Introduction to suffix trees

By Vipin Menon

Suffix Trees

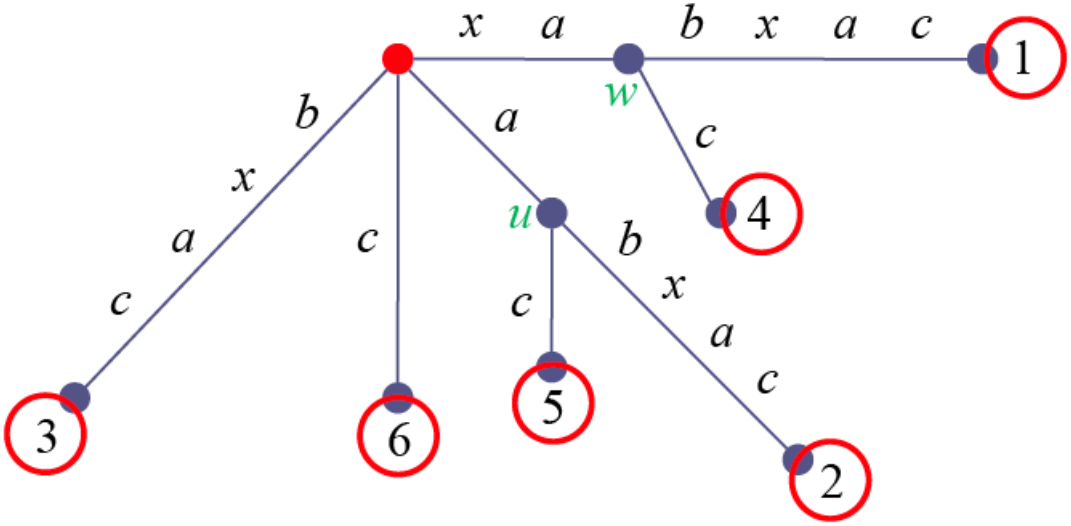
- Suffix tree is data structure that exposes the internal structure of a string in a deeper way than any fundamental preprocessing

- Mostly suffix trees are designed for solving substring problem , and had a varied application in fields like bioinformatics , computer science and other interdisciplinary fields.

- The first linear algorithm was given by Weiner [1973] improved by McCreigtht . Ukkonen developed the linear algorithm by adding certain more features as well as providing simple explanation

Definition:

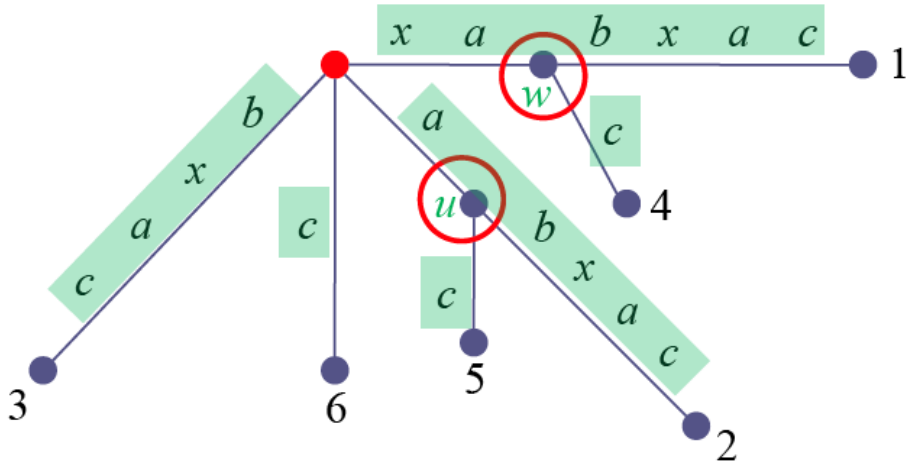
A suffix tree (T) for an m-character string S ,is a rooted directed tree with exactly n leaves numbered 1 to m



Suffix tree for string ‘*xabxac*’.

Definition:

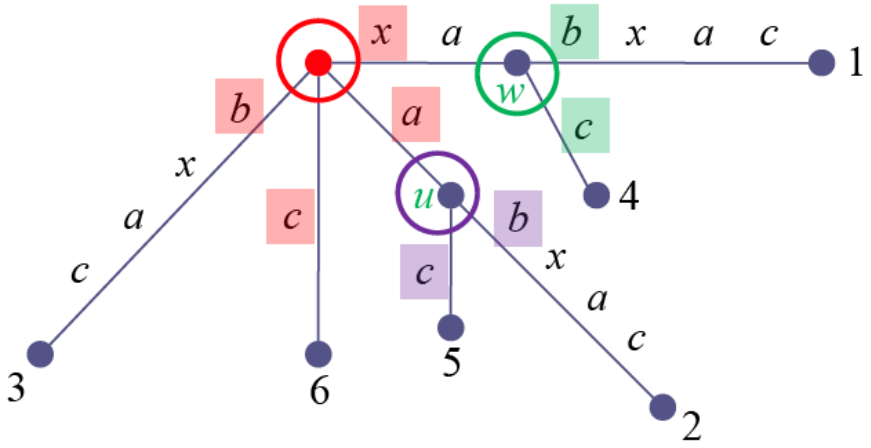
Each internal node, other than the root, has at least two children and each edge is labeled with a nonempty substring of S.



Suffix tree for string '*xabxac*'.

Definition:

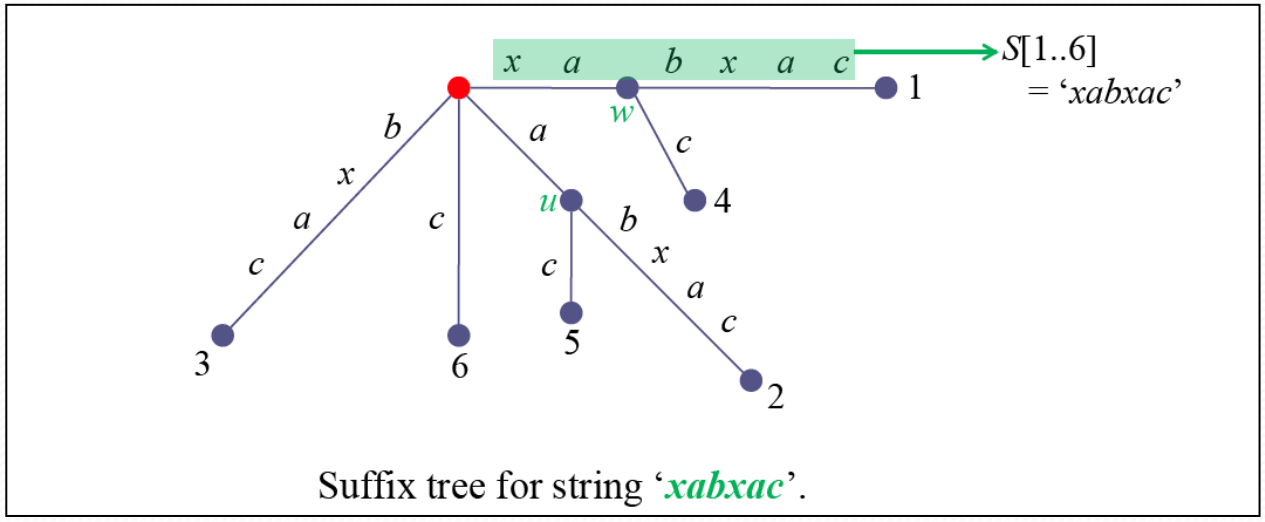
No two edges out of a node can have edge-labels beginning with the same character



Suffix tree for string '*xabxac*'.

Definition:

The key feature of the suffix tree is that for any leaf i , the concatenation of the edge-labels on the path from the root to leaf i exactly spells out the suffix of S that starts at position i. That is, it spells out S[i..m].

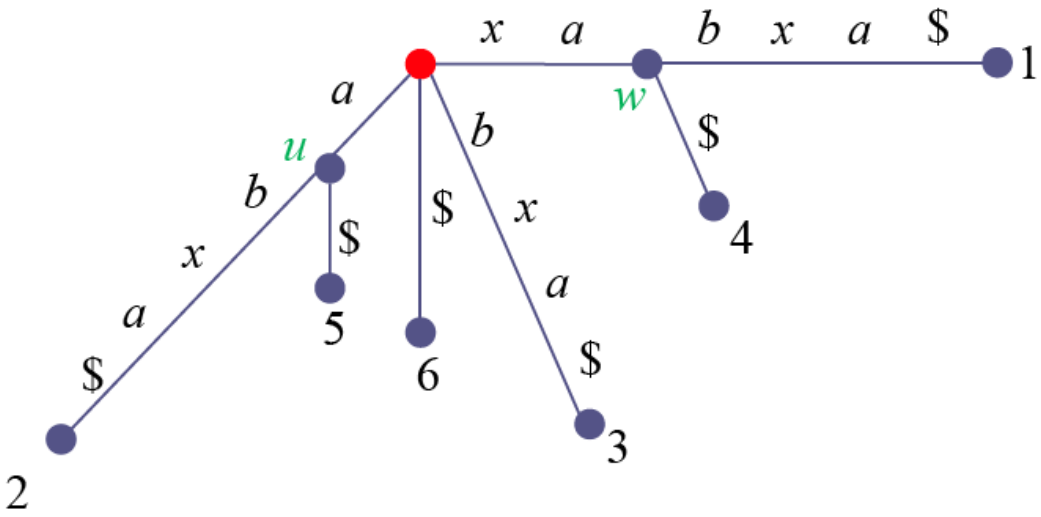


Suffix tree for string '***xabxac***'.

Suffix Tree

Definitions:

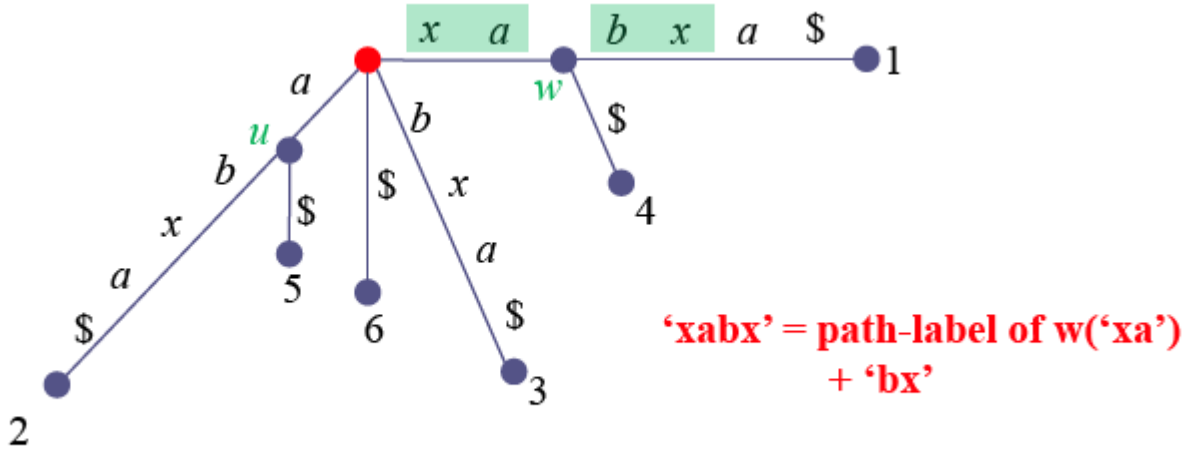
- The label of a path from the root that ends at a node is the concatenation, in order, of the substrings labeling the edges of that path.
- The path-label of a node is the label of the path from the root of T to that node.



Suffix tree for string '*xabxa*$'.

Definition:
A path that ends in the middle of an edge (u, u) splits the label on (u, v) at a designated point.

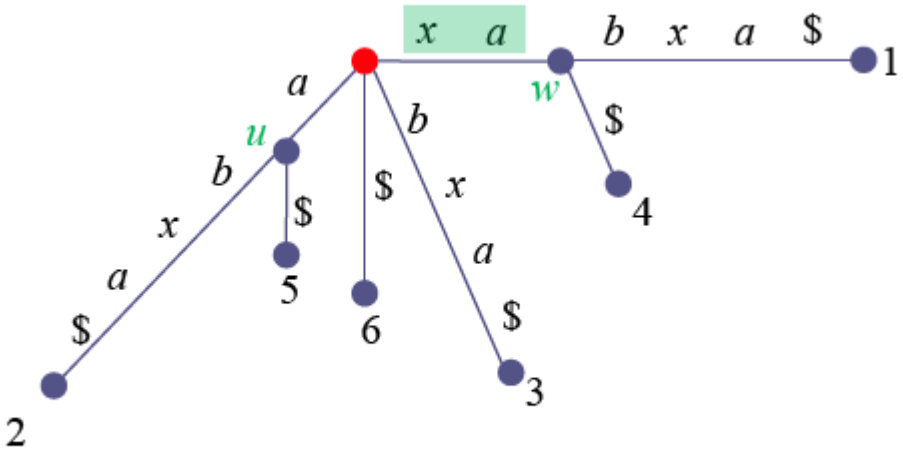
Define the label of such a path as the label of u concatenated with the characters on edge (u, v) down to the designated split point



Definitions:

For any node v in a suffix tree, the string-depth of u is the number of characters in u's label. In simpler words string depth is the largest common prefix between two suffix.

Example : string depth of node ‘w’ is 2 (‘xa’)



Suffix tree for string ‘*xabxa*$’.

Suffix Tree

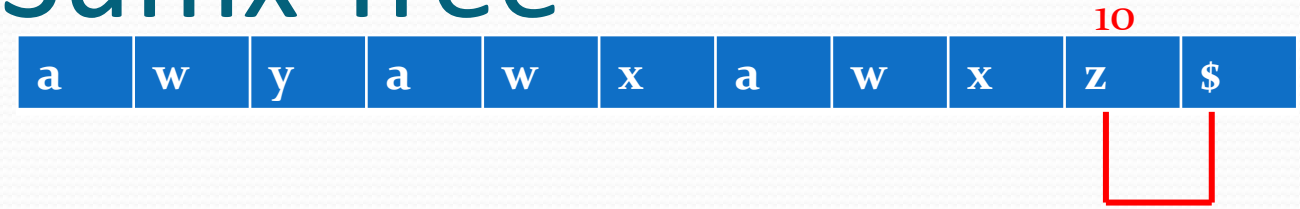
String = n

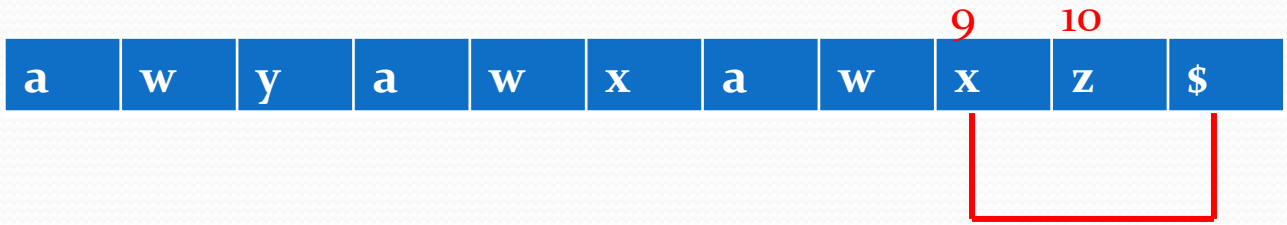
a	w	y	a	w	x	a	w	x	z	$

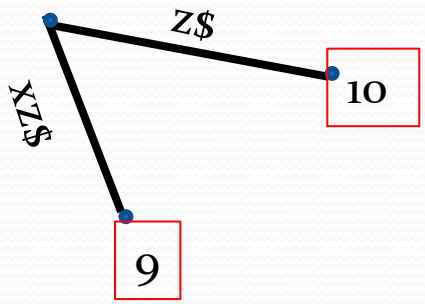
Index = i

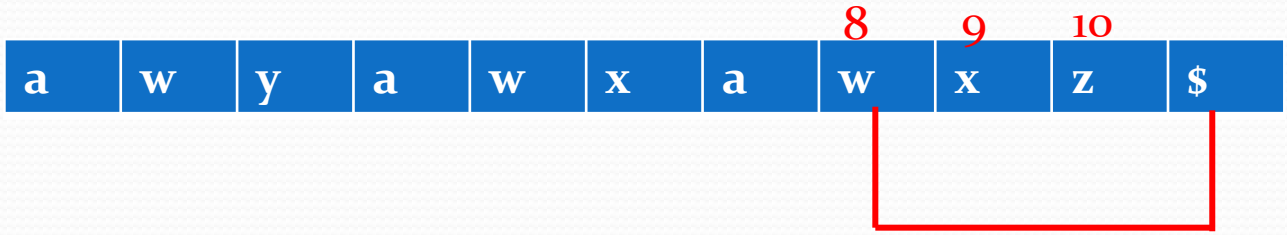
1	2	3	4	5	6	7	8	9	10	11

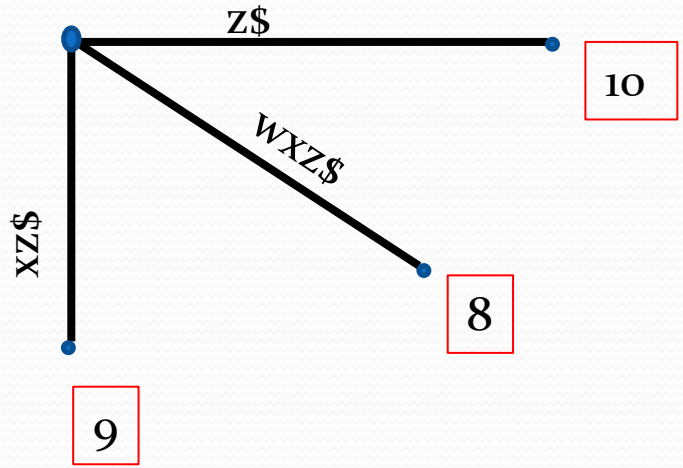
Suffix Tree

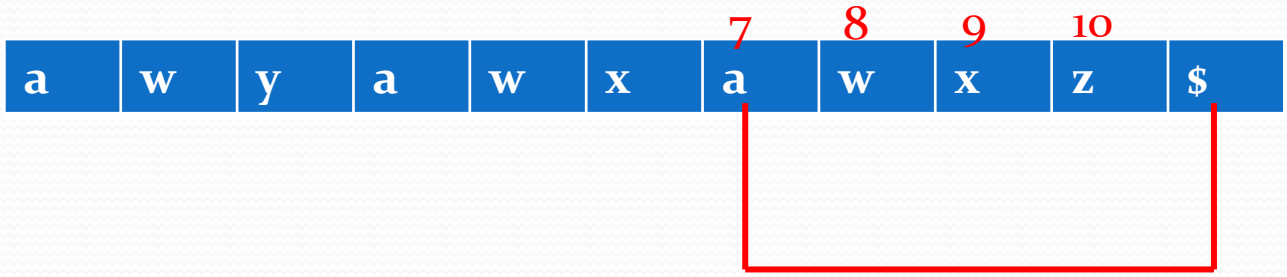


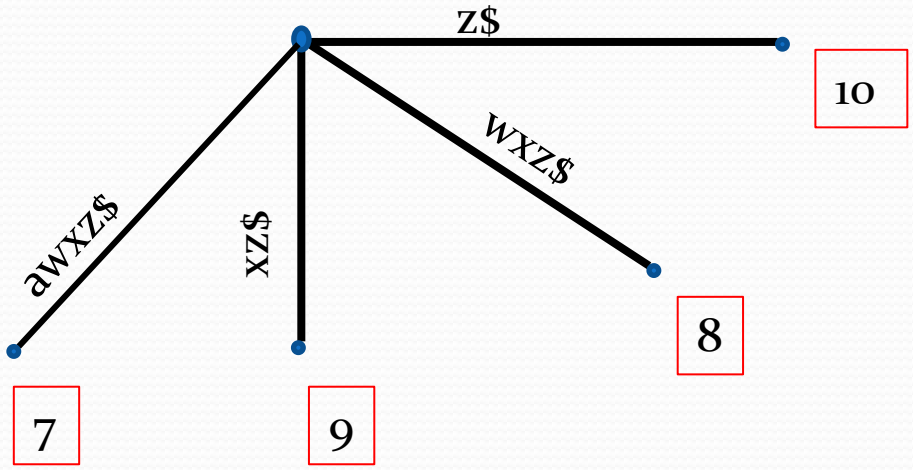

9
10
a
w
y
a
w
x
a
w
x
z
$

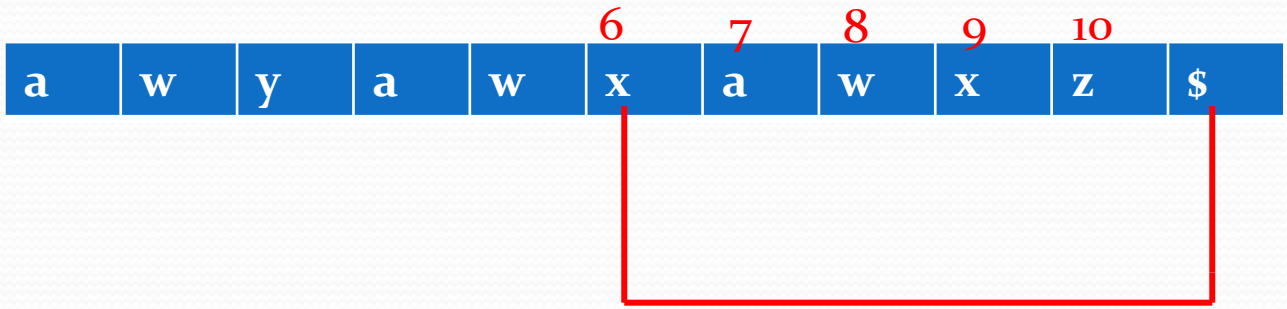

z$
10
xz$
9

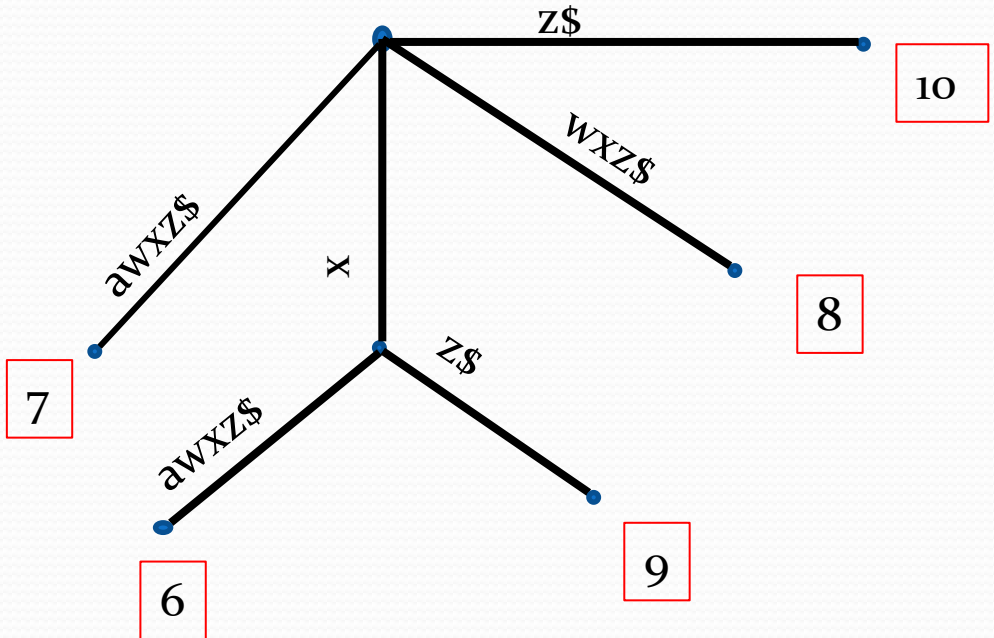

8
9
10
a
w
y
a
w
x
a
w
x
z
$

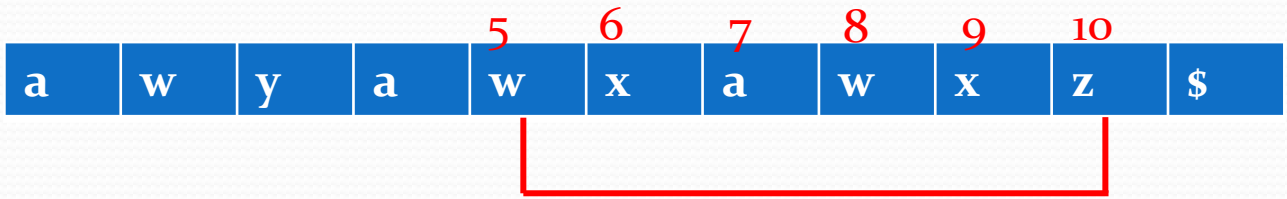

Z$
10
WXZ$
XZ$
8
9

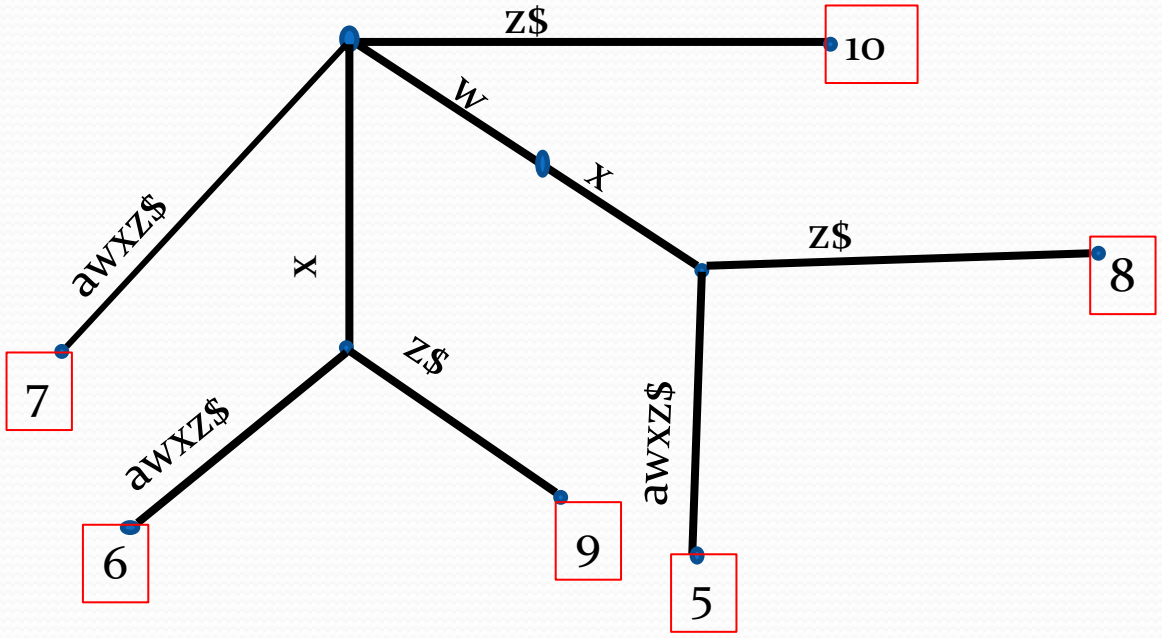

7
8
9
10
a
w
y
a
w
x
a
w
x
z
$

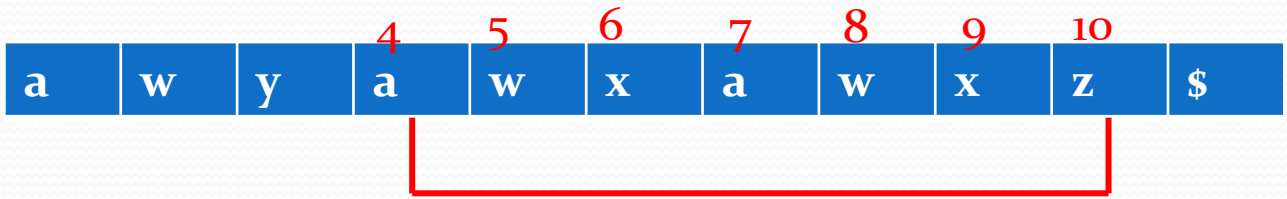

z$
10
wxz$
8
awxz$
xz$
7
9

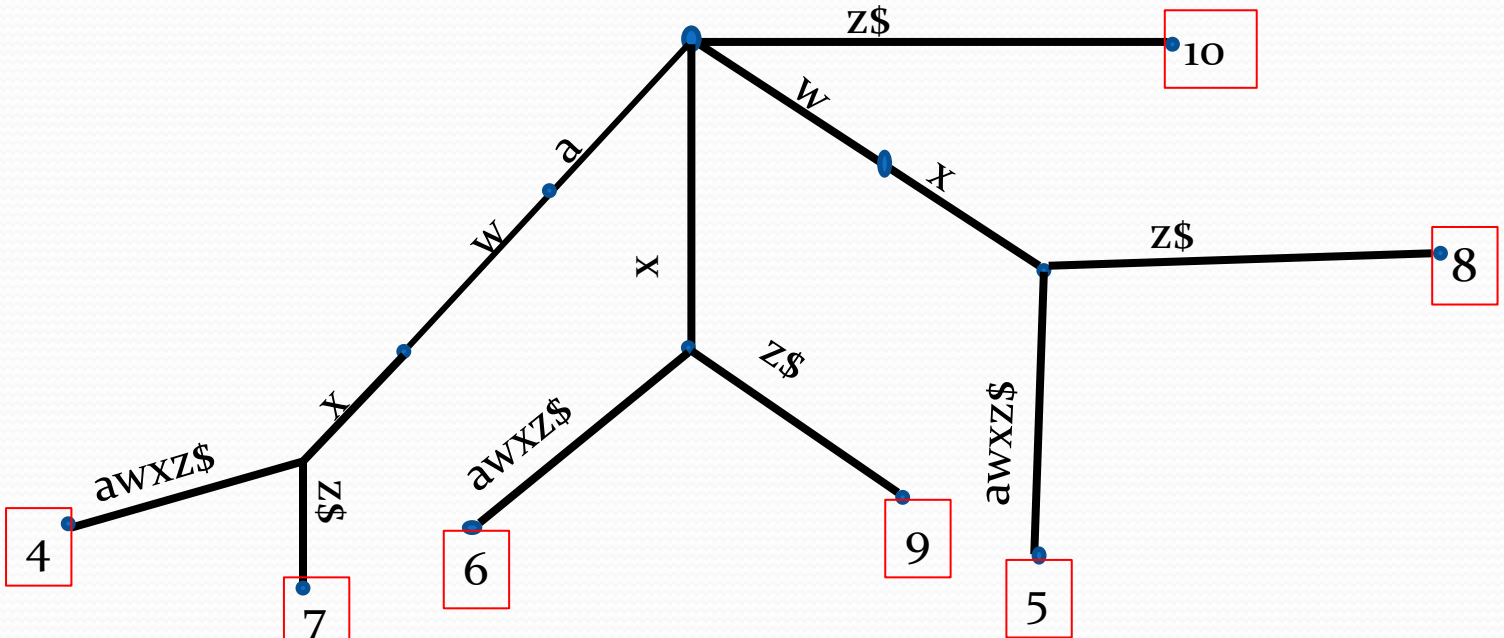

6
7
8
9
10
a
w
y
a
w
x
a
w
x
z
$

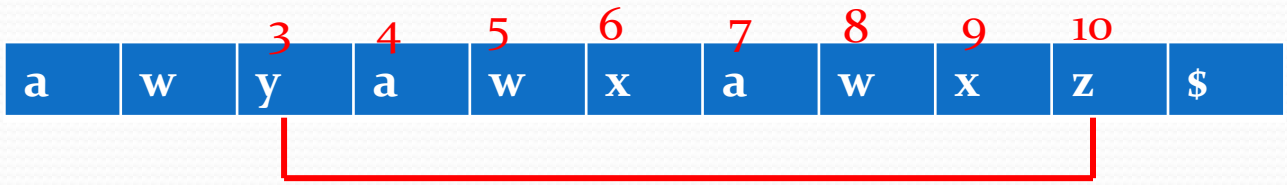

z$
10
wxz$
8
awxz$
x
7
z$
awxz$
9
6

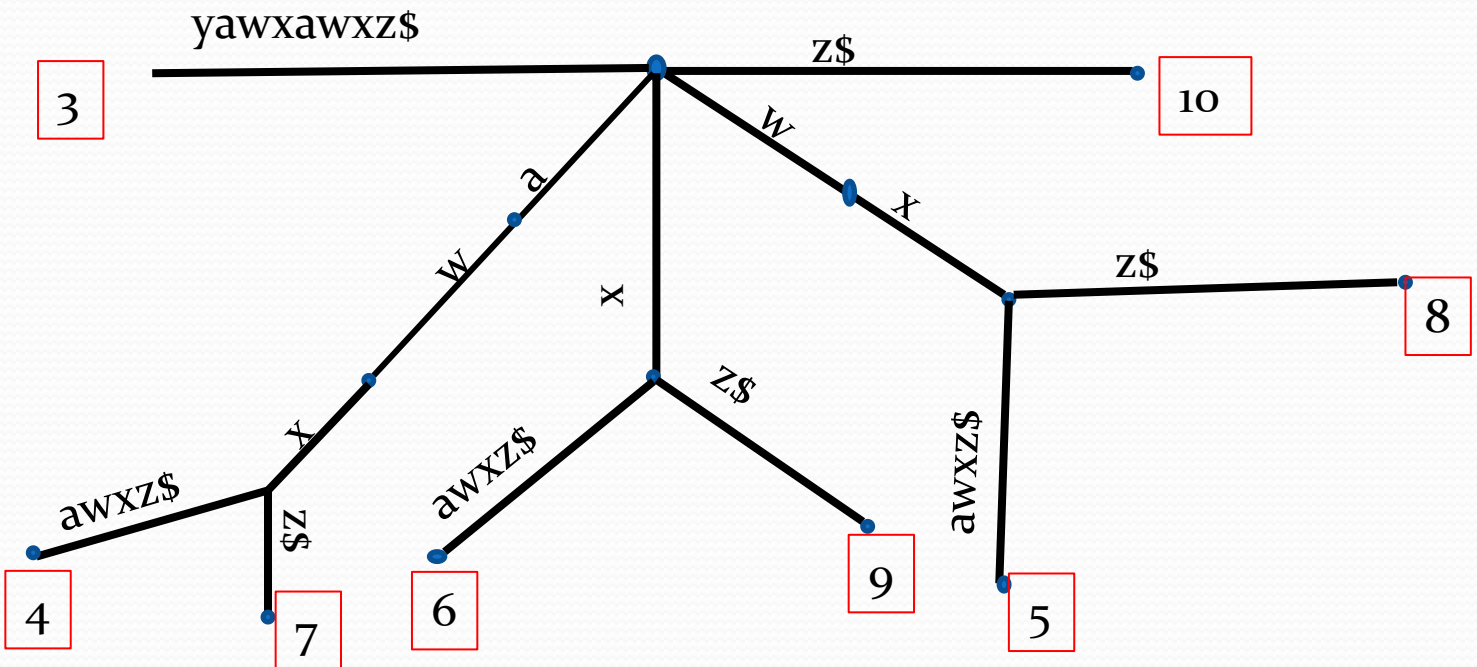

5
6
7
8
9
10
a
w
y
a
w
x
a
w
x
z
$

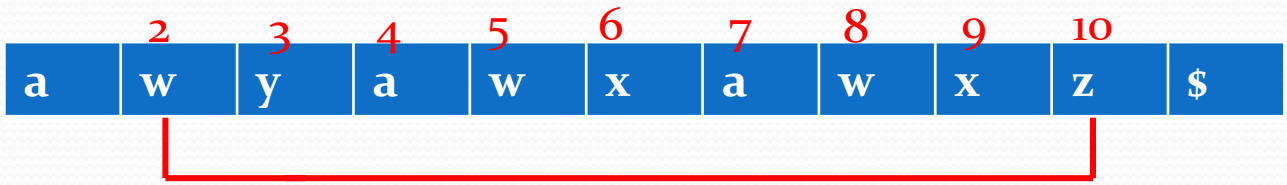

z$
10
w
x
z$
8
awxz$
x
7
z$
awxz$
6
9
awxz$
5

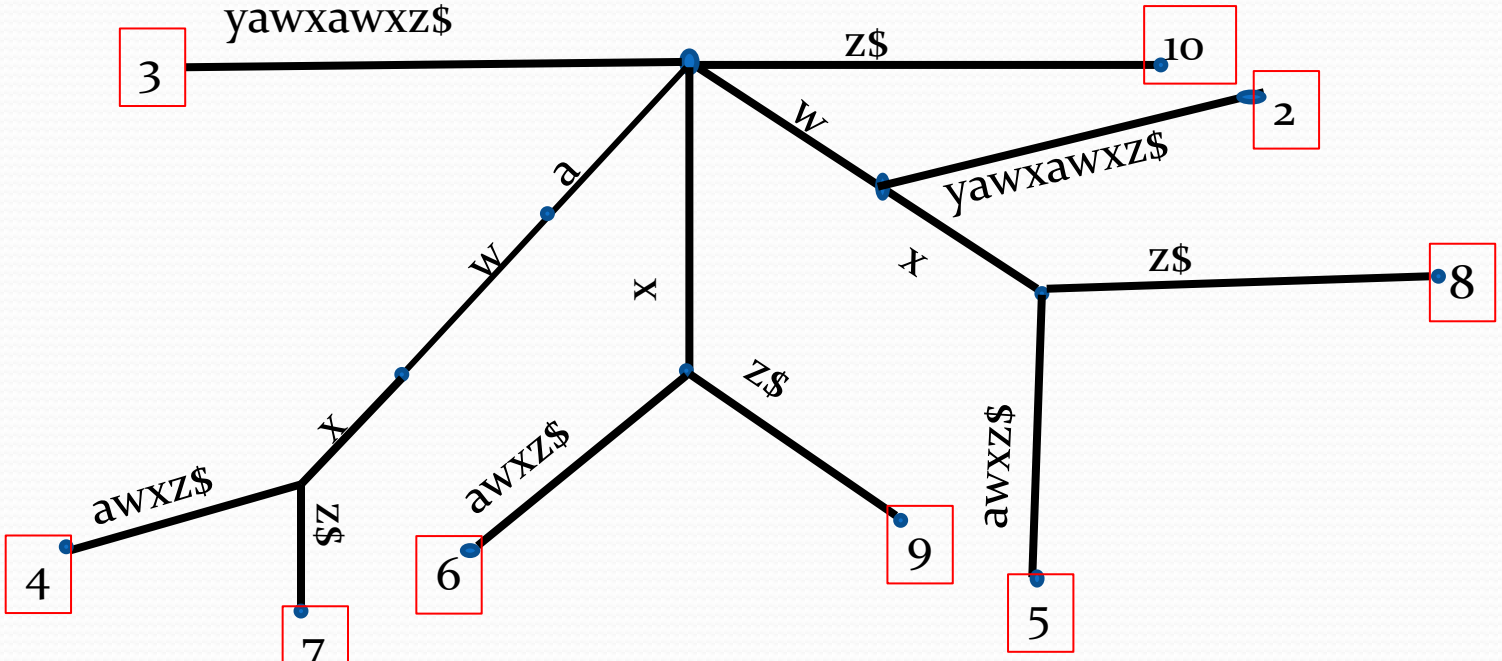

4
5
6
7
8
9
10
a
w
y
a
w
x
a
w
x
z
$


z$
10
w
x
z$
8
a
w
x
awxz$
$z
4
7
x
awxz$
6
z$
9
awxz$
5

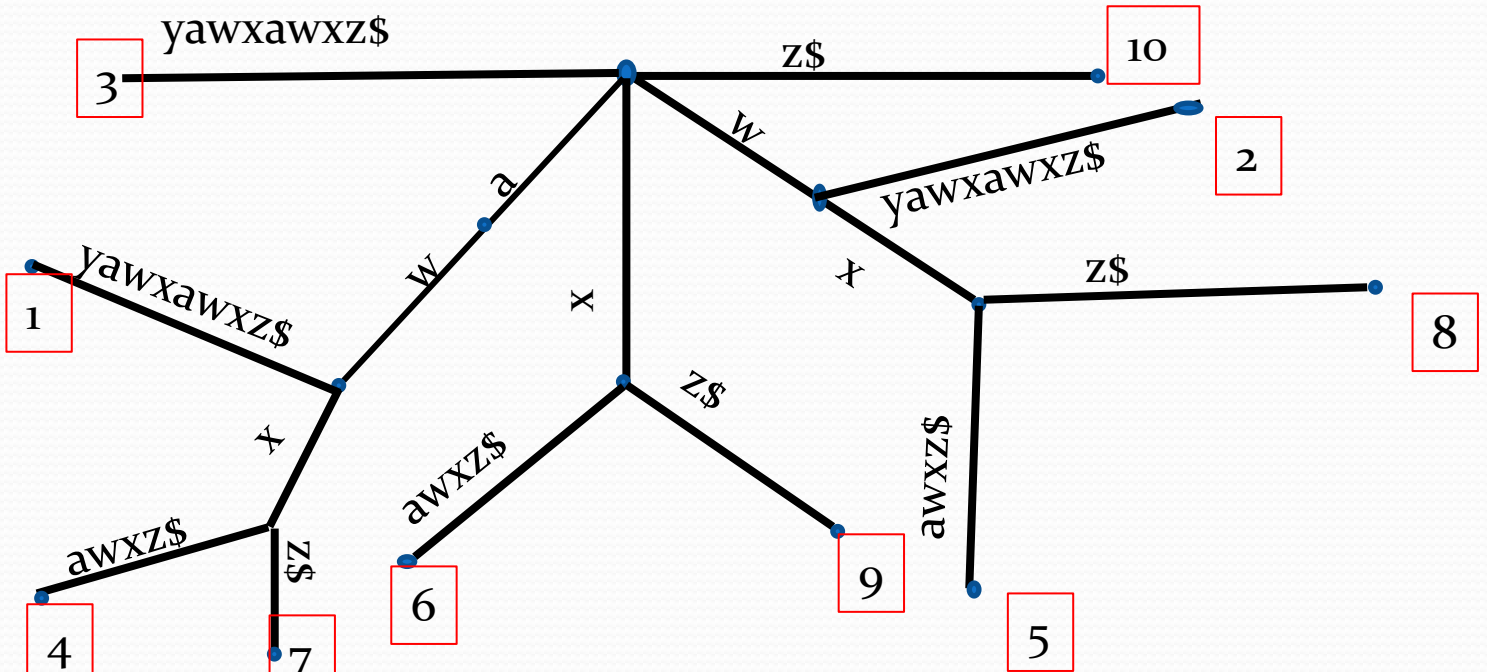

3 4 5 6 7 8 9 10
a w y a w x a w x z $


yawxawxz$
3
z$
10
w
x
z$
8
a
w
x
x
z$
awxz$
awxz$
awxz$
$z
4
7
6
9
5


2 3 4 5 6 7 8 9 10
a w y a w x a w x z $


yawxawxz$
3
z$
10
2
w
yawxawxz$
a
w
x
x
z$
8
z$
awxz$
awxz$
awxz$
x
$z
4
6
9
5
7

1	2	3	4	5	6	7	8	9	10	
a	w	y	a	w	x	a	w	x	z	$



Time Complexity

Major difference between other algorithms and suffix tree is the time complexity , the time complexity in suffix tree is linear and hence suffix tree construction is more fast. In order of computing time complexity of suffix tree ,we have to consider , (m) which is text length and P which is called Pattern