Tries

05 March 2023 17:57

Tries :-

- Problem - Given a list of strings, and many queries, in each query we have a string **word**, and we have to tell the number of the strings in which **word** comes as a prefix.
- brute
- better (sorting)
- better (tree)
- Hint (tries)
- Let's generalise this
- Inserting a word
- Searching for a word
- Searching for count of words
- Searching for count of prefixes
- Deleting a word
- Code
- Let's see the problem mentioned at the top
- Tries on Numbers - Standard Thing
- New problem - given a list of positive integers, find the maximum xor of any two numbers. (Maximum XOR of Two Numbers in an Array)
- Code
- Bonus problem - Sum of Prefix Scores of Strings

list of strings

queries

string word

↓

no. of string inwhich word is a prefix

brute

N

cnt++;

sorting

target tattoo

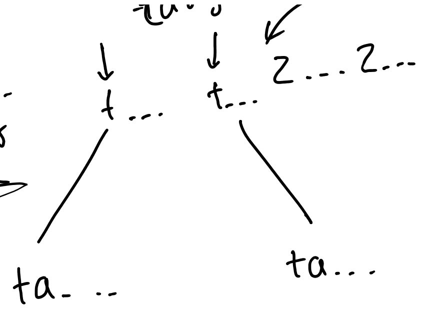
a a... ↓ ↓ 2...2...

sorting

X

a_ _a...

logN * no. of character of word



tree

BST

~~int~~

string

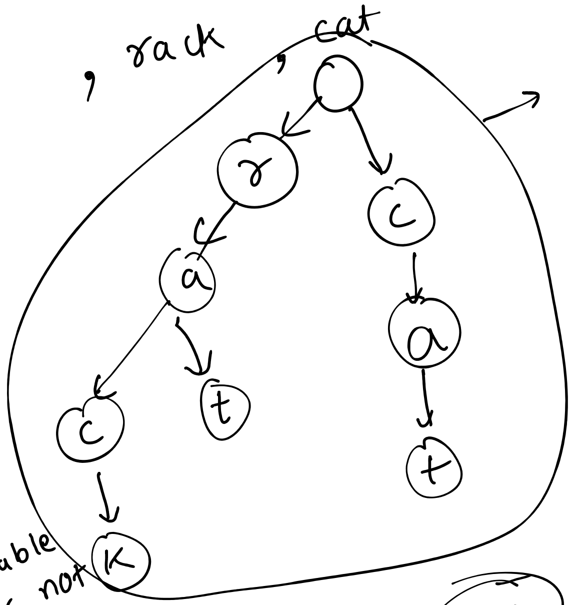
pbds → ordered set

~~search...~~

hashing

Tries

rat , rack , cat



insert ✓

check if it is available or not

Tree

rat

rac

ra

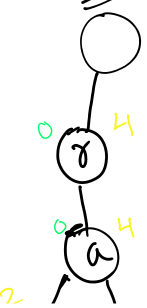
rat

rat

Tree

rac

rack



count

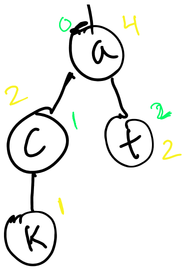
Node

struct {
char ch;
End;

green

N

```
struct {
    char ch;
    int cntEnd; green
    int cntStart; yellow
}
```



rac...

insert (word) → void

count (word) → int

startsWith (word) → int

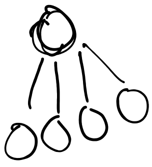
```
struct {
    int ~~cntStart~~ → no. of times
    int cnt End  →   I went through
                     this node
                     while inserting
                     some word
}
```

```
Struct TrieNode {
        int  cntStart = 0;
        int  cnt End = 0;
        TrieNode *  children[26] ;
                          ↓
                         NULL
}
```

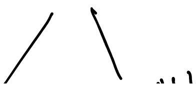
lowercase

a, - - -, z

a



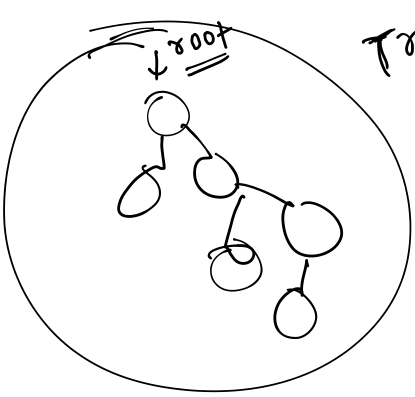
Binary



cntStart
cntEnd
26

DFS

26
NULL
NULL
NULL



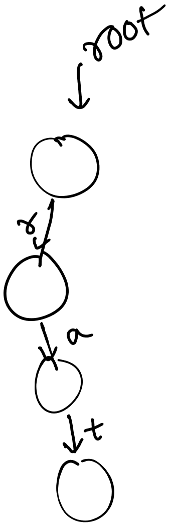
Trie → root

- insertion (word)
- count (word)
- startsWith (word)

YES/NO

{ check if node exist
if not, then creat
go to that node
cntStart++;

{ cntEnd++;



rat

2-3

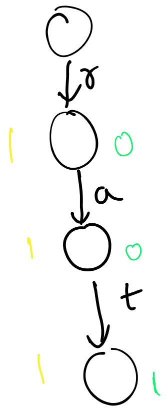
~~200-300 memory~~

insert rat

delete ~~rat~~k

Data Structure

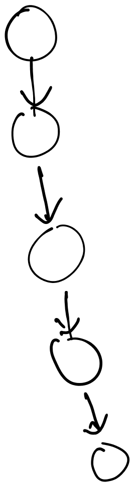
simplest way



erase {

①

```
erase (word {
    if (count(word) == 0)     return;
}
```



Homework

- stack
- if (cntstart == 0)
  delete node;

list of strings

```
Trie trie;
for   each string word
        trie.insert(word);
for each query :-
        trie.startsWith(word);
```

Tries on Numbers

int digit

char int digit

5 7 2

Binary form

17

10001

→

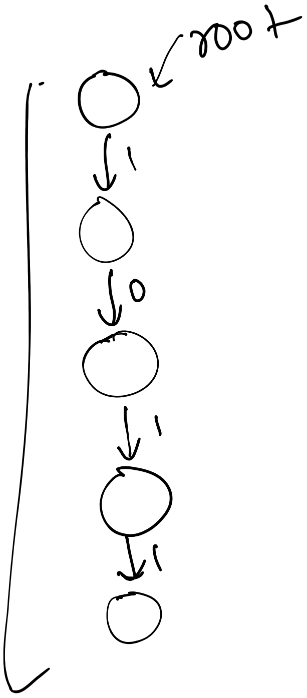
11

1011

3 11

1011

should I store a no. like this



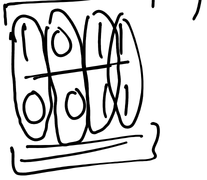
32 bits

1e17

64 bits

11 → 1011

3 → 11



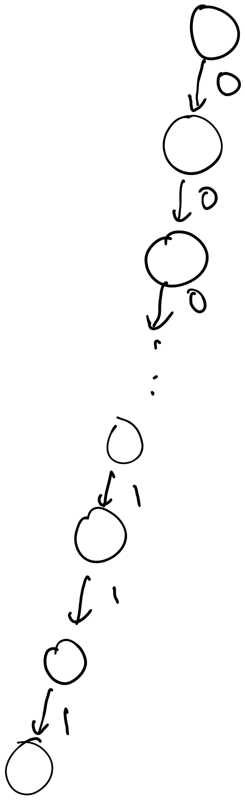
$\underbrace{\leq 1e9}$ → 32

=

7

$\underbrace{\underbrace{000\ldots\ldots0}111}_{32 \text{ bits}}$

17



number

$\boxed{000\ldots1101}$ ↓ binary no. of 32 bits

Array of integers

— — — — — — — — — —

xor

n^2

11 22 27 13 - -

↓

1011

Tries

$\boxed{0\ldots\ldots01011}$

$0\ldots\ldots110110$

^1 = 1

0....10110

$0 \wedge 1 = 1$

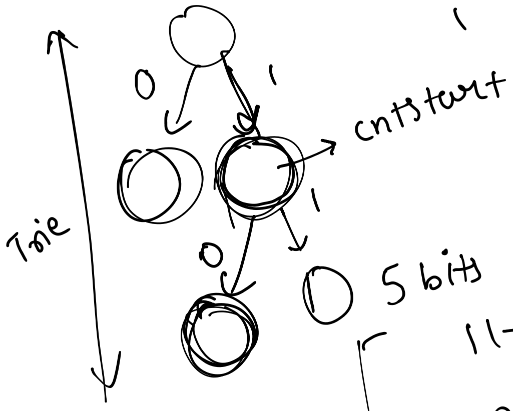
$1 \wedge 0 = 1$

$0 \wedge 0 = 0$

$1 \wedge 1 = 0$

11→ "000.....01011"

22→ "000 ---10110"



5 bits

11→ "01011"

22 → "10110"

27 → "11011"

dfs

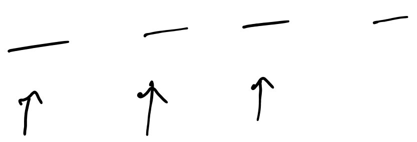
cool

1e17 1e18

64

M → 1e9

N integers



TC → O(N logM)

S.C → O(N or N logN)

Homework

1. number → string
not hard

010

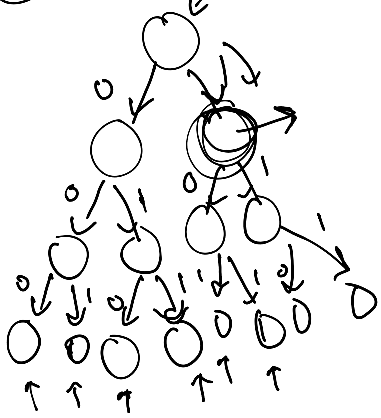
2. Trie → insert

3. iterating over every element is not hard either

4. integer maxXor

④. integer maxXor either

key
↓
010 (key^1) ← node challenging part



Striver's first video Tries

Live coding

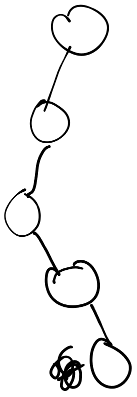
struct Node{
~~cnt~~ cnt
children[2]
};

cntstart

~~cntend~~

~~cnt~~

Trie



list of strings
↓
for every string

score →

"word"

"w" + "wo" + "wor" + "word"