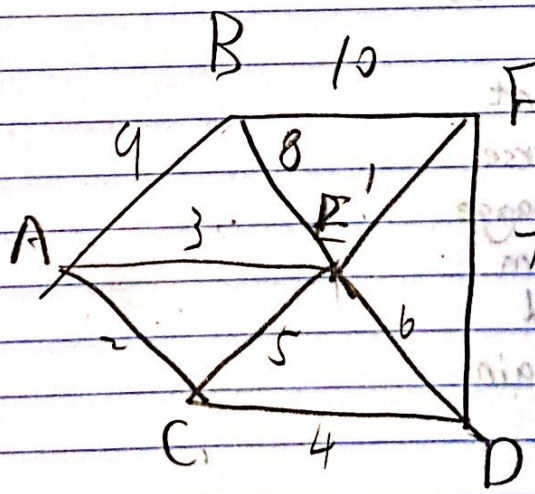
1.a)

Top	TOO	T oo	OPs
POT	To P	T oP	OPT
Too	POP	P OP	POP
OPT →	SOP →	SOP →	POT
OPs	OPs	POT	SOP
SOT	POT	SOT	SOT
POP	OPT	OPs	TOO
SOP	SOT	OPT	TOP

✓

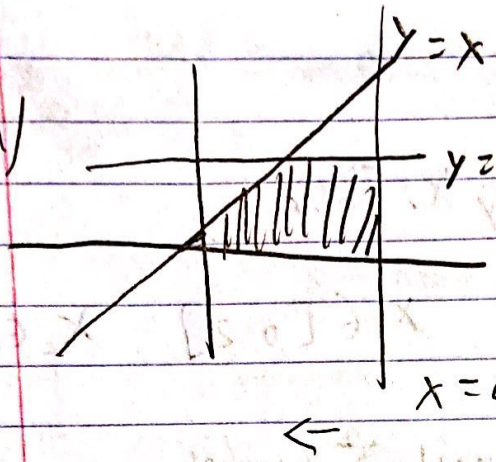
2 a) 12348



b) 23148

c) 23149

CMSC351 Final Yizhan Ao



n points

Prob 1)

Show some points by their distance to y axis in avg linear time.

Assum bucket sort works in linear time. $n = 6k$

We can divide the points into 6 intervals based on the n value, there will be points (x, y) that $\frac{2}{3}(i-1) \leq x_i \leq \frac{2}{3} i$

There will be points (x, y) that $\frac{2}{3}(i-1) \leq x_i \leq \frac{2}{3} i$

the using an empty array and the to use the bucket sort to sort the Array from 0 to n-1 that we append each point to be one of the elements.

b. pseudo code

To Input Array X Array Y, Int n

Split x into X_1 & X_2 $\quad X_1 \in [0,2)$, $X_2 \in [2,4]$

for $X_1 \in [0,2)$ we need $\frac{n}{3}$ buckets

for $(i = 1$ to $\frac{n}{3})\{$

put X[i] into Bucket $B[\frac{n}{6} X_1[i]]$

end for

sort each Bucket

Making each Bucket

For $[X_2 \in [2,4]$ use $\frac{2n}{3}$ Buckets

for $(i = 1$ to $\frac{2}{3}n)$

put $X_2[i]$ to $B[\frac{1}{3}n \cdot [X_1(i) - 2]]$

en sort each Bucket

Make each Bucket

end for

sort X_1, X_2

end for

4. a) $\overline{T(n)} = \frac{1}{n}\sum_{i=1}^{n-1} T(i) + n-1 \quad T(1)=0$

b) since, on average, the list is cut in half we think $T(n)$ is bounded by a linear function $an=b$

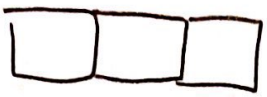
So the rest is good

5a) Since each pair of neighbors can only be involved with communication, we can have every other column of processor send their number to the right neighbor. Then we can have every other row of processor to be able to move up. The move to the right neighbor. Then we have every other row of processor to move up. The move to the right of each added other column to its neighbor column then decrease by a factor of 2. Then move to the up added every other row to its neighbor decrease the factor by 2 again. Repeat the process until we add up all numbers.

5b) $O(n^3)$

5c) $O(\sqrt[3]{n})$

Problem 6 Room Room hold 3 students



H(i,j,k) n×n×n Matrix

n is multiple of 3

partition {1...n} into T to maximize H

1) optimization H(i,j,k) n is 3k k∈ℕ

T

Assume: i, j, k rankings are going from top to low ①

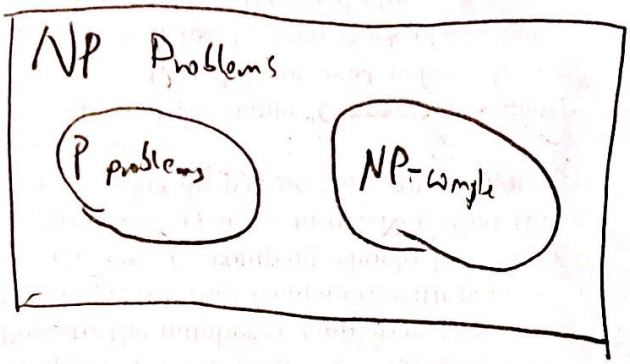
---

2) Decision version of roomate asignment n×n×n

H(i,j,k) ∃ n G //goal T ⇒ {1...n}

Q: is there T into sum H(i,j,k) At least G

a) SHOW Decision of Roomate assignment Problem is in NP



We are proving this is in P
Therefore since P is inside of NP problem. we can tell this problem is a NP problem

Prove the verification is in a polynomial time

Given a matrix of H (i,j,k)

Assume there are 5 students representing A, B, C, D, E, -

H Matrix

	i	j	k
A:	B.	C.	D)
B:	C	E	D)
C:	B	E	D)
D:	A	E	C)
E:	A	B	C)

Assume (i,j,k) are different people &
Assume they can't choose themselves to be (i,j,k)

Don't consider whe i j are same. ☆
We don't know what G is

To make i j k meaningful we make i=1, j=0, k=-1, not in ijk is -2

Algorithm: Firstly, we let each person ask their favorite people (i) to live together, then move on to the 2nd favourite (j) Each response will be "no" or "yes", If response Yes. then live together. No move on

☆ Room(1 2)

Secondly. If the (i) says no, then each person ask their second choice if it is "Yes" together. If Yes, live together. If no, move on.

Thirdly. ~~Ask~~ If (j) says no, then move on to $(\vec{k})$. ~~The rest~~

The rest will be move to a same room if the answ is "No" again. The rest of $(\vec{k})$ people will form a room

The complexity so far is $(n^{\vec{k}})$ Therefore, it is a polynom algorithm. Therefore it is a NP problem - completeness.

If G is solvable

b) Show solve optimization in Polynomial time

```
function optimaation (matrix H(i,j,k), int n){
  it threepeople[n House == true then
     return true
  end
  else
     retun false
  end
  Graph X = (V, E);
     if( V = (1·n) mod(3) ) // n is divisible by 3
        if(each (T) . contains 3E)
           for(each T set {x1, x2, x3})
           Make Ti = {x1, x2, x3}
              {x1, x2}, {x2, x3} {x1, x3} ∈ E.
              threepeople In House == true
        end if
     end if
}
```

A = [ ] new matrix

B = [ ] new matrix

C = [ ] new matrix

M = H // represently all.

If there is a V into n sets of H (T)

Runtime $O(n^3)$ If Optimization is solved by this therefore decision is also in NP-completeness

4) If solve decision in Poly then solve opt'd poly

function decision (matrix H(i,j,k), int n, int G) {

if threepeople In the House (== true ; < threepeople In the House == false

return true;

end if

else

return false;

M = H;

if M matches I(G) // I(G) is the solution of perfect matching

threepeople In the House = true; so that every one can get a house guaranteed

end if

else I = [a, b, c, d] // a, b, c, d are in the M

a b c and are people representing here

I = output [a, b, c, d] // d is the match

Match M for I with {a, b, c} & {(e, x, y)}

three people In the house = true // based on their level in {i, j, k}

For (each element in I)

add (i, j, k); // add value to represent likeness

end for

end else

}

runtime $O(\sum_{i=1}^{m} r_i) = O(n^2)$