Today's Content:

→ Tries Basics

→ Trie problems

: prefix Query count

: String Search

: Max XOR pair

: Max Subarray XOR

} Saturday's Session

// String A : 0 1 2 3 4 5 6 7
a n a c o n d a B : a n a
0 1 2

A[0 2] = B

String A : a_0 a_1 a_2 a_3 a_4 a_5 -- a_{n-1} a_n - - a_m

String B : b_0 b_1 b_2 b_3 . _ b_{n-1}

→ O(n) : n is length of B

Given N words & Q queries → 'a' = w[i] < = 'z'

Calculate number of prefix strings of given words = given Query

Substrings starts at index 0

1 < = len(word) < = M

1 < = len(query) < = M → O(M)

Eg:

N = 20

20 words:

data	store
draw	strong
drew	list
dark	linked
algorithm	link
stack	stamp
structure	sound
struct	drunk
drake	dried
damp	almond

Queries: 7

st: 6

if: 3

dr: 5

da: 3

dat: 1

dark: 1

do: 0

Ideas:

for every query iterate on all strings, check if query is prefix or not

TC: Q * [N * M] (to check query is prefix or not for a word)

Q: Queries, N: words

Ideas: LPS: Patter P & Text T (M, M)

we can search if Text T is prefix in P → O(M) *

Ideas: Sort all words in lexicographical (dictionary)

Queries: no: of words have prefix dr

d r

s t r

Idea:

Sort N words of M length +

For Every Query of M → apply Binary search length of Query times

Tc: $(N \log N) * M + Q * M * \log N$

0	algorithm
1	almond
2	damp → P_1
3	dark
4	data
5	drake → P_1
6	draw
7	drew
8	dried
9	drunk → P_2 P_2
10	sound → P_1
11	stack → P_1
12	stamp
13	store
14	strong → P_1
15	strut
16	structure → P_2
17	stud → P_2 → P_2
18	link
19	linked
20	list

// cnt = $P_2 - P_1 + 1$
= 9 - 5 + 1
= 5

// $P_2 - P_1 + 1$
→ 3

Ideas:

→ N children : 26 children :

N-array Tree :

damp

dark

data

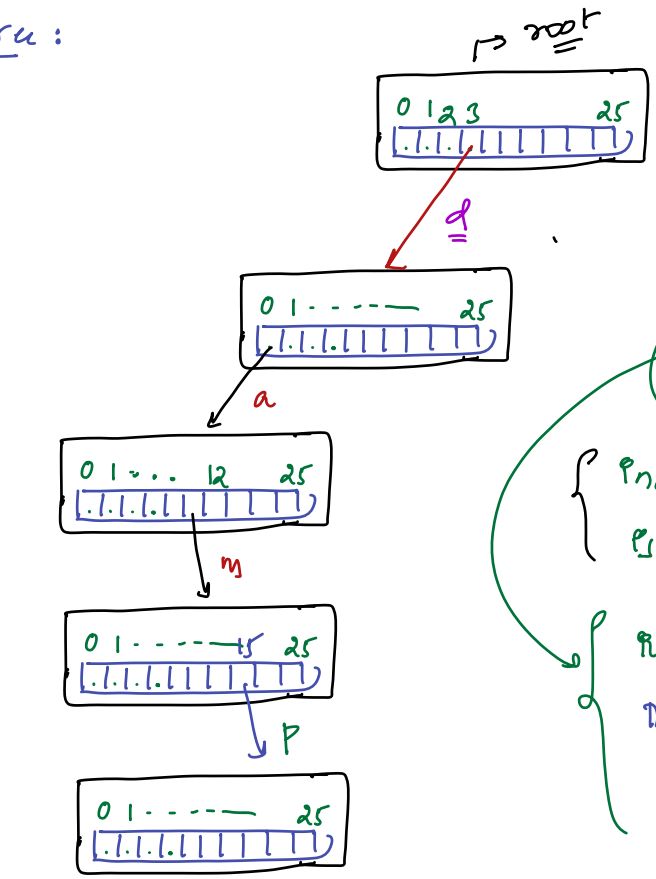
drake

draw

drew

dried

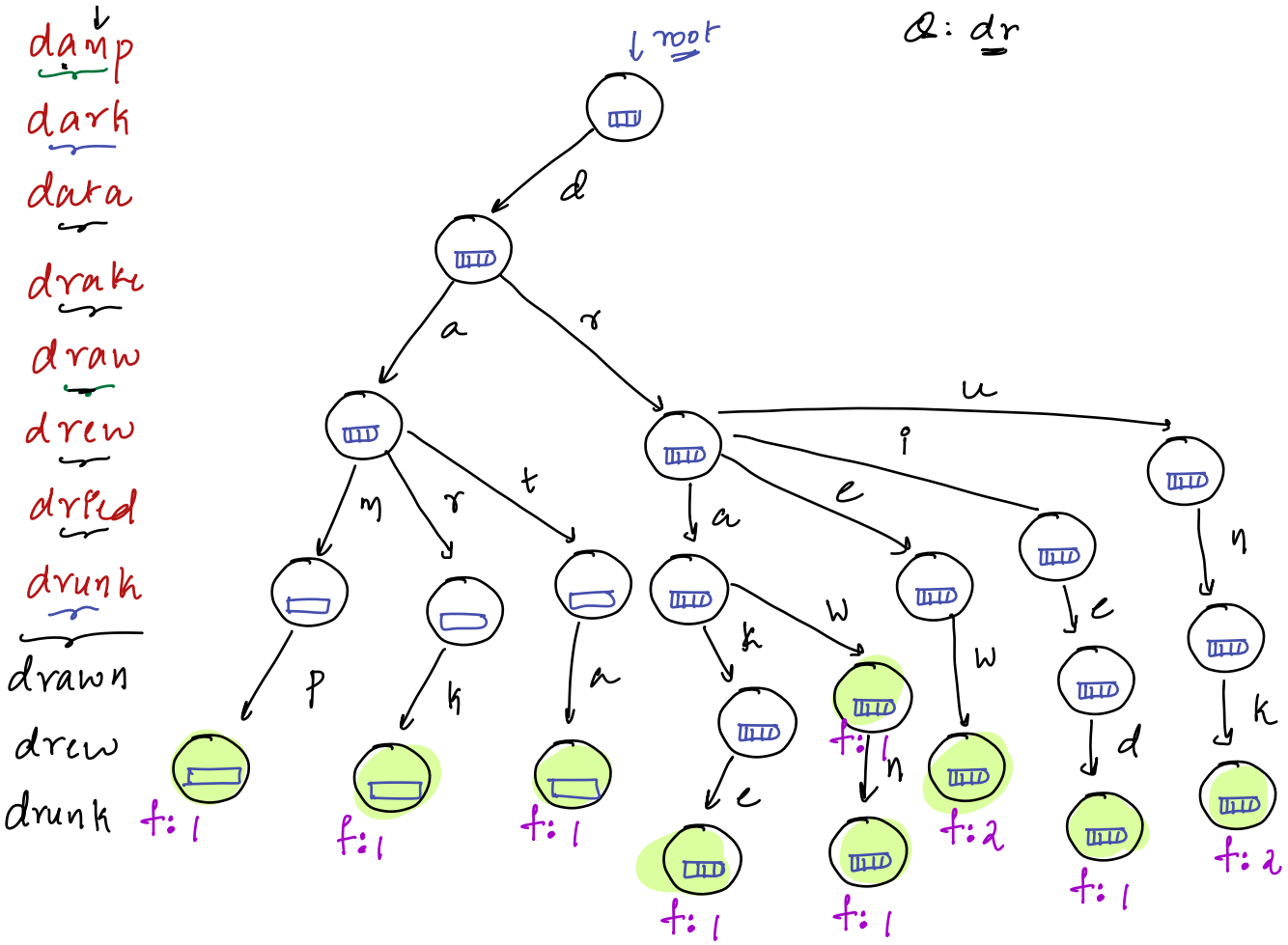
drunk




```
class Node {
    Node c[26]
    // we initialy
    all children
    with NULL
    bool isEnd;
    int count;
};
```

{ Indicates: whether node is ending there or not }

{ Indicates: frequency of the word }



Q: dr

how many words have same prefix dr

→ Traversal from root node to dr : ✗

→ Calculate no: of leaf Nodes present in Subtree

→ Is end in Node Structure : While inserting make last node of word = T

Traversal from root node to dr : ✗

→ Calculate no: of Nodes in Subtr with isEnd = True

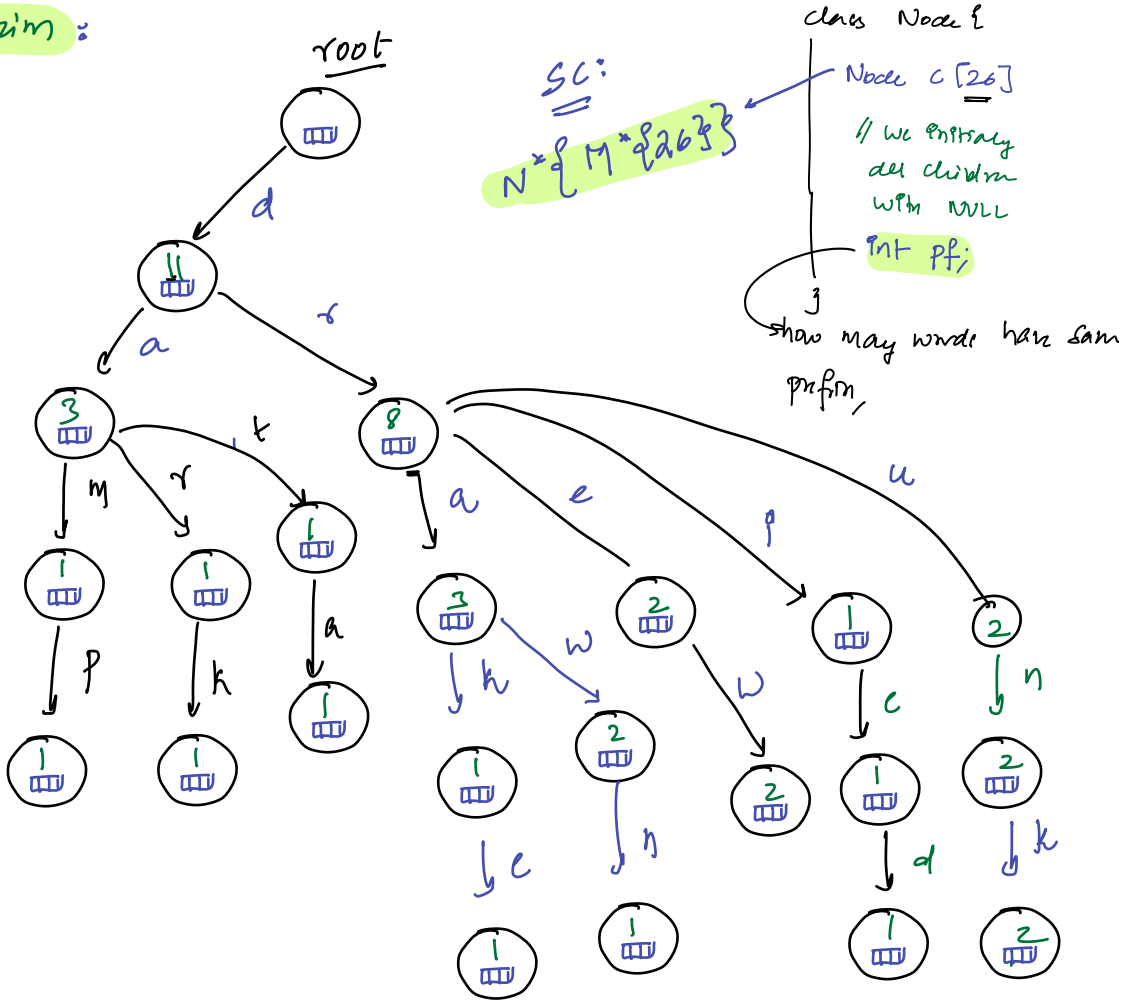
→ count in Node Structure : ✓

Traversal from root node to dr :

→ Calculate freq of an Nodes

Ideas Contribution:

damp ✓
dark ✓
data ✓
drake ✓
draw ✓
drew ✓
dried ✓
drunk ✓
drawn ✓
drew ✓
drunk ✓



Q: dap

Q: dr:

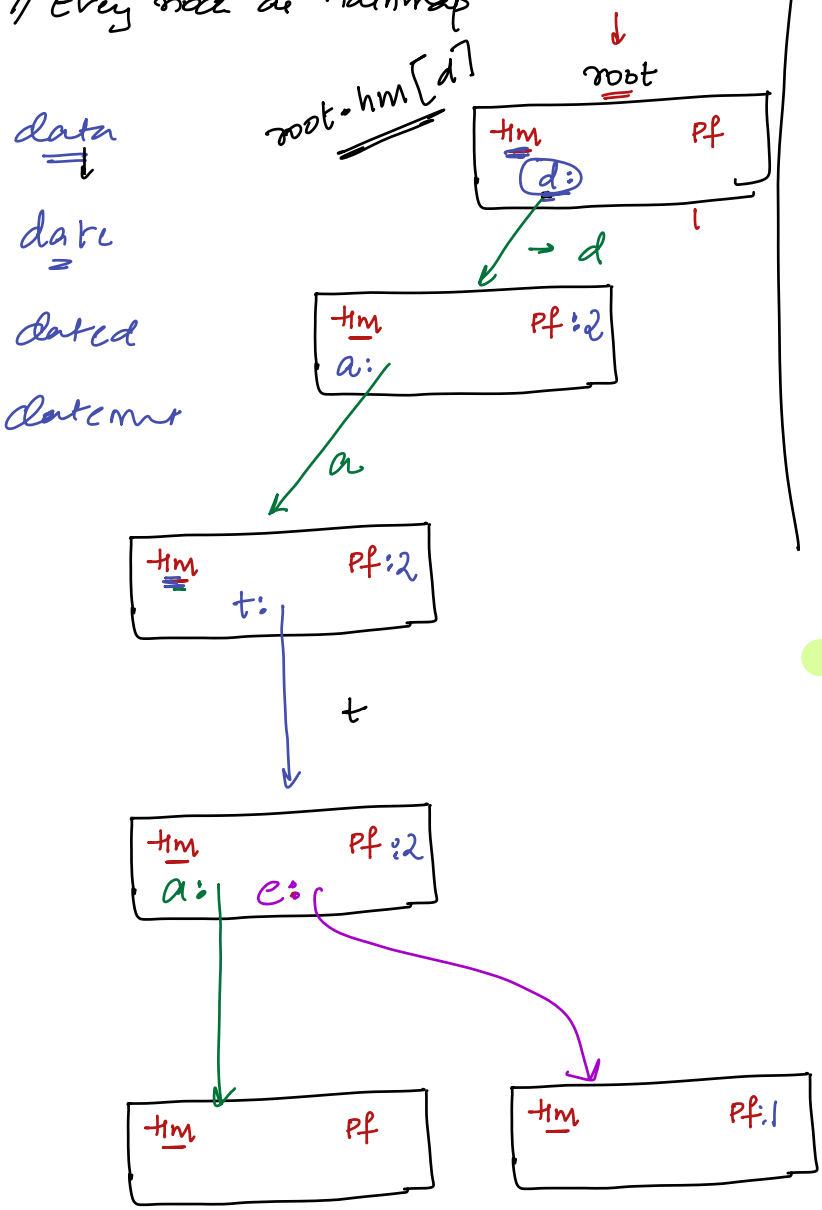
pf: how many words are passing through current Node

Traversal from root node to dr:

→ get pf of current node

Q: da

// Every node as hashmap



data

data

dated

datemr

TC:

m is length of each word

N * M + Q * M

no of words

SC:

O(N * M)

at max M nodes we create for a word

Answer SC ∝ (N * M)

11:03pm

Ignore

Pseudocode:

```
class Node {
    Hashmap <char, Node> hm;
    int pf;
}
```

Rough Pseudo Code

```
Node root = new Node();
Read N;
loop N;
    Read word
    Insert (root, word)
}
Read Q
loop Q
    Read word
    print (query(root, word))
}
```

```
void Insert(Node curr, String w) {
    for i = 0; i < w.length; i++) {
        char ch = w[i];
        if ( ch is not in curr.hm){
            Node t = new Node();
            curr.hm.Insert(ch, t)
        }
        curr = curr.hm[ch] } ✓
        curr.pf + +
    }
}
```

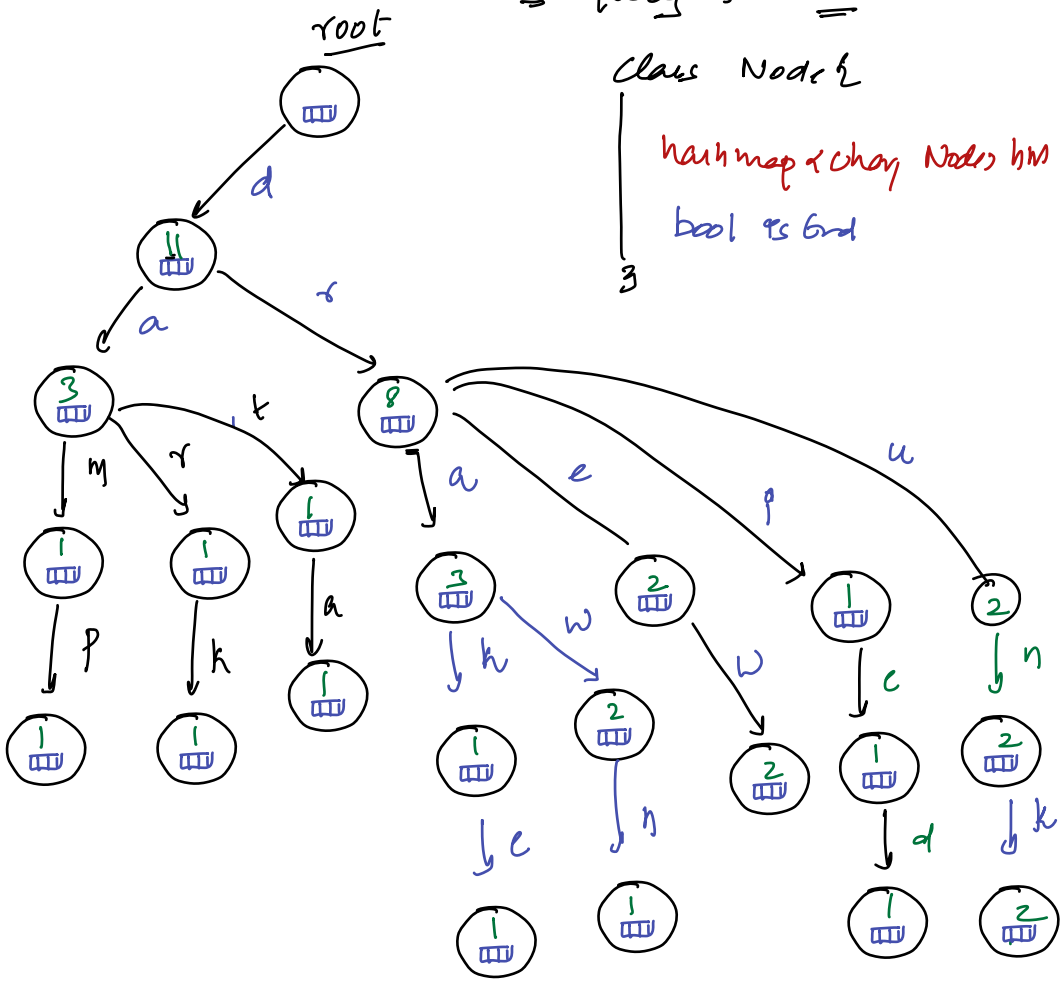
getting addr from hashmap

```
int query (Node curr, String w) {
    for i = 0; i < w.length(); i++){
        char ch = w[i];
        if ( ch is not in curr.hm)
            return 0
        curr = curr.hm[ch]
    }
    return curr.pf;
}
```

28) Given N words

Q queries for each query check if there exists a word as a

damp
dark
data
drake
draw
drew
dried
drunk
drawn
draw
drop



→ query → dre:

```
class Node {
    hashmap<char, Node> hm
    bool isEnd
}
```

pdca:

1) pf count is not at all useful

2) We need isEnd at node, indicating whether a word is ending at that pos.

Saturday : Thr, Thursday | Tue | Thu

Trie | heap1 | Heaps | Heaps

→ TODO:

Trie data structure

(Retrieve) {

(Saturdays)

3(8) Given N Elements find max xr pair → (i, j) → ar[i] ^ ar[j]
max

N=4 : 0 1 2 3 } ans = ?
5 3 2 7

Idea:

Input:

Ex:

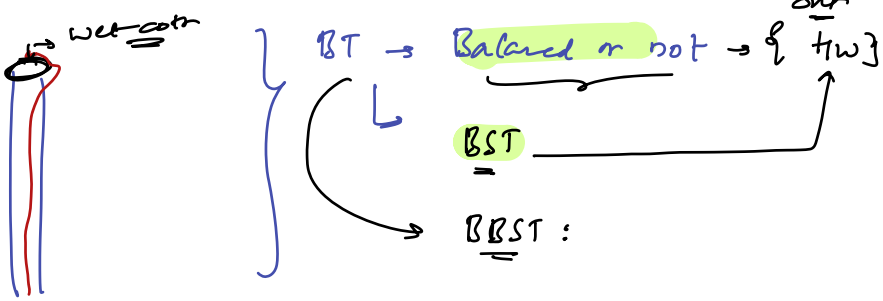
N=9 : 0 1 2 3 4 5 6 7 8
22 61 38 27 21 34 42 37 43

/ Ipad → Notability → paid → { paid } / 1k for notability
→ unpaid : { 3 colours }

Touchpad → (Wacom) → 3000

↳ you can write note on 1 note

→ pen/paper → Red/ green/ black pen



google / Facebook / Amazon

↳ Trie