Lecture26

Friday, 29 May 2020 6:52 PM

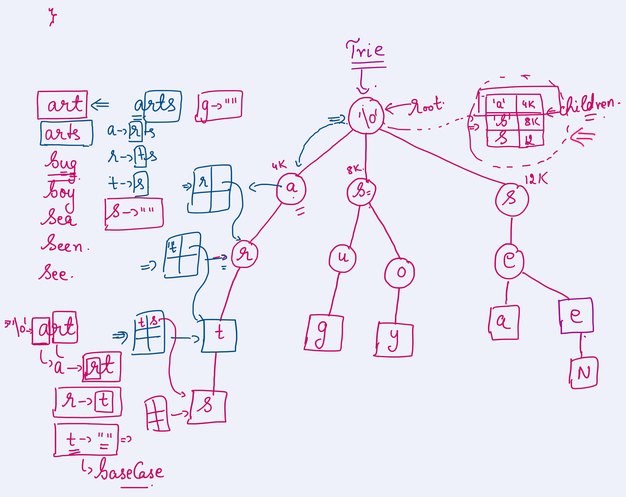
Tries → text processing.

Aman
Amit
Amrit
- - - -

→ Am

Tries is also a kind of tree.
that can have any number of children.

```
Node {
 { Character data;
 { boolean isterminal;
 { Hashmap < Character, Node > children;

}
```

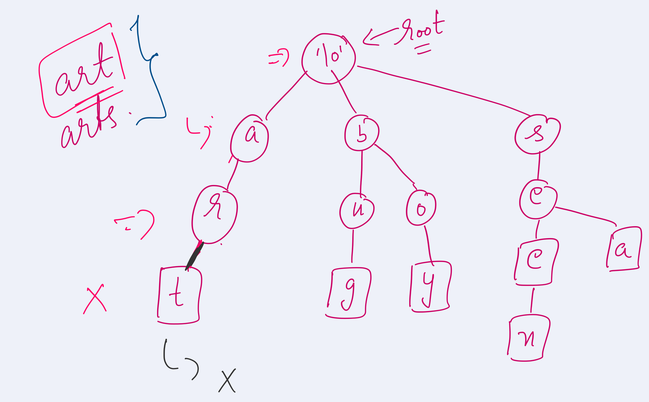



```
class Trie {
    Node {
        data
        children = new HashMap
        isterminal -> true -> last node of my word.
    }
    Node root;
    int numwords;
    Trie.() {
        => root = new Node ('\0', false); =>
        numwords = 0;
    }
}
```

```
public void addword (String word) {
    this.addword (root, word);
}

private void addword (Node node, String word) {
    if (word.length() == 0) {
        if (node.isTerminal) {
            } return true. ⇐
        } else {
            numwords++;
            node.isterminal = true;
            } return false; ⇐
        }
    }
    char cc = word.charAt(0);          // d r  -> a
    String ros = word.Substring(1);    // rt   -> rt
    Node child = node.children.get(cc); // null or getanode. ⇐ a
    if (child == null) {
        child = new Node (cc, false);  // d r ⇐
        node.children.put (cc, child); ⇐   } [return false;]
    }
    this.addword (child, ros);  a -> => Recursion ⇐
}
```

SearchWord

```
public void display() {
    this.display (this.root, "");
}

private void display (Node node, String osf) {
    if (node.isTerminal) {
        Syso (osf);
    }
    Set<Map.Entry<Character, Node>> entries = node.children.entrySet();
    for (Map.Entry<Character, Node> entry : entries) {
        this.display (entry.getValue, osf + entry.getKey());
    }
}
```



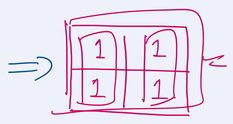

```
Remove Word (String word) {
    this.removeWord (this.root, word);
}

private void removeWord (Node node, String word) {
    if (word.length() == 0) {
        if (node.isTerminal()) {
            node.isterminal = false;
        } else {
            return;
        }
    }
    char cc = word.charAt(0); = a
    String ros = — — (1) => rts
    Node child = node.children.get(cc);
    if (child == null) {
        return;
    }
    this.removeWord (child, ros);
    => if (!child.isTerminal && child.children.size() == 0) {
        node.children.remove(cc).
    }
}
```

node	word	cc	ros	child
t	s	s	""	s
r	ts	t	s	t
a	rts	r	ts	r
'\0'	arts	a	rts	a

t → s

X

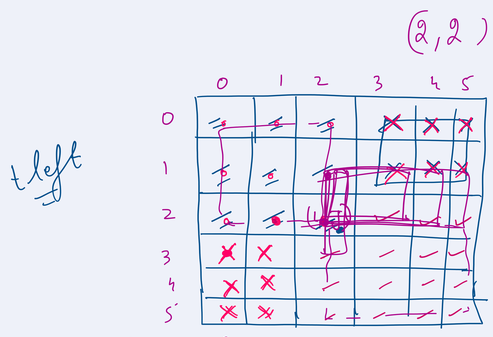
Q ⇒

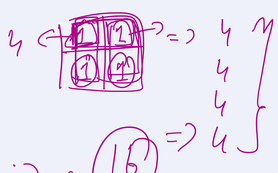


Sum of all the Submatrices?

Size → 1x1 => 4 => 1+1+1+1 => 4.
Size → 1x2 => 2 => 2+2 = 4
Size → 2x1 => 2 => 2+2 = 4
Size = 2x2 => 4 = 4 => 4

16 Ans





(6-i) * (6-J)
(6-2) * 6-2
(4 * 4) = 16

Contribution of each cell in total.
↳ Sum of all Submatrices

n = 6.

```
tl = (i+1) * (J+1).      -> for (i = 0 -> n) {
                            for (J = 0 -> n) {
br = (n-i) * (n-J);
Sum = Sum + tl * br * arr[i][J];
}
```

O(n^6) ⇓ O(n^2)

Tomorrow -> Graphs, D.P.

Sunday X
Saturday -> tomorrow -> 30/5/2020. -> } => D.P
Monday -> 1/6/2020
Tuesday -> 2/6/2020 } ->
↳ 29 lec

t → s