Implementation of Suffix Array.

R Siva Girish
Computer Science
PES University
PES1201700159
sivagirish81@gmail.com

Abstract

A Suffix Array is an array consisting of all possible suffixes of a string sorted according to the suffixes.

Keywords

Suffix Array; Partial Match; Full match; documents; Queries

I. Problem Statement

The Problem at hand is to construct a suffix array and use the suffix array determine all the matches of a particular pattern across multiple documents.If the pattern matches completely then output the document number as well as the index otherwise find the longest partial match of the pattern across all documents.In case there is no partial matches as well then output -1.

II. Implementation

To implement a suffix array we use a SuffixArray object which consists of the position of the suffix in the document and the suffix itself.We don't necessarily construct the suffix array for all the documents but we do it only in case there exists no full match.Construction of suffix Array takes O(n^2) to implement which is a costly operation and there is no need for a suffix array to match entire strings.Therefore only when there is a partial match needed to be found do we actually construct a suffix array.

- ❖ Matcher(documents,queries) : Matches each query in all documents and checks for the first full match.If a full match occurs the prints the document no. as well as position in document otherwise searches for partial match and return the results or -1 depending on whether pattern is found or not.
- ❖ LM(documents,query) : Takes a single query and tries to match the longest substring and returns the results.
- ❖ Longest_Common_Substring(document,query) : Takes the particular document and the query combines both of them.Construct a suffix array for the combined text and finds the length of the longest match between the document and pattern making sure to avoid matches within the document and pattern itself in the suffix array.
- ❖ Locate(a , b) : Returns the first occurance of b in a.
- ❖ match_count(a,b):Returns the number of characters of b matched in a or vice versa.
- ❖ sorter(A,B):Used as a helper function to determine the comparison factor to be used in sorting.

III. Analysis

The overall complexity of the algorithm is O(k n^2) where k is the number of documents and n is the max size of the documents.Construction of suffix array takes O(n^2) as well as to match full strings.Therefore overall complexity is O(Kn^2) ~ O(n^3).
A Suffix tree for this problem is not advisable as it would increase the overall time complexity by a large magnitude precisely because we would end up constructing k suffix trees for all the elements and have to traverse each one of them individually to locate all partial matches.

Therefore a Suffix Array is ideal for this problem and hence the reason for choosing a suffix array over a suffix tree for implementing the algorithm.