211A-COMSCI180-1 Midterm

MENGAN WANG

TOTAL POINTS

93 / 100

QUESTION 1

1 Problem 1 **35 / 40**

- **- 0 pts** Correct
- **- 10 pts** Define: “city” is a graph with edges that are streets and vertices that are their intersections.
- **- 10 pts** show how to go from one intersection to another one
- **- 10 pts** Condition 1: Yupi is reachable
- ✓ **- 10 pts Condition 2: all intersections are visited but Yupi is not found (The graph is not connected)**
- **- 10 pts** Time complexity estimation
- ✓ **+ 5 pts Partial credits**
- **- 10 pts** Miss the algorithm steps
- **- 5 pts** Complexity is wrong

QUESTION 2

2 Problem 2 **28 / 30**

- **- 0 pts** Correct.
- **- 10 pts** Missing analysis/explanation.
- **- 10 pts** Incorrect ordering.
- **- 10 pts** Incorrect analysis.
- **- 5 pts** Incorrect simplification.
- **- 5 pts** F5 and F2 have the same order of growth rate.
- **- 5 pts** Insufficient analysis.
- ✓ **- 2 pts Minor mistakes.**

QUESTION 3

3 Problem 3 **30 / 30**

- ✓ **- 0 pts Correct**
- **- 5 pts** Missing the conclusion that the statement is false.
- **- 5 pts** Counter example is unclear.
- **- 10 pts** The statement is still false when n=3.
- **- 10 pts** Insufficient reasoning.
- **- 25 pts** The statement should be false.

Name: Mengan Wang **SID#** 805605806

CS 180

Midterm Test

Summer 2021

Due not later than 10 a.m. PT *Thursday*, *July* 22, 2021

Problem 1 (40 pts). Men, women and yupi live on the planet Alphaomega. Imagine you find yourself in the city Aleph on this planet. You know that a generous yupi lives there who gives a flying car to a person when he first meets this person. You also know that there is a big letter Y on the house of the generous yupi and there are no other houses with this sign. You want to get a flying car but you don't know where the generous yupi lives and other inhabitants of Aleph cannot help you because you don't know their language and they don't know your language. Design an algorithm that will allow you to find the generous yupi if you know that his castle is at one of the intersections in Aleph.

graph traversal along intersection

a) 20 pts for any correct algorithm.

b) Up to 40 pts for an efficient algorithm with estimation of its time complexity.

work on next page

Problem 2 (30 pts). Take the following functions and arrange them in descending order of growth rate indicating when functions have the same order of growth rate. → sort by big O

1. $\sqrt[3]{n} = n^{\frac{1}{3}}$
2. $\log_{10}2^n = n \log 2 \to n$
3. $\sqrt[4]{n^3} = n^{\frac{3}{4}}$
4. $\sqrt{n^n} = n^{\frac{n}{2}}$
5. $\log_{10}{}^2 n + 5n = (\log n)^2 + 5n \to n$
6. $5n + 5^n$
7. $n^{7/2}$
8. $\log_{10}(n^5 + 5n)$

work on next page

Problem 1:

Design an Algorithm:

Assume that the city and all its intersections can be represented as a graph G with an adjacency list and that you are located at a node currently inside. Also assume that G is undirected and you can travel either direction from intersection to intersection. G may also contain cycles.

With these conditions in mind, we can move on to designing our algorithm.

We will design a depth-first search algorithm because we only need to find the generous Yupi, and not the shortest path to it. DFS is capable of traversing all nodes in a graph.

Because the graph has cycles, we must also keep track of the intersections we visit to not get trapped in loops.

Whenever we reach an intersection, we check for the Generous Yupi. If we find him then we are done. Otherwise we choose to visit another intersection that is connected to where we currently are. If there are no unvisited neighboring intersections, then we backtrack to the last location where there are unvisited neighbor intersections.

pseudocode next pg →

Thus, we have the following algorithm:

```
findGenerousYupi (G, u):
  if house with letter Y at intersection:
      then we found the Generous Yupi and stop
  end if
  mark u as visited
  for ∀v ∈ neighbors of u in G:
      if v not visited:
          then recurse findGenerousYupi (G, v)
      end if
  end for
```

Time Complexity:

The program checks whether we've found the Generous Yupi and marks nodes we've visited, which are $O(1)$ operations. Thus, our time complexity is bounded by the number of times our program recurses and performs the operations. Our program will have a time complexity of $O(m+n)$, which is linear in the input size (m are the nodes and n are the edges). This is because due to checking whether a node has been visited, we access each node at most 1 time. Thus worst case we visit every node and edge before finding the generous Yupi which is $m+n$ intersections visited, and

$O(m+n)$

1 Problem 1 35 / 40

- **- 0 pts** Correct
- **- 10 pts** Define: “city” is a graph with edges that are streets and vertices that are their intersections.
- **- 10 pts** show how to go from one intersection to another one
- **- 10 pts** Condition 1: Yupi is reachable
- ✓ **- 10 pts Condition 2: all intersections are visited but Yupi is not found (The graph is not connected)**
- **- 10 pts** Time complexity estimation
- ✓ **+ 5 pts Partial credits**
- **- 10 pts** Miss the algorithm steps
- **- 5 pts** Complexity is wrong

Name: Mengan Wang **SID#** 805605806

CS 180
Midterm Test
Summer 2021

Due not later than 10 a.m. PT *Thursday*, *July* 22, 2021

Problem 1 (40 pts). Men, women and yupi live on the planet Alphaomega. Imagine you find yourself in the city Aleph on this planet. You know that a generous yupi lives there who gives a flying car to a person when he first meets this person. You also know that there is a big letter Y on the house of the generous yupi and there are no other houses with this sign. You want to get a flying car but you don't know where the generous yupi lives and other inhabitants of Aleph cannot help you because you don't know their language and they don't know your language. Design an algorithm that will allow you to find the generous yupi if you know that his castle is at one of the intersections in Aleph. graph traversal along intersection

a) 20 pts for any correct algorithm.

b) Up to 40 pts for an efficient algorithm with estimation of its time complexity.

work on next page

Problem 2 (30 pts). Take the following functions and arrange them in descending order of growth rate indicating when functions have the same order of growth rate. → sort by big O

1. $\sqrt[3]{n} = n^{\frac{1}{3}}$
2. $\log_{10}2^n = n \log 2 \rightarrow n$
3. $\sqrt[4]{n^3} = n^{\frac{3}{4}}$
4. $\sqrt{n^n} = n^{\frac{n}{2}}$
5. $\log_{10}{}^2 n + 5n = (\log n)^2 + 5n \rightarrow n$
6. $5n + 5^n$
7. $n^{7/2}$
8. $\log_{10}(n^5 + 5n)$

work on next page

Problem 2:

We will start by simplifying each function:

① $\sqrt[3]{n} = n^{\frac{1}{3}}$

② $\log_{10} 2^n = n \log 2 \Rightarrow O(n)$

$$\lim_{n\to\infty} \frac{n \log 2}{n} = \log 2$$

as the limit is constant, $\log_{10} 2^n$ is $\underline{O(n)}$

③ $\sqrt[4]{n^3} = n^{\frac{3}{4}}$

④ $\sqrt{n^n} = n^{\frac{n}{2}}$

⑤ $\log_{10}^2 n + 5n = (\log n)^2 + 5n \Rightarrow O(n)$

$$\lim_{n\to\infty} \frac{(\log n)^2 + 5n}{n} = \lim_{n\to\infty} \frac{\frac{2\ln x}{x} + 5}{1} = \lim_{n\to\infty} \frac{2\ln x}{x} + 5 = 5$$

as the limit is constant, $\log_{10}^2 n + 5n$ is $\underline{O(n)}$

⑥ $5n + 5^n \Rightarrow \underline{O(5^n)}$

⑦ $n^{\frac{7}{2}}$

(8) $\log_{10}(n^5 + 5n) = \log(n^5 + 5n)$

$$\lim_{n \to \infty} \frac{\log(n^5 + 5n)}{\log n} = \lim_{n \to \infty} \frac{\frac{5n^4 + 5}{n^5 + 5n}}{\frac{1}{n}} = \lim_{n \to \infty} \frac{5n^5 + 5n}{n^5 + 5n}$$

$$= \lim_{n \to \infty} \frac{5 + \frac{5}{n^4}}{1 + \frac{5}{n^4}} = 5$$

as the limit is constant, $\log_{10}(n^5 + 5n)$ is $\underline{O(\log n)}$

Now we will sort the functions in descending growth rate:

functions 1, 2, 3, 5, 7 are polynomial

4, 6 are exponential

8 is logarithmic

Let us sort the polynomial functions first:

(1) $n^{\frac{1}{3}}$ (2) n (3) $n^{\frac{3}{4}}$ (5) n (7) $n^{\frac{7}{2}}$

because they are all polynomials, we can compare their growth rates by comparing their exponents. Thus, the order is:

$$(7)\ n^{\frac{7}{2}} > (2)\ n = (5)\ n > (3)\ n^{\frac{3}{4}} > (1)\ n^{\frac{1}{3}}$$

Let's sort the exponential functions:

④ $n^{\frac{n}{2}}$ ⑥ 5^n

top gets multiplied by $\ln n$

$$\lim_{n\to\infty} \frac{n^{\frac{n}{2}}}{5^n} = \lim_{n\to\infty} \frac{\frac{n^{\frac{n}{2}}(\ln(n)+1)}{2}}{5^n \ln 5} = \lim_{n\to\infty} \frac{n^{\frac{n}{2}}(\ln n + 1)}{2\ln 5 \cdot 5^n} = \infty$$

bottom gets multiplied by constant

So ④ $n^{\frac{n}{2}}$ > ⑥ 5^n

We also know exponential growth rate is faster than polynomial.

Finally, we compare the rates of the slowest growing polynomial to the logarithm

$$\lim_{n\to\infty} \frac{n^{\frac{1}{3}}}{\log n} = \lim_{n\to\infty} \frac{\frac{1}{3n^{\frac{2}{3}}}}{\frac{1}{n\ln 10}} = \lim_{n\to\infty} \frac{1}{3}\ln(10)\, n^{\frac{1}{3}} = \infty$$

Thus ① $n^{\frac{1}{3}}$ grows faster than ⑧ $\log n$

The final order is:

$$\boxed{\begin{gathered} ⑥\ O(5n + 5^n) > ④\ O(\sqrt{n^n}) > ⑦\ O(n^{\frac{7}{2}}) > ②\ O(\log_{10} 2^n) = \\ ⑤\ O(\log_{10}^2 n + 5n) > ③\ O(\sqrt[4]{n^3}) > ①\ O(\sqrt[3]{n}) > ⑧\ O(\log_{10}(n^5 + 5n)) \end{gathered}}$$

2 Problem 2 **28 / 30**

- **- 0 pts** Correct.
- **- 10 pts** Missing analysis/explanation.
- **- 10 pts** Incorrect ordering.
- **- 10 pts** Incorrect analysis.
- **- 5 pts** Incorrect simplification.
- **- 5 pts** F5 and F2 have the same order of growth rate.
- **- 5 pts** Insufficient analysis.
- ✓ **- 2 pts Minor mistakes.**

Problem 3 (30 pts). On the planet Alphaomega, there are *n* spaceships and *n* persons having the rank of a spaceship captain. Each captain has the preference list of spaceships and the crew of each spaceship has the preference list of captains. The goal is to find a Stable Spaceship Matching of pairs (*c*, *s*).

Decide whether the following statement is true or false.

In every instance of the Stable Spaceship Matching, there is a stable matching containing a pair (*c*, *s*) *such that, at least, one of them is ranked third on the preference list of the other.*

If it is true, give a short explanation and design an algorithm.

If it is false, give a counterexample and explain that it is correct.

$c_1 \leftrightarrow s_1$ $\quad$ $c_2 \leftrightarrow s_2$ $\quad$ $c_3 \leftrightarrow s_3$

there is no arrangement with ranked 3rd

1 instance

*proof written on next page

Problem 3:

The statement that "every instance of the stable spaceship matching, there is a stable matching pair (c, s) such that, at least, one of them is ranked third on the preference list of each other" is false.

I will demonstrate this through proof by contradiction and providing an example of a stable matching where there does not exist a pair, where either the crew or spaceship is ranked third on the preference list of the other when $n=3$.

Let us suppose spaceships s1, s2, s3 and crew c1, c2, c3 have the following preferences:

	1st	2nd	3rd
s1	c1	c2	c3
s2	c2	c3	c1
s3	c3	c1	c2

	1st	2nd	3rd
c1	s1	s2	s3
c2	s2	s3	s1
c3	s3	s1	s2

Thus, the only stable matching set in this situation is:
$\{(s1, c1), (s2, c2), (s3, c3)\}$

Each spaceship and crew is matched with their top choice, thus no pairs exist where one of them are put with their 3rd ranked preference.

The statement is proved false.

3 Problem 3 **30 / 30**

- ✓ **- 0 pts** Correct
- **- 5 pts** Missing the conclusion that the statement is false.
- **- 5 pts** Counter example is unclear.
- **- 10 pts** The statement is still false when n=3.
- **- 10 pts** Insufficient reasoning.
- **- 25 pts** The statement should be false.