COMS 4995 Parallel Functional Programming
Project Proposal - Burrows-Wheeler transform

David Winograd (dmw2181), Zulal Ozyer (zo2146)

November 22, 2021

In this project, we will implement the Burrows-Wheeler transform. Burrows-Wheeler transform takes a character string (either a word or sentence) and alters it to make it easier for compression. There are many applications of the BTW algorithm, such as in the field of biology, when dealing with lettered genome sequences (A, C, T, G). The general idea consists of constructing a list in which the rows are all rearranged systematically, typically in “dictionary order”. Once elements in the list are clustered together, they are ordered in such a way that the string itself becomes more compressible due to similar letters being close to one another.

There are 3 steps to perform this algorithm.

1. In the first step, we form all the cyclic rotations of a given string. If we use a sample word such as “tree”, the rotations would consist of “tree$”, “$tree”, “e$tre”, “ee$tr”, and “ree$t”.
2. In the second step, we order the formed rotations lexicographically. In our example, this would be sorted as “$tree”, “e$tre”, “ee$tr”, “ree$t”, “tree$”, so our last column would be “eert$”.
3. The last step is outputting the last column of the strings. We do this because the last column will have the best clustering of symbols when compared to all of the other columns. Also, with this BWT output, the remaining rotations of the original word can be recovered. None of the other columns have this unique characteristic.