1. (20pt) Using the $c - n_0$ definitions of the asymptotic notations to answer the following questions:

   (a) Let $f(n) = 2n$, is $f(n) = O(n)$? Why or why not?

   (b) Let $f(n) = 2n$, is $f(n) = \Omega(n)$? Why or why not?

   (c) Let $f(n) = 2n$, is $f(n) = \Theta(n)$? Why or why not?

   (d) Let $f(n) = 2n$, is $f(n) = o(n)$? Why or why not?

   (e) Let $g(n) = n$, is $g(n) = O(n^2)$? Why or why not?

   (f) Let $g(n) = n$, is $g(n) = o(n)$? Why or why not?

Answer:

(a) f(n) = O(g(n)) = O(n) is true.
Use c = 2 and n_0 = 1

(b) f(n) = Ω(g(n)) = Ω(n) is true.
Use c = 1 and n_0 = 1.

(c) f(n) = θ(g(n)) = θ(n) is true.
From (a) and (b), this is obvious.

(d) f(n) = o(g(n)) = o(n) is false.
If f = o(g), then for every c > 0, we will need to find an n_0 such that for all n >= n_0, f < cg.
Observe that if c = 1, f > cg for any n_0. Thus f is NOT o(g).

(e) true. Use c = 2 and n_0 = 1.

(f) False.
Observe that if c = 0.5, f > cg for any n_0. Thus f is NOT o(g).

2. (20pt) Sort the following functions based on their asympotic growth rate with brief explanations. $f(n) = 10^{-10}$, $g(n) = 10^{10}$, n^2, $(\log_2 n)^2$, 2^n, 3^n, $n \log_2 n$, $\log_2 n$, $\log_3 n$, $2^{\log_2 n}$, $(\sqrt{2})^{\log_2 n}$, $(\log_2 n)^{\log_2 n}$, $n^{\log_2 n}$, $\sqrt{n}$, $\log_2 \sqrt{n}$, $\log_2 (\log_2 n)$, $n^n(1 + (-1)^n)$.

Answer: here, n^n(1 + $(-1)^n$) oscillating function and can't be quantified using asymptotic notations.

- f(n) = 10^{-10}, g(n) = 10^{10}
- $\log_2(\log_2 n)$
- $\log_3 n$, $\log_2 \sqrt{n}$, $\log_2 n$
- $(\log_2 n)^2$

- $\sqrt{n}$, $(\sqrt{2}^{\log_2 n})$
- $2^{\log_2 n}$
- n $\log_2 n$
- n^2
- $\log_2 n^{\log_2 n} = \left(2^{log_2 log\, _2 n}\right)^{\log_2 n} = 2^{(log_2 log\, _2 n)(\log_2 n)}$
- $n^{\log_2 n} = 2^{(\log_2 n)^2}$
- 2^n
- 3^n

3. (10pt) Consider a special type of tree data structure called the **Short Tree**, which is defined as follows.

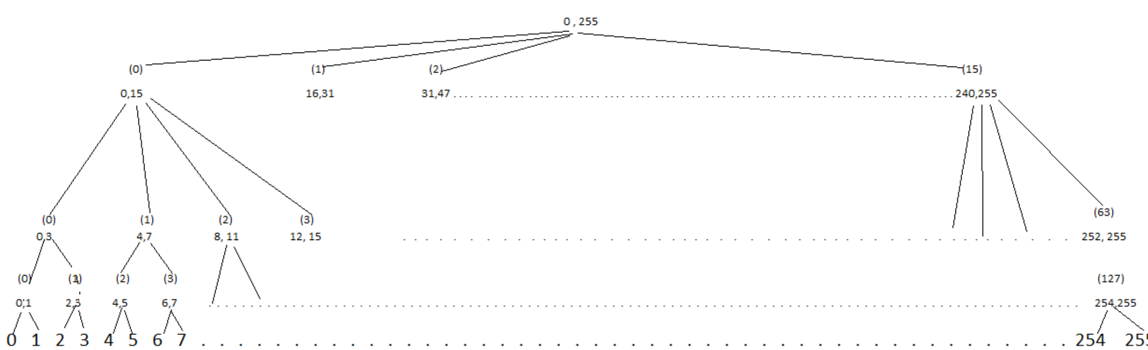
   - Each leaf node of a *Short Tree* is associated with a distinct key, i.e., no two nodes of the tree have the same key.
   - Each non-leaf node of a *Short Tree* is associated with a pair of keys, v_{min} and v_{max}, indicating the smallest key v_{min} and the largest key v_{max} in its sub-tree.
   - Let v be an aibitrary node of a *Short Tree*. Let m be the total number of leaf nodes in the sub-tree rooted at v. For ease of explanation, let the keys be $\{k_0 < k_1 < k_2 < ... < k_{m-1}\}$. Then v has $\Theta(\sqrt{m})$ children. Let the children be indexed $0, 1, ..., \sqrt{m} - 1$, then the $j-$th child is responsible for storing the keys within the range $\{k_{j\sqrt{m}}, ..., k_{(j+1)\sqrt{m}-1}\}$, and has a *min* of $k_{j\sqrt{m}}$ and a *max* of $k_{(j+1)\sqrt{m}-1}$. Note that the above definiion is applied recursively to the children of v.

   Answer the following questions:

   (a) Let the keys of a particular *Short Tree* be $0, 1, 2, ..., 255$. What is the height of the tree?

   (b) What is the assmptotic height of a *Short Tree* with a total of n keys $\{0, 1, ..., n-1\}$?

Answer:



Let n=2^{2^k}

- ⇨ $\log_2 n = 2^k$
- ⇨ $k = \log_2\log_2 n$

We get the recurrence relation,

$T(n) = T(\sqrt{n}) + 1$

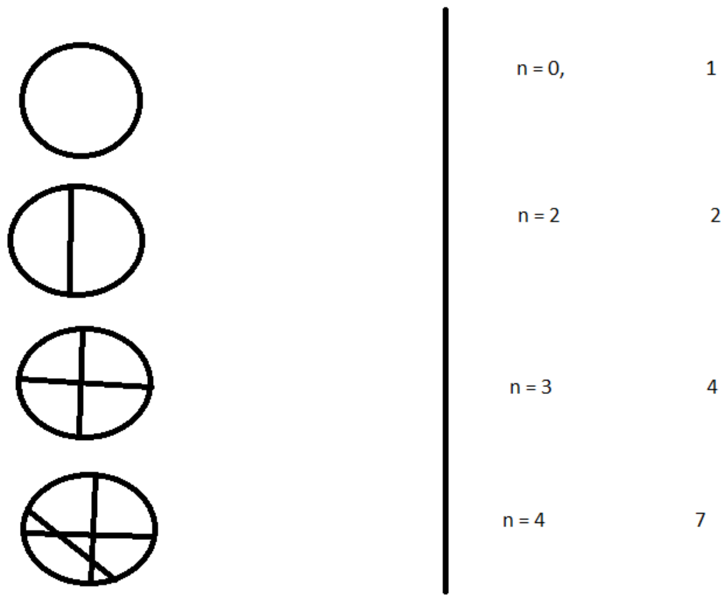
$= T(\sqrt{2^{2^k}}) + 1$
$= T(2^{2^{k-1}}) + 1$
$= T(2^{2^{k-2}}) + 2$

= ……………………….
$= T(2^{2^{k-k}}) + k$
$= T(2) + k$
$= 1 + \log_2\log_2 n$

(a) Height of the tree = $1 + \log_2\log_2(256) = 1 + 3 = 4$
(b) Asysmptotic Height of the tree = $\log_2\log_2 n$

4. (10pt) In computational geometry, an arrangement of lines is the partition of the plane formed by a collection of lines. Observe that the lines partition the plane into disjoint regions. Calculate the maximum number of disjoint regions in an arragement created by n lines.

Answer:



We get, $T(n) = T(n-1) + n$

$= T(n-2) + (n-1) + n$

$= T(n-3) + (n-2) + (n-1) + n$

$= T(n-4)+ (n-3) + (n-2) + (n-1) + n$

$= \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$

$= T(n-n) + 1 + 2 + \ldots\ldots +(n-3) + (n-2) + (n-1) + n$

$= T(0) + 1 + 2 + \ldots\ldots +(n-3) + (n-2) + (n-1) + n$

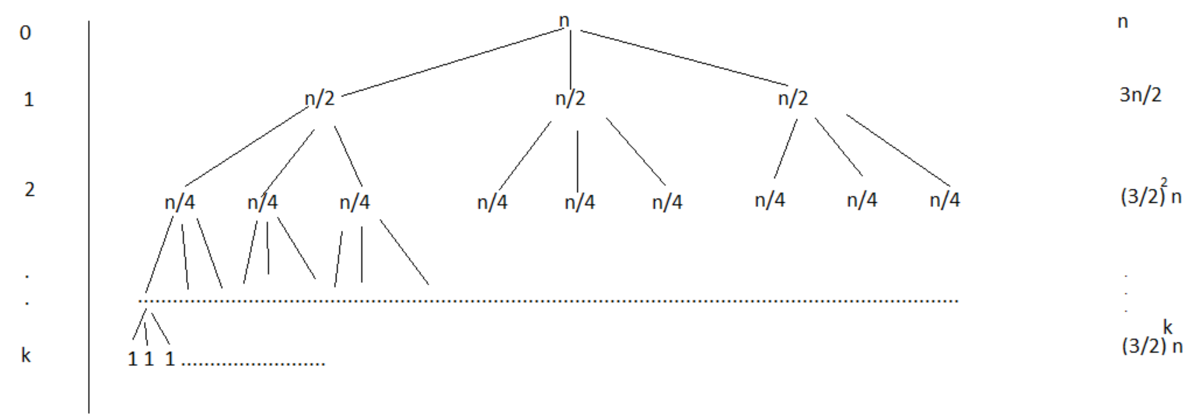
$= 1 + n(n+1)/2$

So, the maximum number of disjoint regions created by n lines = 1 + n(n+1)/2

5. (10pt) Use the recursion tree method to determine a good upper bound on the recurrence

$$T(n) = 3T\left(\lceil \frac{n}{2} \rceil\right) + n$$

Answer:



At kth level n = 1,
Then $n/2^k = 1$, $k = \log_2 n$.

From the tree, we get

$$n + 3/2\, n + (3/2)^2 n + (3/2)^3 n + (3/2)^4 n + \ldots\ldots + (3/2)^k n$$
$$= n \{ 1 + 3/2 + (3/2)^2 + (3/2)^3 + (3/2)^4 + \ldots\ldots + (3/2)^k$$
$$= n \{ ((3/2)^k - 1) / (3/2 - 1)\}$$
$$= n \{ ((3/2)^k - 1) / (1/2)\}$$

$$= 2n\, \{(3/2)^{\log_2 n} - 1\}$$
$$= (3/2)^{\log_2 n}. 2n - 2n$$

$$= (3^{\log_2 n}/2^{\log_2 n})2n - 2n$$
$$= (3^{\log_2 n}/n)2n - 2n$$
$$= (3^{\log_2 n}/n)2n - 2n$$
$$= (3^{\log_2 n})2 - 2n$$
$$= (n^{\log_2 3})2 - 2n$$

Then, T(n) = $O(n^{\log_2 3})$

6. (10pt) Use the guess and substitution method to prove that the T(n) in the following recurrence relation is $O(n)$.

$$\begin{cases} T(n) = T(\frac{n}{2}) + n & \text{for } n \geq 2 \\ T(1) = 1 \end{cases}$$

Answer:

Guess: Suppose we guess the solution to be, T(n) = O(n). This means there exists a positive constant c, such that T(n) ≤ cn for all n sufficiently large (i.e for some positive constant n_0 , $n \geq n_0$)

Basis:

For n = 2

T(2) = T(2/2) +2 = T(1) +2 = 1 + 2 = 3 ≤ c.2, as long as c≥2

Induction Hypothesis:

Assume for 2 ≤ k < n

T(k) ≤ ck is true

Induction Step:

T(n) = T(n/2) + n

≤ [c.n/2] + n

≤ c. n/2 + n

≤ n (c/2 + 1)

For c≥ 2, c/2 +1 ≤ c

Then, we can write T(n) ≤ n (c/2 + 1) ≤ cn is true for c≥2

Therefore , T(n) = O(n)