6.897
L10.3
Mar. 31, 2003

Document retrieval [Muthukrishnan - SODA2002]

- suffix tree applies to multiple text documents
  - each uses their own $
- search in suffix tree finds all occurrences of P
- What if we just want to find distinct documents containing P?
  - i.e. distinct $'s in subtree of suffix tree
  - i.e. distinct $'s in interval of a traversal of suffix tree
    - suppose interval is [i:j]
    - store with each $ the index of previous $ of same type
    - want $'s in [i:j] with stored index < i (⇒ NEW)
      - compute min index in range [i:j] → k
      - if index at k is < i:
        - add it to results
        - recurse in [i:k-1]
        - recurse in [k+1:j]

Time: O(#answers × cost of range min query)

Range min. queries [Gabow, Bentley, Tarjan - STOC 1984]

- construct Cartesian tree of an array of numbers:
  - root = min. in array
  - left & right subtrees = recursive Cartesian trees of left & right pieces partitioned by root (by index)
- min in range [i:j] = LCA of i & j (corresp. nodes)

↓

Least/Lowest Common Ancestor

LCA in suffix tree

- in trie, LCA gives longest common prefix match of 2 strings
- in suffix tree, finds longest common prefix of 2 substrings
- in suffix tree of multiple strings, can compare substrings of different strings
- e.g.: find longest palindrome centered at a character i by finding longest common prefix of x[i] & reverse(x)[|P|-i+1:]
  - ↳ computable in O(1) time with suffix tree (x, reverse(x))
- e.g.: find all positions of P in T with ≤ k mismatches in [redacted] time
  - find lcp(T[i:], P) → first mismatch; etc. O(k(|P|+|T|))