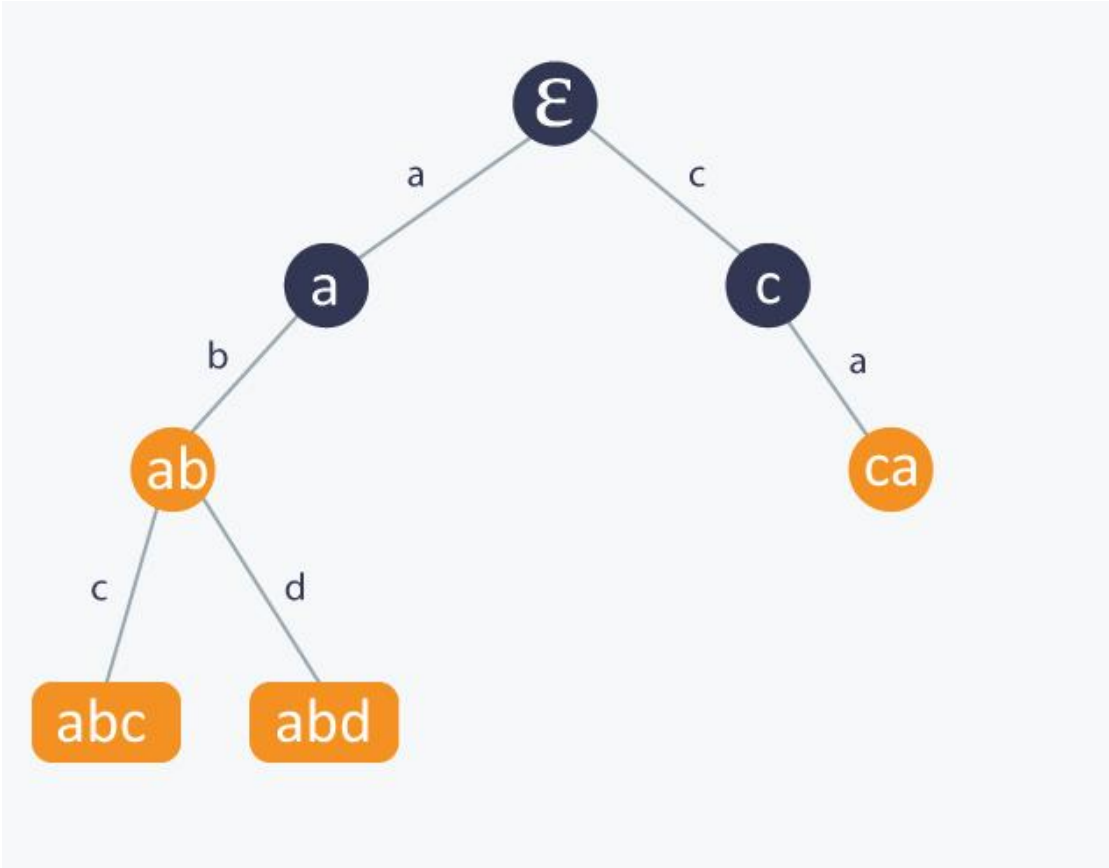
SUFFIX TREE

- In computer science, a **suffix tree** is a compressed trie containing all the suffixes of the given text
- Suffix trees allow particularly fast implementations of many important string operations.

Trie

- Trie is probably the most basic and intuitive tree based data structure designed to use with strings.
- Let S be a set of k strings, in other words S = {s1, s2, ..., sk}
- We can model the set **S** as a rooted tree **T** in such a way, that each path from the root of **T** to any of its nodes corresponds to a prefix of at least one string of **S**.
- Let **S = {abc, abd, ca, ab}**. Let **ε** corresponds to an empty string. Then a trie for **S** looks like this:



Suffix tree

- A suffix tree **T** is a natural improvement over trie used in pattern matching problem, the one defined over a set of substrings of a string **s**.
- A Suffix Tree for a given text is a compressed trie for all suffixes of the given text.

How to build a Suffix Tree for a given text?

As discussed above, Suffix Tree is compressed trie of all suffixes, so following are very abstract steps to build a suffix tree from given text.

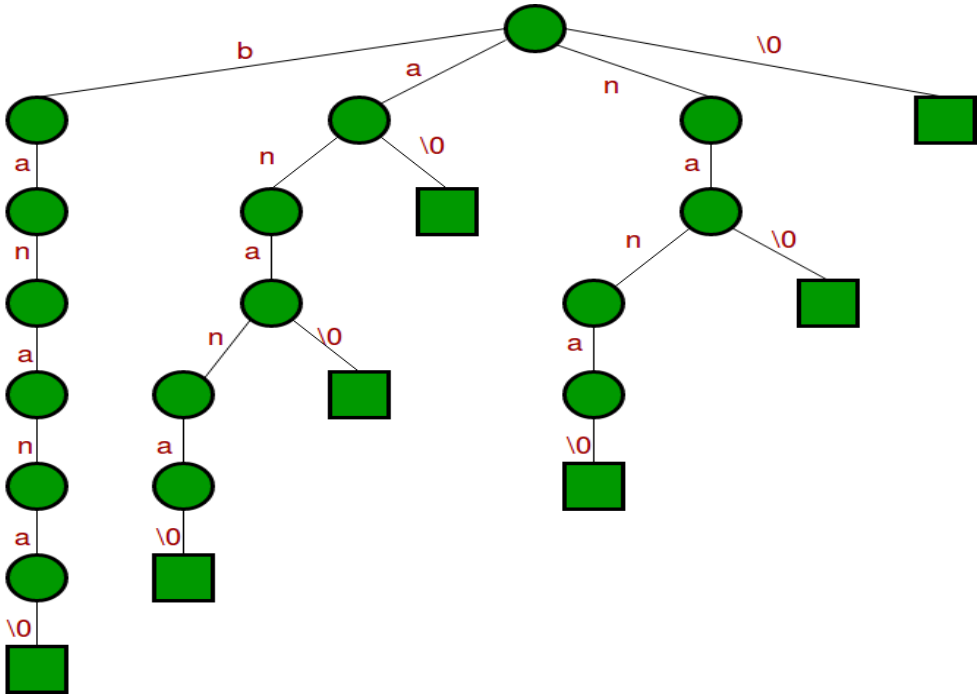
1) Generate all suffixes of given text.

2) Consider all suffixes as individual words and build a compressed trie.

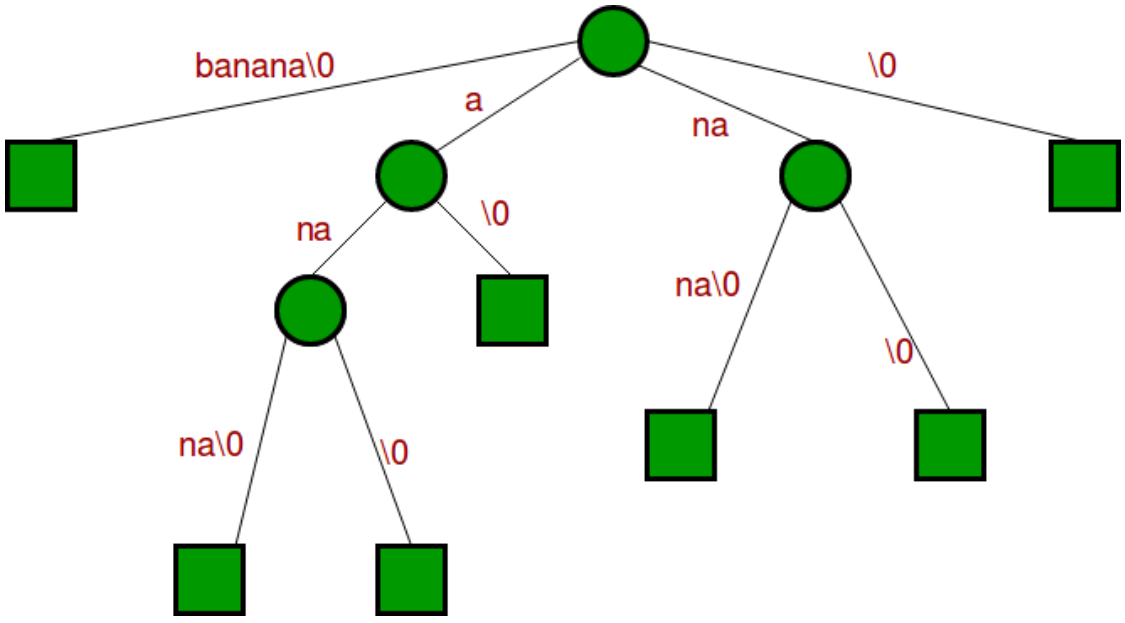
- Let us consider an example text "banana\0" where '\0' is string termination character.
- Following are all suffixes of "banana\0"

1. banana\0
2. anana\0
3. nana\0
4. ana\0
5. na\0
6. a\0
7. \0

- If we consider all of the above suffixes as individual words and build a trie, we get following.



- If we join chains of single nodes, we get the following compressed trie, which is the Suffix Tree for given text “banana\0”



The suffix tree for the string S of length n is defined as a tree such that

- The tree has exactly n leaves numbered from 1 to n.
- Except for the root, every internal node has at least two children.
- Each edge is labelled with a non-empty substring of S.
- No two edges starting out of a node can have string-labels beginning with the same character.
- The string obtained by concatenating all the string-labels found on the path from the root to leaf i spells out suffix S[i...n], for i from 1 to n.

Applications of Suffix Tree

- Suffix tree can be used for a wide range of problems.
- Following are some famous problems where Suffix Trees provide optimal time complexity solution.

  1) Pattern Searching

  2) Finding the longest repeated substring

  3) Finding the longest common substring

  4) Finding the longest palindrome in a string