Ch 2. Combinatorial Pattern Matching Algorithms

2.1. Finite automata & Regular Expressions

2.1. Knuth-Morris-Pratt Alg
2.2. BW-Transform
2.3. Suffix Tree Algorithms
2.3 Suffix Tree Algorithm

A data structure for storing all the substrings of a string.

An example.

\[ \Sigma = \{a, b\}, \quad \$ \notin \Sigma \]

Input string: \( W = abaabbaa\$ \)

Set of suffixes of \( W \):

1. \( abaabbaa\$ \)
2. \( abbaa\$ \)
3. \( aabb\$ \)
4. \( abbaa\$ \)
5. \( bbaa\$ \)
6. \( baaa\$ \)
7. \( aaa\$ \)
8. \( aa\$ \)
9. \( a\$ \)
If \( |w|=n \) there are \( n \) \( b \)'s.

and \( 1+2+\ldots+n = \frac{n(n+1)}{2} \geq O(m^2) \) storage

The most practically used data structure in computer biology/bioinformatics

\[ w = ababaababacaq $ \]

\[ 1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7 \ 8 \ 9 \]

Path length of suffix 5 is \( 264aaq $ \)

\( O(m^2) \) time

\( O(m^2) \) space

is \( abb \) substring of \( w \)?

YES

is \( aabb \) substring of \( w \)?

NO
A suffix tree for string \( W \) is a data structure satisfying the following properties:

1. It stores the starting position of each suffix (marked as the leaves).

2. It stores every substring of \( W \).

3. Each suffix of \( W \) can be identified as a path label from the root vertex to some leaf, AND vice-versa: every path from a leaf to the root is a path label for a suffix of \( W \).

4. Every internal node has at least two successor vertices/links.
(5) edges leaving some node are labeled with substrings of $w$ with different first characters

(6) every leaf is labeled with the starting position of the suffix of $w$ that looks suffix as a path label

$w = a b a a b b a a a q$ $\$ 
1 2 3 4 5 6 7 8 9

Diagram of a suffix tree with nodes labeled with suffixes of $w$. The root is labeled with the entire string, and each node is labeled with a substring of $w$. The leaves are numbered from 1 to 9, corresponding to the starting positions of the suffixes.
Construct the suffix tree in linear space: $O(m)$

It's a very simple trick.

Represent a string by a pair of positions: begin substring and end substring.

$w = abaabbaaaq$

1 2 3 4 5 6 7 8 9

Substring $aaba$ $[3, 6]$,

Substring $abaq$ $[1, 4, 7]$.

Representing each substring that way gives $O(m)$ space for $ST$. 
t = text (long string)
p = pattern (short string)

You can solve the KMP problem with Suffix Trees

**Solution**
- Construct a Suffix Tree for t
- Ask if p is a substring of t

- Start spelling p from a root of the suffix tree
  - If you can finish testing p as a path from the root, then answer YES p occurs exactly in t.
- If along the way you get stuck, cannot continue reading p then
answer no.

Solve more

w = abaabbbabaa

The EX.

Again

123456789
Now in $O(m)$ space