Xueying Chen
OUID: 113366600

DSA4413 – Fall 2019
Homework #2

1)

1 day = 3600 * 24 sec = 86400 sec

T(n) = 15n^2 * 10^-9 = 86400
n^2 = 86400 / (15 * 10^-9)
n = sqrt (86400 / (15 * 10^-9))
n = [0, 2400000], when T(n) = 15n^2, the can solve problem when n=0 to n=2400000

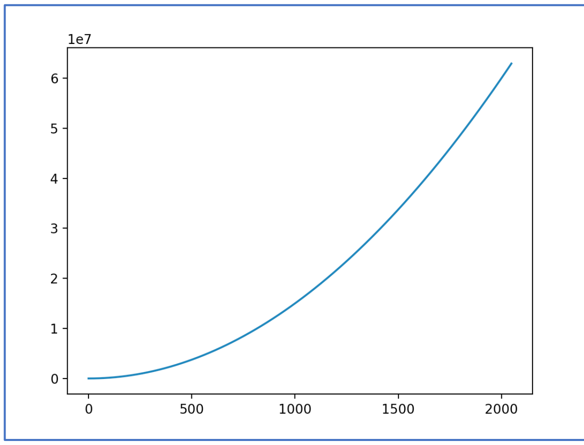
T(n) = 8n^3 * 10^-9 = 86400
n^3 = 86400 / (8 * 10^-9)
n = (86400 / (8 * 10^-9)) ^ (1/3)
n = 22104.18899
n = [0, 22104] for T(n) = 8n^3

T(n) = 2^n * 10^-9 = 86400
2^n = 86400/10^-9
n = log (86400/10^-9, 2)
n = 46.3
n = [0, 46] for T(n) = 2^n

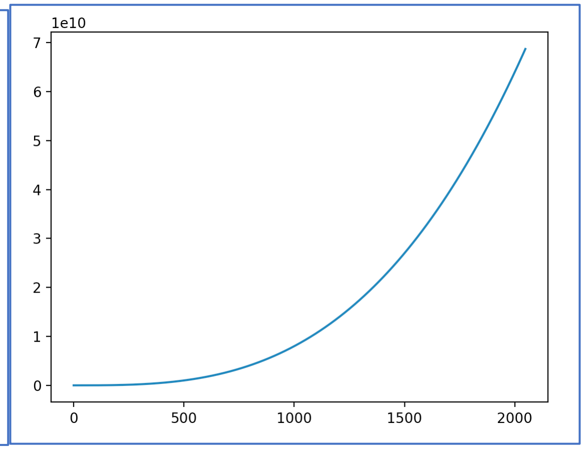
T(n) = 3^n * 10^-9 = 86400
n = log (86400/10^-9, 3)
n = 29.2
n = [0, 29] for T(n) = 3^n

T(n) = n! * 10^-9 = 86400
n! = 86400 / 10^-9
n! <= 8.64E+13
n <= 16
n = [0, 16] for Tn= n!

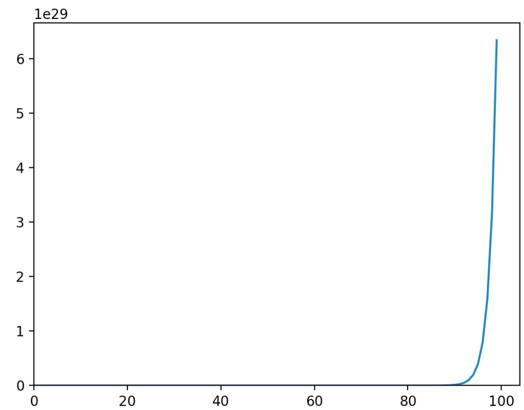
T(n) = n log n * 10^-9 = 86400
n log n = 86400 / 10^-9
n = (86400 / 10^-9) / lambertw(86400 / 10^-9), where W is lambert W function.
Solve n using python
n = 3007100369769
n = [0, 3007100369769] for T(n) = n log n



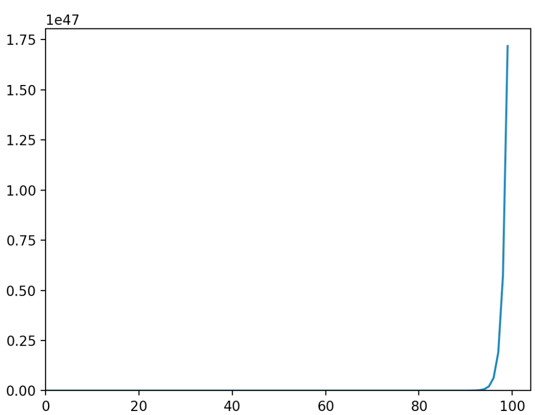
T(n) = 15n^2



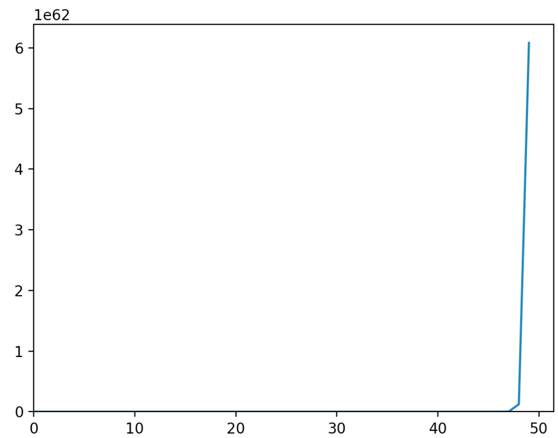
T(n) = 8n^3



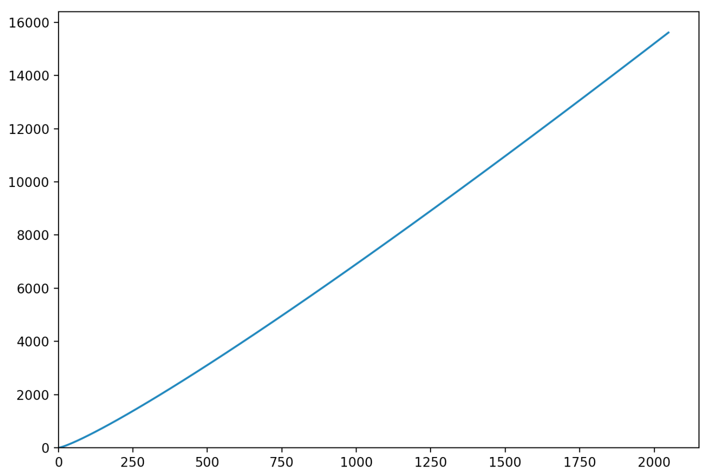
T(n) = 2^n, only plot n=[0, 100], n goes to infinite quickly



T(n) = 3^3, only plot n=[0, 100], n goes to infinite quickly.



T(n) = n!, only plot n=[0, 50], n goes to infinite quickly.



T(n) = nlogn

2)

Insertion sort:

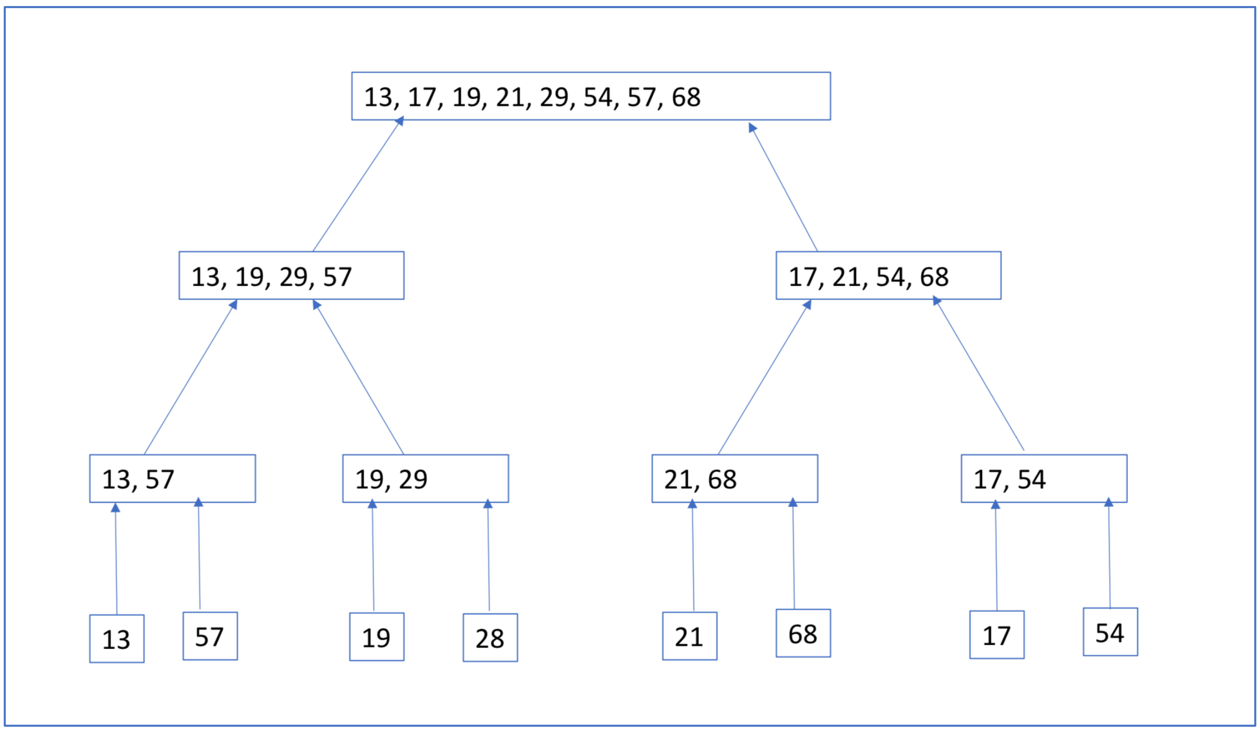
A = <13, 57, 19, 28, 21, 68, 17, 54>

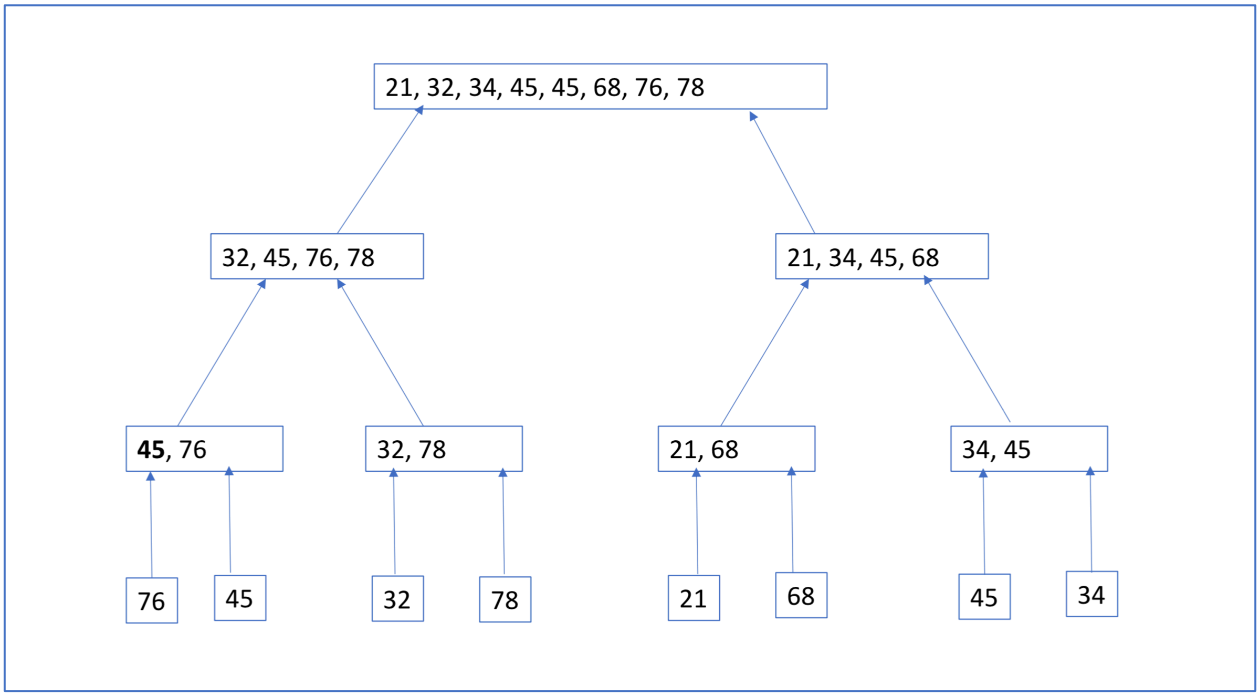
a) 13, 57, 19, 28, 21, 68, 17, 54 (0 insertion)
b) 13, 19, 57, 28, 21, 68, 17, 54 (1 insertion, swap(57, 19)
c) 13, 19, 28, 57, 21, 68, 17, 54 (1 insertion, swap(57, 28)
d) 13, 19, 21, 28, 57, 68, 17, 54 (2 insertions swap(57, 21) and (28, 21)
e) 13, 19, 21, 28, 57, 68, 17, 54 (1 insertion)
f) 13, 17, 19, 21, 28, 57, 68, 54 (5 insertions)
g) 13, 17, 19, 21, 28, 54, 57, 68 (2 insertions)

B = <76, 45, 32, 78, 21, 68, 45, 34>

a) 45, 76, 32, 78, 21, 68, 45, 34 (1 insertion, swap(76, 45))
b) 32, 45, 76, 78, 21, 68, 45, 34 (2 insertions, swap (76, 32), swap(45, 32))
c) 32, 45, 76, 78, 21, 68, 45, 34 (0 insertion)
d) 21, 32, 45, 76, 78, 68, 45, 34 (4 insertions)
e) 21, 32, 45, 68, 76, 78, 45, 34 (2 insertions)
f) 21, 32, 45, 45, 68, 76, 78, 34 (3 insertions)
g) 21, 32, 34, 45, 45, 68, 76, 78, (5 insertions)

Merge Sort





Merge Sort performs better. Both A & B are not sorted. We expect insertion sort to have time complexity of T(n) = C*n^2. As you can see from insertion steps above, each step we have to some swapping. Mergesort has average complexity of T(n) = C*nlog(n). For n=8, insertion sort has have C*64 operations, mergesort has C*7.2 operations, where C is a constant. Therefore, mergesort is the better performing algorithm.

3)

Worst-case complexity – for inputs size n, worst-case complexity of the algorithm is the maximum number of operations it takes to solve the problem

Best-case complexity – for inputs size n, worst-case complexity of the algorithm is the maximum number of operations it takes to solve the problem

Average-case complexity – for input size n, average-case complexity of the algorithm is the average = number of operations it takes to solve the problem

For sequential search

Worst-case complexity T(n) = n, we compared every element in inputs

Best-case complexity T(n) = 1, we found element in first input.

Average-case-complexity

There are (n+1) distinct events. Found an element can occur in n ways, and Not found can occur in 1 way, therefore there are (n+1) events

Ta(n) Expected number of operations = Sum (probability of event i * number of operations in event i) for i = 1 to n

$Ta(n) = (1/n+1) * 1 + (1/n+1) * 2 .... + (1/n+1) * n + (1/n+1) * n$
$Ta(n) = (1/n+1) * [sum(i, for i=1 to n) + n]$

Using formula sum(i, for i=1 to n) = n(n+1) /2

$Ta(n) = (1/n+1) * (n(n+1)/2 + n)$
$Ta(n) = (n(n+1) / 2(n+1) + n/n+1$
$Ta(n) = n/2 + n /(n+1)$
$Ta(n) = n/2 + 1$ when n is large.

Prove
sum(i, for i=1 to n) = n(n+1) /2 using induction

For n = 1
sum(i, for i=1 to n) = 1, and 1(1+1)/2 = 1. It is true for n=1.

Assume that sum(i, for i=1 to n) = n(n+1) /2 is true to 2 to k, for some value of k

For n = k+1
Left hand side, we have
We have sum(i, for i=1 to k+1) = sum (i, for i=1 to k) + (k+1)
= k(k+1) / 2 + (k+1)
= [k(k+1)+2(k+1)] /2
= [(k+1)(k+2)]/2

Right hand side, we have
(k+1) (k+1+1)/2 = (k+1)(k+2)/2
=[(k+1)(k+2)]/2

Left hand side = right hand side

Therefore, sum(i, for i=1 to n) = n(n+1) /2 is true for n = k+1

By mathematic induction, the result is true for all n.

4)

$64n\log(n) < 8n^2$

$64 \log(n) < 8n$

8 log(n) < n

n – 8log(n) > 0

Solving this equation using python,

```
sym.init_printing()
x = sym.symbols('x')
f = sym.Eq(x - 8*sym.log(x), 0)
result = sym.solve(f, x)
print(result) → [-8*LambertW(-1/8), -8*LambertW(-1/8, -1)]
```

(1.1553708251000778, 26.0934854766119)

Assuming n is an integer,

Algorithm-2 performs better than algorithm-1 when n ={ 1 and [27, inf] }

Check
N=0 → T1(0) = 0 T2(0) = 0
N=1 → T1(1) = 8 T2(1) = 0
N=27 → T1(27) = 5832, T2(27) = 2473