NATIONAL UNIVERSITY OF SINGAPORE
SCHOOL OF COMPUTING

TERM TEST FOR
Semester 2, AY2019/2020

CS3230 – DESIGN AND ANALYSIS OF ALGORITHMS

7 Mar 2020 Time Allowed: 2 hours

Instructions to Candidates:

1. This paper consists of **FOUR** questions and comprises **FOURTEEN (14)** printed pages, including this page.
2. Answer **ALL** questions.
3. Write ALL your answers in this examination book.
4. This is an **OPEN BOOK** examination.

Matric. Number: _____
Tutorial Group Number: _____

QUESTION	POSSIBLE	SCORE
Q1	20	
Q2	20	
Q3	30	
Q4	20	
TOTAL	**90**	

IMPORTANT NOTE:

- *You can **freely quote** standard algorithms and data structures covered in the lectures and homeworks. Explain **any modifications** you make to them.*
- *Unless otherwise specified, you are expected to **prove (justify)** your results.*

Q1. (20 points) Sorting out order of growth rates

(a) Rank the following functions in *increasing order of growth*; that is, if function $f(n)$ is *immediately* before function $g(n)$ in your list, then it should be the case that $f(n)$ is $O(g(n))$.

$g_1(n) = \sum_{i=1}^{\lg n - 1} \lg \lg \frac{n}{2^i}$ $\quad$ $g_2(n) = \sum_{i=1}^{n-2} \lg \lg (n - i)$ $\quad$ $g_3(n) = (\lg n)!$

$g_4(n) = 2^{\lg \lg \lg n}$ $\quad$ $g_5(n) = 10^{\lg((\lg \lg n)!)/ \lg \lg n}$ $\quad$ $g_6(n) = \lg\big((\lg n)!\big)$

$g_7(n) = n^3$ $\quad$ $g_8(n) = 2^n$

To simplify notations, we write $f(n) << g(n)$ to mean $f(n) = o(g(n))$ and $f(n) \equiv g(n)$ to mean $f(n) = \Theta(g(n))$. For example, the four functions n^2, n, $(2013n^2 + n)$ and n^3 could be sorted in increasing order of growth as follows: ($n << n^2 \equiv (2013n^2 + n) << n^3$). *Proofs are not required for this problem.*

Soln: $g_1(n) = \Theta(lg\ n\ lg\ lg\ n)$, $g_2(n) = \Theta(n\ lg\ lg\ n)$, $g_3(n) = 2^{\Theta(lg\ n\ lg\ lg\ n)}$, $g_4(n) = \Theta(lg\ lg\ n)$, $g_5(n) = (lg\ lg\ n)^{\Theta(1)}$, $g_6(n) = \Theta(lg\ n\ lg\ lg\ n)$, $g_7(n) = \Theta(n^3)$, $g_8(n) = 2^n$.

$g_4(n) << g_5(n) << g_1(n) \equiv g_6(n) << g_2(n) << g_7(n) << g_3(n) << g_8(n)$.

(b) Can you show that $n^2 = O(e^n)$ by limit method?

Soln: $\lim_{n\to\infty} \frac{n^2}{e^n} = \lim_{n\to\infty} \frac{\left(\frac{dn^2}{dn}\right)}{\left(\frac{d\,e^n}{dn}\right)} = \lim_{n\to\infty} \frac{2n}{e^n} = \lim_{n\to\infty} \frac{\left(\frac{d\,2n}{dn}\right)}{\left(\frac{d\,e^n}{dn}\right)} = \lim_{n\to\infty} \frac{2}{e^n} = 0.$
Hence, $n^2 = O(e^n)$.

Q1. (continued…)

(c) Can you show that $n^2 + 3n \lg n = O(e^n)$ by definition? Please show the steps and state clearly what are c and n_0.

Soln: n0=2 and c=4. Note that $\frac{n}{\ln n}$ is an increasing function. When $n \geq n_0 = 2$, $\frac{n}{\ln n} \geq \frac{2}{\ln 2} \geq 2$. This implies $e^n \geq n^2$ when $n \geq n_0 = 2$.

$n^2 + 3n \lg n \leq n^2 + 3n^2 = 4n^2 \leq 4e^n = ce^n$ when $n \geq n_0 = 2$.

By definition, $n^2 + 3n \lg n = O(e^n)$.

Marking scheme: (8+6+6=20marks)

(1) total 8 marks for sorting 8 functions.

Give marks by finding the max sub string of the correct answer.
(for f1 = f6, can change order)
For each correct function, I have mark it use small check mark.

(2) total 6 marks

Most people use l'hospital, this is right.
Each wrong derivative of function will lost 1 mark.
(So many people get 5 because of the wrong derivative of e^x)

Common mistake:
derivative of e^x is x*e^x
Some people get the conclusion: limit is infinite.

There are several people use other ways to calculate the limit, but not very common.
For example, one students says: lim(n^2/e^n)=lim(2ln(n)/n) = lim(1/n) * lim (2ln(n)) = 0 * infinite = 0. This is wrong.

(3) total 6 marks

Many people write c and n0 but not show the proof.

3 correct way to proof : (1) induction
(2) nlglgn < n^2 and n^2< e^n so, n^2+3nlglgn<4e^n
(3)h(x)=f(x)-g(x), h(n0)<0 and h(x) is decrease func

Common mistake:
01. f(n0) > g(c0), so for n >n0, f(x) > g(x) (very common)
02. f(n0)>g(n0) and f'(n0) > g'(n0), so f(x) > g(x)
03. h(x)=f(x)-g(x), h(n0)<0 so f(x) < g(x)
04. use fact nlglgn < e^x but not show the proof

Q2. (20 points) Solving recurrence

(a) Among the following recurrences, can you find their time complexities? Please show the detail steps.

(i) $\mathbf{T(n) = 9T(n/3) + n^2 / \lg n}$

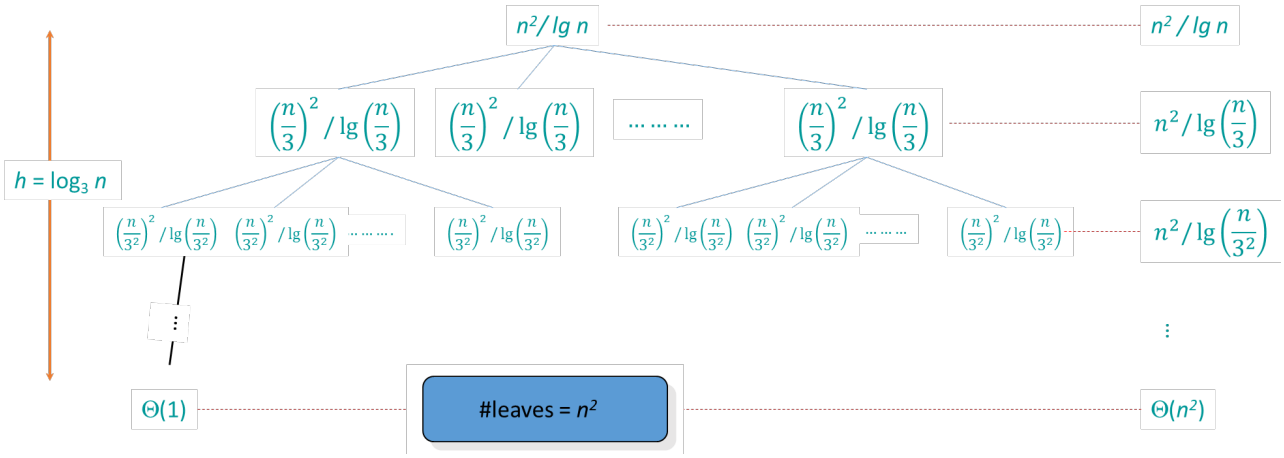
(ii) $\mathbf{T(2^n) = T(2^0) + T(2^1) + \ldots + T(2^{n-1}) + n^2}$

(iii) $\mathbf{T(n) = 5\ T(n/4) + n^4 / \lg \lg n}$

(iv) $\mathbf{T(n) = 4\ T(n/4) + n \lg \lg \lg n}$

(i) As shown by the following recursive tree, we have $T(n) = \frac{n^2}{\lg n} + \frac{n^2}{\lg(n/3)} + \frac{n^2}{\lg(n/3^2)} + \frac{n^2}{\lg(n/3^3)} + \cdots + \frac{n^2}{\lg(n/3^{h-1})}$, where $h = \log_3 n$.

Then, we have $T(n) = \frac{n^2}{\lg 3}\left(\frac{1}{\log_3 n} + \frac{1}{\log_3 n - 1} + \frac{1}{\log_3 n - 2} + \cdots + \frac{1}{1}\right) = \Theta(n^2 \lg \lg n)$.



Another solution: $\frac{T(n)}{n^2} = \frac{T(n/3)}{(n/3)^2} + \frac{1}{\lg n}$.

By telescoping, we have $\frac{T(n)}{n^2} = \frac{T(1)}{1^2} + \frac{1}{\lg 3}\left(\frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{\log_3 n}\right) = \Theta(\lg \lg n)$.

Hence, $T(n) = \Theta(n^2 \lg \lg n)$.

(ii) $T(2^n) = T(2^0) + T(2^1) + \ldots + T(2^{n-1}) + n^2$

Hence, $T(2^n) - T(2^{n-1}) = T(2^{n-1}) + n^2 - (n-1)^2$.

So, $T(2^n) = 2\ T(2^{n-1}) + 2n - 1$.

Set $N = 2^n$.

$T(N) = 2\ T(N/2) + 2 \lg N - 1$.

By Master theorem, a=2, b=2, $f(N) = 2 \lg N - 1$, $N^{\log_b a} = N$.

It is case 1. Hence, $T(N) = N$.

In other words, $T(2^n) = \Theta(2^n)$.

Q2. (continued…)

(iii) $T(n) = 5\,T(n/4) + n^4 / \lg\lg n$
$a=5, b=4, f(n)=n^4 / \lg\lg n, n^{\log_4 5} = n^{1.161}$.
This is case 3.

$af(n/b)=5f(n/4) = 5(n/4)^4/\lg\lg n = 5/256\ n^4/\lg\lg n < c\ f(n)$ for $1>c>5/256$. The regularity condition is satisfied.

Hence, $T(n) = \Theta(n^4 / \lg\lg n)$.

(iv) $T(n) = 4\,T(n/4) + n \lg\lg\lg n$

We can solve it using telescoping. Dividing both sides of equation (i) by n, we have:

$\frac{T(n)}{n} = \frac{T(n/4)}{n/4} + \lg\lg\lg n$.
By telescoping, we also have:
$\frac{T(n/4)}{n/4} = \frac{T(n/16)}{n/16} + \lg\lg\lg\frac{n}{4}$.
$\frac{T(n/16)}{n/16} = \frac{T(n/64)}{n/64} + \lg\lg\lg\frac{n}{16}$.
$\frac{T(n/64)}{n/64} = \frac{T(n/256)}{n/256} + \lg\lg\lg\frac{n}{64}$.
…
$\frac{T(4)}{4} = \frac{T(1)}{1} + \lg\lg\lg 4$.

By summing over all equations, we have:
$\frac{T(n)}{n} = \frac{T(1)}{1} + \lg\lg\lg n + \lg\lg(\lg n - 2) + \lg\lg(\lg n - 4) + \lg\lg(\lg n - 6) \ldots + \lg\lg\lg 4$.
$\frac{T(n)}{n} = \frac{T(1)}{1} + \Theta(\lg n \lg\lg\lg n)$.

Hence, $T(n) = \Theta(n \lg n \lg\lg\lg n)$.

Q2. (continued…)

Marking scheme:
5 marks per each part. If either answer is incorrect or step is incorrect, 2 marks. If you get correct answer but you don't use the standard method, I will still give full mark.

For (i), many student use Master theorem case 1 and get $T(n)=O(n^2)$. Not correct!
For (ii), many students say $T(n)=O(n^2)$. It is not tight and I mark it not correct.
For (iii), many students don't state the regularity condition.
For (iv), many students say $T(n) = \Theta(n \lg \lg(\lg n)!)$, which is not correct.
Many students use master theorem for (i), (ii) and (iv), which is not correct.

Q3. (30 points) Divide-and-conquer

You are given a set of points P={ (x_1, y_1), (x_2, y_2), …, (x_n, y_n) }. Assume the points are sorted such that $y_1 \le y_2 \le \ldots \le y_n$. We hope to find the closest pair of points. (Distance is measured using Euclidean metric.)

(a) Consider the following algorithm Closest(P) that return the distance of the closest pair of points. Can you show that the algorithm is correct?

Closest(P)
1. When P has 1 point, return ∞;
2. When P has 2 points, return their distance;
3. Find the median x_{med} of the x-coordinates of P by linear time select
4. Partition the point set P into two equal halves P_1 and P_2 by x_{med}
5. d_1=Closest(P_1);
6. d_2=Closest(P_2);
7. d=min{d_1, d_2};
8. for (x_i, y_i) in P_1,
9. for (x_j, y_j) in P_2,
10. $d = \min\left\{d, \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}\right\}$
11. Return d;

Soln: Step 3 finds the minimum distance between any two points in P_1.

We prove the correctness by induction.

Base cases: When P has 1 or 2 points, steps 1 and 2 report the answer correctly.

By strong induction, assume Closest(P) returns the shortest distance when P has less than n points.

The closest pair of P is either (1) both in P1, (2) both in P2, or (3) one in P1 while another one in P2.

For case 1, step 5 finds the pair by the induction hypothesis. (|P1|=n/2 < n)
For case 2, step 6 finds the pair by the induction hypothesis. (|P2|=n/2 < n)
For case 3, steps 8-10 find the pair.

By induction, the algorithm is correct for all n>0.

Hence, the algorithm is correct.

Q3. (continued…)

(b) Let T(n) be the running time of Closest(P) when P has n points. Can you give the recursive equation for T(n)? Please also compute the time complexity of Closest(P)?

Soln: $T(n) = 2\,T(n/2) + \Theta(n^2)$.
By Master theorem, it is case 3. Hence, $T(n) = \Theta(n^2)$.

Q3. (continued…)

(c) Suppose d_1=Closest(P_1), d_2=Closest(P_2) and d=min{d_1, d_2}. Let Q_1={ (x_i, y_i)∈P_1 | $x_{med}-x_i \leq d$} and Q_2={ (x_i, y_i)∈P_2 | $x_i - x_{med} \leq d$}. Let d' be the shortest distance among all point pairs between Q_1 and Q_2. Can you show that the shortest distance among all points in P is min{d, d'}?

Soln:
If d is the shortest distance, done.
Otherwise, there exists some point pair (p,q)∈ $P_1 \times P_2$, the distance between point p and point q is shorter than d.
The set of point pairs $P_1 \times P_2$ can be partitioned into three subsets: (1) $P_1 \setminus Q_1 \times P_2$, (2) $P_1 \times P_2 \setminus Q_2$ and (3) $Q_1 \times Q_2$.
For (1), for (p, q)∈$P_1 \setminus Q_1 \times P_2$, the distance between p and q must be bigger than d since the difference of the x-coordinates of p and q is bigger than d.
Similarly, for (2), for (p, q)∈$P_1 \times P_2 \setminus Q_2$, the distance between p and q must be bigger than d since the difference of the x-coordinates of p and q is bigger than d.
Hence, only the distance among point pairs between Q_1 and Q_2 can be shorter than d.

Q3. (continued…)

(d) For each point (x,y) in Q_1, only points (x', y') in Q_2, where $|y'-y| \leq d$, are within distance d from (x, y). The number of such points is at most 6. Can you show that this statement is true?

Soln:
By definition, all points in Q_2 are within width of d.
If the distance between (x, y) and (x', y') is within d, we have $x-d \leq x' \leq x+d$.
Hence, the points in Q_2 which have distance at most d from (x, y) should be within a rectangle of width d and height 2d.
Note that a rectangle box of width d and height 2d can contain at most 6 points whose pairwise distance is at least d.
Hence, there are at most 6 such points.

Q3. (continued…)

(e) Denote $Q_1=\{(x_1, y_1), (x_2, y_2), \ldots, (x_r, y_r)\}$ and $Q2=\{(x'_1, y'_1), \ldots, (x'_s, y'_s)\}$. Assume the points in both Q_1 and Q_2 are sorted by their y-coordinates (i.e. $y_1 \le y_2 \le \ldots \le y_r$ and $y'_1 \le y'_2 \le \ldots \le y'_s$). Can you give an O(n)-time algorithm Shortest(Q_1, Q_2, d) that computes min{d, d'} where d' is the shortest distance among the point pairs in $Q_1 \times Q_2$?

Shortest(Q_1, Q_2, d)

1. set j=1;
2. for i=1 to r
3. while ($y'_j < y_i - d$) j++;
4. h=j;
5. while ($y'_h \le y_i + d$)
6. $d = \min\left\{d, \sqrt{(x_i - x'_h)^2 + (y_i - y'_h)^2}\right\}$;
7. h++;
8. return d;

For steps 4-7, as there are at most 6 possible points (x'_h, y'_h), steps 4-7 takes O(1) time.

Steps 2 and 3 just scan the points in Q1 and Q2 in increasing order. They require O(n) time.

Hence, this function runs in O(n) time.

Q3. (continued…)

(f) Can you give an O(n log n) time algorithm to find the shortest distance among all points in P?

Closest(P)
1. When P has 1 point, return ∞;
2. When P has 2 points, return their distance;
3. Find the median x_{med} of the x-coordinates of P by linear time select
4. Partition the point set P into two equal halves P_1 and P_2 by x_{med}
5. d_1=Closest(P_1);
6. d_2=Closest(P_2);
7. d=min{d_1, d_2};
8. Set $Q_1 = \{ (x_i, y_i) \in P_1 \mid (x_{med}-x_i) \leq d \}$;
9. Set $Q_2 = \{ (x_i, y_i) \in P_2 \mid (x_i-x_{med}) \leq d \}$;
10. d = Shortest(Q_1, Q_2, d);
11. Return d;

$T(n) = 2\,T(n/2) + \Theta(n)$

By master theorem, it is case 2. Hence, $T(n) = \Theta(n \log n)$.

Marking Scheme:

a)
Error in inductive proof (-0.5 to -1pts)
Fail to prove the loop compute d' = min distance of pair in P1 and P2 (-1.5pts)

b)

Show the correct recursive equation: T(n) = 2T(n / 2) + c(n^2) (2pts)

Solve the equation (3pts) (the equation must be first correct)

If using master theorem, lacking each condition -0.5pts (why case 3 and regularity condition)

c)

Argue that if the pair is not in (Q1, Q2) then distance is larger than d. (4pts)

Show that min(d, d') cover all pair of point in P (1pts)

d)

Mention that the points must be in rectangle size (d, 2d) (2pts)

Completeness of the proof (3pts)

e)

Give correct algorithm (4pts)

Proof of time-complexity (1pts)

f)

Give correct algorithm (4pts)

Proof of time-complexity (1pts)

Common mistake:

a)

Mix up iterative algorithm proof and induction.

Hypothesis that Closest(P) is true (what the question ask)

Using hypothesis that Closest(P') is true for $|P'| < n = |P|$, without last induction step to prove the algorithm is correct for all n.

Assume Closest(P1) is true without mentioning $|P1| = n / 2 < n$

b)

Using Master Theorem without proving the asymptotic bound.

Using case 3 of Master Theorem without checking regularity condition.

c)

Proving the distance (or x-distance) to median point need to be less than d(then?)

Proving that can exclude (a, b) in (P1 - Q1, P2 - Q2) only (need Q1 x (P2 - Q2) also)

e)

for a in Q1 : for b in Q2 : If (dist(a, b) > d) break; (wrong)

f)

Not mention how to build Q1, Q2 from P1, P2

Sort points in P1, P2 (already sorted)

Shortest(P1, P2, d) : not O(n)

No time complexity proof.

Note: Challenge may result in lower score. The schemes for Q3 are already designed to be flexible where possible.

Q4. (20 points) Randomized Algorithm and Lower bound

(a) Consider a hash table with 100 entries. We store k arbitrary values $x_1, x_2, \ldots, x_k$ into the hash table. What is the minimum k so that we expect there are at least 5 pairs of values (x_i, x_j) such that $h[x_i]=h[x_j]$? (Note that the hash function h is a randomized procedure satisfying the universal hashing assumption.)

Soln:
Let X_{ij} be the indicator random variable that $h[x_i]=h[x_j]$. So, $E[X_{ij}]=1/n$.

Let X be the random variable of the number of pairs (x_i, x_j) such that $h[x_i]=h[x_j]$. In other words, $X = \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} X_{ij}$.

$E(X) = \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} E(X_{ij}) = \binom{k}{2} \frac{1}{n} = \frac{k(k-1)}{2n}$.
We aim to find the minimum k such that $E(X) \geq 5$.
So, $k^2 - k - 10n \geq 0$. Hence $k \geq \frac{1+\sqrt{1+40n}}{2}$.
So, the minimum k is $\left\lceil \frac{1+\sqrt{1+40n}}{2} \right\rceil$.
Since n=100, $k = \lceil 32.13 \rceil = 33$.

Marking scheme:

1. If you can correctly write the universal hashing assumption, I give you 1 point
2. If you can correctly write the expression of E[X] where X is the number of collisions, I give you 3 more points.
3. If you can argue E[X] <= (k choose 2) 1/100, I give you 2 more points.
4. 4 final points will be given for correct argument for the answer.
5. Partial points are given:
    a. For some insights of the problem but went wrong at some point (for example, assump that h is uniformly distributed).
    b. Miscalculation.
    c. You wrote something to prove that you have some certain insights about the problem, not just copying from slides.

Common mistakes:

1. Incorrect citation of UHA. UHA just guarantees you that Pr(h(x_i) = h(x_j)) <= 1 / m.
2. Argument about chains' length, which is irrelevant.
3. Assumption that h is uniformly distributed.

Q4. (continued…)

(b) There are 2^k balls. All balls have the same weight and the same volume except that exactly one ball is heavier and exactly one ball is bigger. (It is possible that the ball which is heavier is the same as the ball which is bigger.) You have a weight machine and a volume machine. Given two groups of balls, the weight machine reports the group which is heavier; otherwise, it reports same weight. The volume machine reports the group which has bigger volume; otherwise, it reports same volume.

i. Can you give the exact lower bound (not asymptotic bound) of the number of measurements to determine the heavy ball and the big ball?

ii. Can you give an optimal algorithm that identifies the heavy ball and the big ball?

Soln: (i) Let $n=2^k$. Suppose the i^{th} ball is the heavy ball and the j^{th} ball is the big ball. There are n choices for i and n choices for j. Hence, the number of combinations is n^2. By decision tree model, the lower bound is $\lceil 2\log_3 n \rceil = \lceil 2k \log_3 2 \rceil$.

(ii) The algorithm is simple. First, for iteration i=1, 2, … $\lceil k \log_3 2 \rceil$, we separate $3^{\lceil k \log_3 2 \rceil - i + 1}$ balls into 3 groups where each group has $3^{\lceil k \log_3 2 \rceil - i}$ balls. compare weights of two groups of balls. If one group is heavier, we know the heavy ball is in the heavier group. If both groups are of each weight, we know the heavy ball is in the remaining group. In other words, after the weight, we can identify $3^{\lceil k \log_3 2 \rceil - i}$ balls that contain the heavy balls. Then, we go to the next iteration. In total, we use $\lceil k \log_3 2 \rceil$ iterations of the weight machine to identify the heavy ball. Next, using the same approach, we use $\lceil k \log_3 2 \rceil$ iterations of the volume machine to identify the big ball. In total, we use $2\lceil k \log_3 2 \rceil$ iterations.

Marking scheme:

1. For part 1, 5 marks for correct EXACT bound (even you don't give argument).
2. For part 2, 5 marks for correct algorithm.
3. If you can give binary search algorithm, I give you 2 marks.
4. Partial marks are given to almost correct answers:
    a. You lose 1 mark if you forget to include the sum lower bound of two machines.
    b. You lose 2 marks if you give asymptotic bound.

Common mistakes:

1. Give asymptotic bound (by wrapping it inside big-O, Omega notations). We want an exact bound (clearly stated).
2. Use a decision tree to reason about lower bound, but present a **binary** tree. Remember that a measurement has 3 outcomes (left, right, equals), then the tree must be a **ternary** tree.
3. Many of you gave lower bounds of 2, 3, or some constants. Remember when it comes to lower bound, we (mostly) want lower bounds in the **worst case,** not in the best case.

-- End of Paper ---