**How pstringDist works:**

The method is called by passing in x (a character vector or an XStringSet object) and optionally setting the method to be hamming, and the filename to be the name of the file to save the results to. The hamming method is always used for the calculation and the option to set it is just to match the stringDist method.

pstringDist(x, method="hamming", filename=NULL). A temporary ff file will be created if no filename is entered.

The function compares each string with every other string and counts the number of positions at which the strings differ. Identical strings have a hamming distance of zero. The strings must all be of the same length. The result is an n by n matrix (where n is the number of strings). The size of the result is the limiting factor on the size of the data that can be handled. When the number of data strings to be compared is greater than 46 340 then the number of result values (input squared) is greater than .Machine$integer.max (2^31-1) and the method fails.

The algorithm:

• Step 1: The master process broadcasts the input dataset to all worker processes.

• Step 2: The worker processes are assigned a section of the data to process (length = dataset/number of processes).

.

• Step 3: Then the processes try to determine how much memory they can assign for the result, but due to virtual memory usage it is unlikely that malloc ever fails and invokes this step.

• Step 4: The worker processes calculate the hamming distance between each string in their section of data and all of the other strings. The worker processes then write the output to the appropriate section of the output file.

• Step 5: The data in the output file is returned as an ff object in R.

n = number of strings to be compared

L= string length

p = processes/node

m = memory/node

n\*n\*8

nodes needed = m-((n\*L\*8)\*p)

The number of nodes needed to process a given amount of data will be equal to the size of the output data (n\*n\*8) divided by the memory remaining on each node after the input data is considered (m-((n\*L\*8)\*p)). As the input data is copied across to each worker, the maximum memory will be available when only one process runs on each node.