CSE 5311 Lab Assignment 3

Due August 12, 2011

Goals:

1. Understanding of suffix arrays, along with the related notions of suffix rank and longest common prefix (lcp).

2. Review of elementary state machine concepts.

3. Application of the above concepts to the ***longest common substring*** problem.

Requirements:

1. Write (and test) a C/C++ program to determine a longest common substring of three input strings with no more than 500 symbols each. Each string will be on a line by itself. These strings will be readable using `scanf` or streams, since they will only include upper/lowercase alphabetic symbols or digits. Your code should run in time $O(n)$, not including the time to compute the suffix array, where n is the total number of input symbols.

   Your output is any maximum length string common to all three input strings. It is important to find a ***substring*** (i.e. contiguous symbols) rather than a ***subsequence***.

   Your program must compile and execute on at least one of omega.uta.edu or Visual Studio.

2. Email your code (as attachments) to `miao.zhang@mavs.uta.edu` before 3:15 pm on August 12. The subject should include your name as recorded by the University.

Getting Started:

1. You may use suffix array code from the course webpage or elsewhere. Code to find a longest common substring for two input strings is available on the course webpage.

2. Symbol-to-symbol comparisons should occur only during the preprocessing before finding the longest common substring.

3. When scanning the tables for the final result, you will need to examine various intervals that include a suffix for each of the three input strings. Since the longest common substring for the interval will have the minimum lcp value in the interval as its length, you only need to process minimal intervals such that leaving out the first or last suffix will leave an interval with only two of the three input strings represented. This notion may be summarized using regular expressions: if x y z is a permutation of {0, 1, 2}, where these correspond to the three input sequences, then a minimal interval will include suffixes according to the regular expression $x\ y^{+}\ z$. Thus, the scanning can be done by a simple automaton that tracks the current values (if any) for x, y, and z, along with maintaining the indices and updating the longest known common substring.