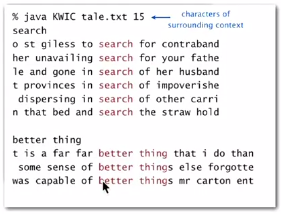
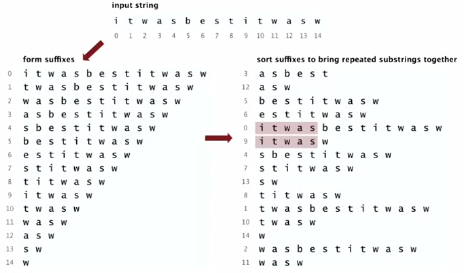
Suffix arrays

Given a text of N characters, preprocess it to enable fast substring search (find all occurrences of query string context)



Suffix sort

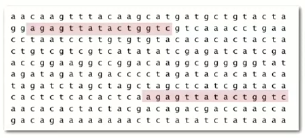


1. Preprocess: suffix sort the text
2. Query: binary search; scan until mismatch

Related problem:

Longest repeated substring

Given a string of N characters, find the longest repeated substring



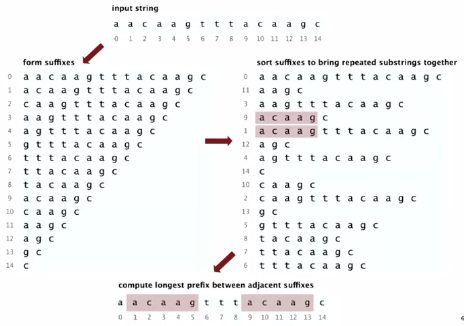
Applications: bioinformatics, cryptanalysis, data compression, etc.

Solution 1: Brute-force algorithm

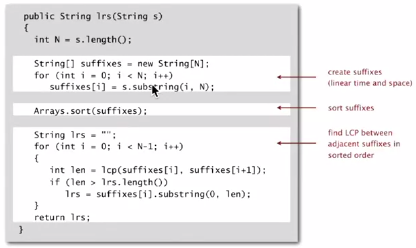
* Try all indices i and j for start of possible match
* Compute longest common prefix (LCP) for each pair

Analysis: Running time <= D N2, where D is length of the longest match

Solution 2: Use suffix sorting

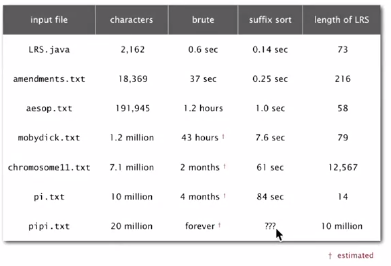


LRS Java implementation



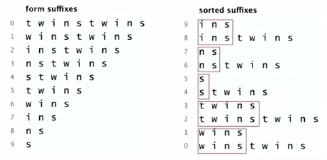
PROBLEM:

If longest repeating substring is too long, this will take far too long, if not virtually forever, to complete



E.g.:

* Same letter repeated N times
* Two copies of the same Java codebase



D = length of longest match

LRS needs at least 1 + 2+ 3 + … + D character compares

Running time: quadratic OR WORSE in D for LRS (also for sort)

Problem: suffix sort an arbitrary string of length N

What is the worst-case running time for this problem?

* Linearithmic (Manber-Myers algorithm)
* Linear (Suffix trees)

Manber-Myers

* Phase 0: sort on first character using key-indexed counting sort
* Phase i: given array of suffixes sorted on 2i-1 characters,   
  create array of suffixes sorted on first 2i characters

Worst-case running time: N ln N

* Finished after lg N phases
* Can perform a phase in linear time.

