n) + g(n)) -2
where f(n) represents (max (f(n) , g(n))
c2 represents 1
g(n) represents (f(n) + g(n))
combining 1 and 2

1. **(20 points)**

Two linked lists (simple link, not double link) heads are given: headA, and head B;it is also given that the two lists intersect, thus after the intersection they have the same elements to the end. Find the first common element, without modifying the lists elements or using additional datastructures.

(a) A linear algorithm is discussed in the lecture: count the lists first, then use the count difference as an offset in the longer list, before traversing the lists together. Write a formal pseudocode (the pseudocode in the lecture is vague), using "next" as a method/pointer to advance to the next element in a list.

Solution:

Algorithm Steps
1. Count nodes in First Linked List (length of L1)
2. Count nodes in Second Linked List (length of L2)
3. Find the difference difference = abs(L1-L2)
4. Traverse the bigger list from first node to difference (L1-L2)

Pseudocode :
begin

FUNCTION getCount() to calculate the count of linked lists
WHILE curr !=NULL
count++
curr = curr.next

FUNCTION getNode() to get intersection point of linked lists

declare var **length**

calculate length of linked lists

$l1 = getCount(head1);$

$l2 = getCount(head2);$

calculate the absolute difference between lengths of linked lists

length = l1-l2 ;

FUNCTION intersect() to find intersection of linked lists
WHILE curr1 != null **AND** curr2 != null

$(max(f(n), g(n)) = \theta((f(n) + g(n))$

---

3. (5 points) Exercise 3.1-4

Solution: **Is** $2^{n+1} = O(2^n)?2^{2n} = O(2^n)?$
Part 1 -

YES.
By the definition of Big O
$f(n) \leq D.(g(n))$ where D is any constant and f(n) and g(n) are 2 functions

If we take D = 2 , and apply the above logic -
$2^{n+1} = 2^n * 2$
$f(n) \leq D.(g(n))$

$f(n) = O(g(n))$

Part 2 : $2^{2n} = O(2^n)?$

NO.
By the definition of Big O
$f(n) \leq D.(g(n))$ where D is any constant and f(n) and g(n) are 2 functions

$2^{2n} = 2^n * 2^n$
there is no constant to make it Big O .

---

4. (15 points)

Rank the following functions in terms of asymptotic growth. In other words, find an arrangement of the functions $f_1, f_2, ...$ such that for all i, $f_i = \Omega(f_{i+1})$.

$$\sqrt{n}\ln n \quad \ln\ln n^2 \quad 2^{\ln^2 n} \quad n! \quad n^{0.001} \quad 2^{2\ln n} \quad (\ln n)!$$

Solution: We assume $n = 10$ AND ln=$\log_2$ for all calculations
1) $\sqrt{n}\ln n$
$= \sqrt{10}log10$
$= 3.16 * 3.32$
≈ 9.6

2) $\ln\ln n^2$
$= \log\log(10^2)$
$= \log(2 * \log 10)$
$= \log(2 * 3.32)$

$= \log(6.6)$
$= 0.82$

3) $2^{\ln^2 n}$
polylograthimic function $2^{\ln^c}$
polynomial grows faster than a polylograthimic function

4) $n! = 10! = 10 * 9 * 8 * 7 * 6 * 5 * 4 * 3 * 2 * 1$

5) $n^{0.001}$
$= 10^{0.001}$
$= 1.002$

6) $2^{2 \ln n}$
$= 2^{2 \log 10}$
$= 2^{2*3.62}$
$= 2^6.64$
≈ 90

7) $(\ln n)!$
$(\log 10)!$
$= (3.62)! \approx 3 * 2$

asymptotically we have $factorial(x) > exp(x) > polynomial(x) > polylograthimic > log(x)$

Hence-
$\ln \ln n^2 \leq 2^{\ln^2 n} \leq n^{0.001} \leq (\ln n)! \leq \sqrt{n} \ln n \leq 2^{\ln^2 n} \leq n!$

---

5. (40 points) Problem 4-1 (page 107)

Give asymptotic upper and lower bounds for $T(n)$ in each of the following recurrences. Assume that $T(n)$ is constant for $n \leq 2$. Make your bounds as tight as possible, and justify your answers.

(a) $T(n) = 2T(n/2) + n^4$

Solution:

Using master's theorem -T(n) = aT(n/b) + f(n)where $f(n) = \theta n^k * \log^p(n)$

T(n) has the following asymptotic bounds:

case 1 If $a > b^k$, then T(n) = (nlog^b a)

case 2 If $a = b^k$,
1) $p > -1$, then T(n) =$\theta(n^{\log_b a} \log^{p+1} n)$
2) $p = -1$, then T(n) = $\theta(n^{\log_b a} \log \log n)$
3) $p < -1$, then T(n) = $\theta(n^{\log_b a})$

$\boxed{\textbf{case 3} \text{ If } a < b^k,}$

1) $p \geq 0$, then T(n) = $\theta(n^k . \log^p n)$

2) $p < 0$, then T(n) = $\theta(n^k)$ From the equation we have values -

a=2,
b= 2,
k =4,
p= 0
using case 3(1) of Masters Theorem
T(n) = $\theta n^k log^p n$

$\boxed{\text{T(n)} = \theta n^4}$

(b) $T(n) = T(7n/10) + n$

Solution:
From the equation we have values -
a=1,
b= $\frac{10}{7}$,
k =1,
p= 0
using case 3(1) of Masters Theorem
T(n) = $\theta(n^k log^p n)$

$\boxed{\text{T(n)} = \theta(n)}$

(c) $T(n) = 16T(n/4) + n^2$

Solution:
From the equation we have values -
a=16,
b= 4,
k =2,
p= 0
using case 2(1) of Masters Theorem
T(n) =$\theta(n^{\log_b a} \log^{p+1} n)$
T(n)= $\theta(n^{2 \log_4 4} \log^{0+1} n)$

$\boxed{\text{T(n)} = \theta(n^2 \log n)}$

(d) $T(n) = 7T(n/3) + n^2$

Solution:
From the equation we have values -
a=7,
b= 3,
k =2,
p= 0

using case 3(1) of Masters Theorem
T(n) = $\theta(n^{\log_b a})$
T(n)= $\theta(n^k \log^p n)$
$\boxed{\text{T(n)= } \theta(n^2}$

(e) $T(n) = 7T(n/2) + n^2$

Solution:
From the equation we have values -
a=7,
b= 2,
k =2,
p= 0
using case 1 of Masters Theorem
T(n)= $\theta(n^{log_b a})$
$\boxed{\text{T(n)= } \theta(n^{log_2 7})}$

(f) $T(n) = 2T(n/4) + \sqrt{n}$

Solution:
From the equation we have values -
a=2,
b= 4,
k =$\frac{1}{2}$,
p= 0
using case 2(1) of Masters Theorem
T(n) = $\theta(n^{\log_b a} \log \log n)$
T(n)= $\theta(n^{log_4 2} \log n)$
T(n)= $\theta(n^{\frac{1}{2}} \log n$
$\boxed{\text{T(n)= } \theta(\sqrt{n} \log n}$

(g) $T(n) = T(n-2) + n^2$

Solution:
using substitution
T(n)= $T(n-2-2) + (n-2)^2 + n^2$
T(n)= $T(n-2-2-2) + (n-4)^2 + (n-2)^2 + n^2$ T(n) =$T(n-2k) + (n-2k)^2$
T(n) = $T(n-2k) + \sum_{i=1}^{k-1} (n-2k)^2$
finding the last k : T(0) = $n - 2k = 0$
n=2k
$\frac{n}{2} = k$
T(n)= $T(0) + \sum_{i=1}^{k-1} (n-2k)^2$
putting value of k= n/2 in the equation
T(n)= $T(0) + \sum_{i=1}^{\frac{n}{2}-1} (n-2k)^2$
using formula $\frac{n(n+1)(2n+1)}{6}$
T(n)= $T(0) + \sum_{i=1}^{\frac{n}{2}-1} \frac{n(n+1)(2n+1)}{6}$

T(n)= $T(0) + \frac{(\frac{n}{2}-1)*\frac{n}{2}*(n-1)}{6}$
T(n)= $\frac{n^3}{4} - \frac{3n^2}{4} + \frac{n}{2}$
$\boxed{\text{T(n)= } \theta(n^3)}$

---

6. (30 points) Problem 4-3 from (a) to (f) (page 108)

Give asymptotic upper and lower bounds for $T(n)$ in each of the following recurrences. Assume that $T(n)$ is constant for sufficiently small n. Make your bounds as tight as possible, and justify your answers.

(a) $T(n) = 4T(n/3) + n \lg n$

Solution:
From the equation we have values -
a=4,
b= 3,
k =1,
p= 1
using case 1 of Masters Theorem

T(n) = $\theta(n^{\log_b a})$
$\boxed{\text{T(n)= } \theta(n^{log_3 4})}$

(b) $T(n) = 3T(n/3) + n/\lg n$

Solution:
using substitution
T(n)= $3[3.T\frac{n}{3^2} + \frac{\frac{n}{3}}{\log\frac{n}{3}}] + \frac{n}{\log n}$ .... k=1
T(n) = $3*3[3T[\frac{n}{3^3}] + \frac{n}{\log\frac{n}{3^2}} + \frac{n}{\log\frac{n}{3}} + \frac{n}{\log n}]$ ....k=2
T(n) = $3^k.T[\frac{n}{3^k}] + n[\frac{1}{\log n} + \frac{1}{\log\frac{n}{3}} + \frac{1}{\log\frac{n}{3^2}} + ...]$ ....k
T(n) = $3^k.T[\frac{n}{3^k}] + n\sum_{i=1}^{k-1}\frac{1}{(\log\frac{n}{3^k})}$ –(eq 1)

last k : T(1)=1
$\frac{n}{3^k} = 1$
taking log both sides : $n\log n = k.\log 3$
assuming log base 3
$\boxed{\log n = k}$
Also , $\boxed{k = 3^k}$ putting this value in (1)
and using formula $\frac{\log a}{\log b} = \log a - \log b$

T(n) = $3^k.T(1) + n\sum_{i=1}^{logn-1}(\frac{1}{\log n - \log 3^k})$
Put $3^k = k$
T(n) = $3^k + n\sum_{i=1}^{\log n-1}\frac{1}{\log n - \log k}$

T(n) = $3^k + n\sum_{i=1}^{\log n-1}(\frac{1}{\log n - \log n+1}) + (\frac{1}{1+1} + (\frac{1}{1+1+1}) + ...$
T(n) = $3^k + n(1 + \frac{1}{2} + \frac{1}{3} + ... + \frac{1}{\log n})$
using formula of harmonic functions: $1 + \frac{1}{2} + \frac{1}{3} + ... + \frac{1}{n} = \log n$

T(n) = $3^k + n\log(\log n)$
$\boxed{\text{T(n)} = \theta(n\log(\log n))}$

(c) $T(n) = 4T(n/2) + n^2\sqrt{n}$

Solution:
From the equation we have values -
a=4,
b= 2,
k =$\frac{5}{2}$,
p= 0
using case 3 of Masters Theorem

T(n) = $\theta(n^k \log^p n)$
$\boxed{\text{T(n)}= \theta(n^{\frac{5}{2}})}$

(d) $T(n) = 3T(n/3-2) + n/2$

Solution:
Collaborator - Dhruv Saini

We need to modify T(n) to get rid of the -2 in the recurrence. We will then use Masters theorem to get the solution and prove the upper bound and lower bound separately.

T(n) = $3T(n/3-2) + n/2$
using masters theorem , case 2(1) -
a=3 , b=3 , k=1 , p =0
$\theta(n^{\log_3 3} * \log n)$
$\boxed{\theta(n\log n)}$

Proving upper and lower bounds -
$C.f(n) \leq T(n) \leq D.f(n)$
$C.n\log n \leq T(n) \leq D.n\log n$

For lower bound
$C * nlogn \leq T(n)$

Induction Step :
$C\frac{n}{3}log(\frac{n}{3}) \leq T(\frac{n}{3})$ want => $Cnlogn \ \leq T(n)$ ....(eq1)

$T(n) = 3T(n/3) + n/2 \ \leq 3C(\frac{n}{3}log(\frac{n}{3})) + \frac{n}{2}$

$T(n) = nClog(\frac{n}{3}) + \frac{n}{2}$ ......(eq2)
using (1) and (2)
$C.n\log(\frac{n}{3}) + \frac{n}{2} \geq \ Cnlogn$
$C.log_3 \geq \frac{1}{2}$
Therefore for all $C \geq \frac{1}{2log3}$ lower bound is valid.

for upper bound we can take value of D = n^2
$\frac{1}{2log3}f(n) \leq n\log n \leq n^2.f(n)$

Hence $T(n) = 3T(n/3) + n/2 = \theta(nlogn)$

(e) $T(n) = 2T(n/2) + n/\lg n$

Solution:
using substitution
T(n)= $2.2[T\frac{n}{2^2} + \ \frac{\frac{n}{2}}{\log\frac{n}{2}}\ ] + \frac{n}{\log n}$ ..k=1
T(n) = $2*2[2T[\frac{n}{2^2}] + \frac{n}{\log\frac{n}{2^2}} + \frac{n}{\log\frac{n}{2}} + \frac{n}{\log n}]$ ..k=2
T(n) = $2^k.T[\frac{n}{2^k}] + n[\frac{1}{\log n} + \frac{1}{\log\frac{n}{2}} + \frac{1}{\log\frac{n}{2^2}} + ... \ \ ]$ ..k
T(n) = $2^k.T[\frac{n}{2^k}] + n\sum_{i=1}^{k-1}\frac{1}{(\log\frac{n}{2^k})}$ –(eq 1) last k : T(1)=1
$\frac{n}{2^k} = 1$
taking log both sides : $n\log n = k.\log 2$
assuming log base 2
$\boxed{\log n = k}$
Also , $\boxed{k = 2^k}$ putting this value in (1)
and using formula $\frac{\log a}{\log b} = \log a - \log b$

T(n) = $2^k.T(1) + n\sum_{i=1}^{logn-1}(\frac{1}{\log n - \log 2^k})$
Put $2^k = k$
T(n) = $2^k + n\sum_{i=1}^{\log n-1}\frac{1}{\log n - \log k}$ T(n) = $2^k + n\sum_{i=1}^{\log n-1}(\log\frac{1}{n-\log n+1}) + (\frac{1}{1+1} + (\frac{1}{1+1+1}) + ...$
T(n) = $2^k + n(1 + \frac{1}{2} + \frac{1}{3} + ... + \frac{1}{\log n})$
using formula of harmonic functions: $1 + \frac{1}{2} + \frac{1}{3} + ... + \frac{1}{n} = \log n$
T(n) = $n + n\log(\log n)$
T(n) = $\theta(n\log(\log n))$

(f) $T(n) = T(n/2) + T(n/4) + T(n/8) + n$

Solution:
using recursion tree-

$T(n) \quad + n \quad$ ....level 1

$T(\frac{n}{2}) + T(\frac{n}{4}) + T(\frac{n}{8}) \quad + [n[\frac{1}{2} + \frac{1}{4} + \frac{1}{8}] => (\frac{7}{8})n]$ ...level 2

$T(\frac{n}{4}) + 2T(\frac{n}{8}) + 3T(\frac{n}{16}) + 2T(\frac{n}{32}) + T(\frac{n}{64}) \quad + [n[\frac{1}{4} + 2(\frac{1}{8}) + 3(\frac{1}{16}) + 2(\frac{1}{32}) + (\frac{1}{64})] => ((\frac{7}{8})^2)n]$ ...level 3

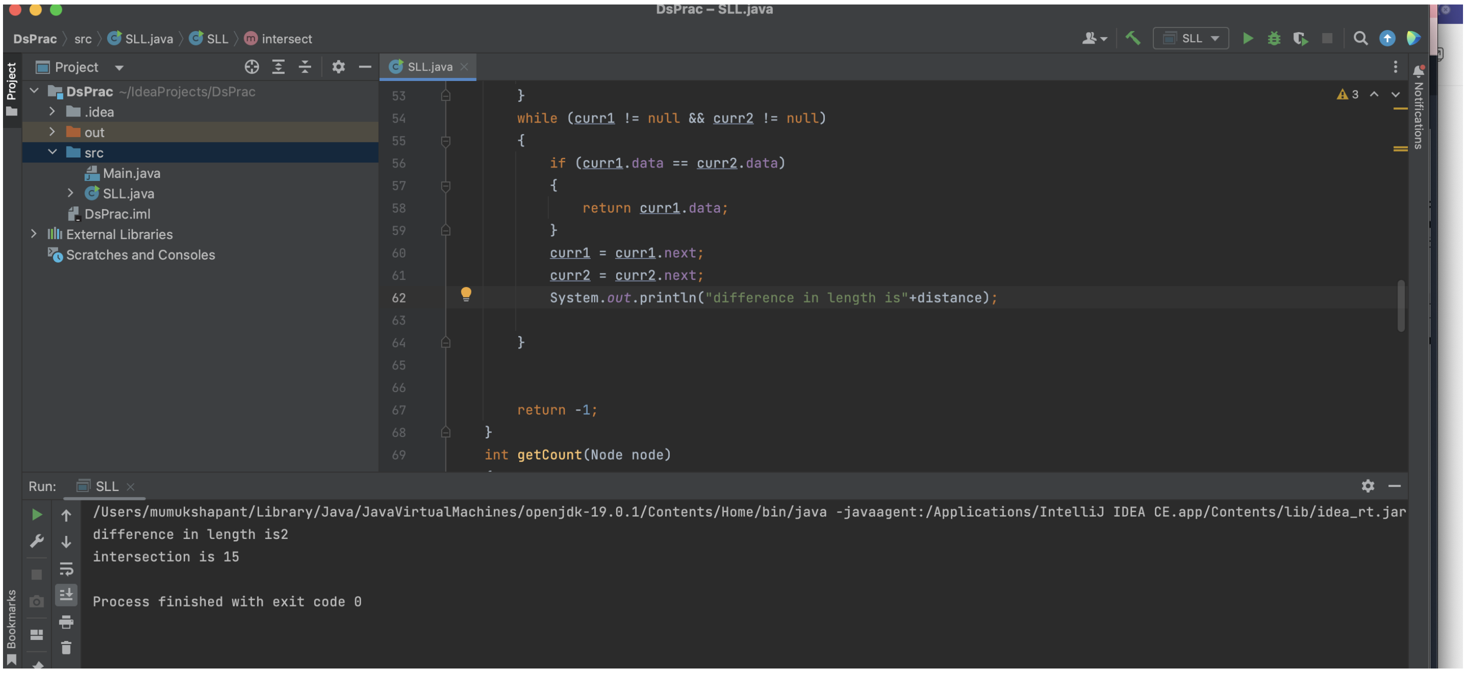
For last k:
total terms = 3^k
largest term = $T(\frac{n}{2^k})$
smallest term = $T(\frac{n}{8^k})$

This implies :
$(3^k)T(\frac{n}{2^k}) \geq T(n) \geq (3^k)T(\frac{n}{8^k})$

using value of last k and T(1)
$T(1) = T(\frac{n}{2^k})$
$n = 2^k$
putting log both sides and assuming log base as 2
$log_2 n = k$
$k = logn$

$= 3^k T(1)$
putting value of k
$= 3^{(logn)}$
$=> 3^{logn} = n^{log3}$
$T(n) = \theta(n^{log_2 3})$ for recursive part
for non recursive, $\quad T(n) = n(\frac{7}{8}^k) = \theta(n)$

$T(n) = \theta(n) + \theta(n^{\log_2 3})$
$\boxed{T(n) = \theta(n^{\log_2 3)}}$


DsPrac – SLL.java
DsPrac src SLL.java SLL intersect
Project
DsPrac ~/IdeaProjects/DsPrac
.idea
out
src
Main.java
SLL.java
DsPrac.iml
External Libraries
Scratches and Consoles
SLL.java
}
while (curr1 != null && curr2 != null)
{
if (curr1.data == curr2.data)
{
return curr1.data;
}
curr1 = curr1.next;
curr2 = curr2.next;
System.out.println("difference in length is"+distance);
}
return -1;
}
int getCount(Node node)
Run: SLL
/Users/mumukshapant/Library/Java/JavaVirtualMachines/openjdk-19.0.1/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA CE.app/Contents/lib/idea_rt.jar
difference in length is2
intersection is 15
Process finished with exit code 0