Prediction with Partial Match(ppm)

The best-known context-based algorithm is the ppm algorithm, first proposed by Cleary and  
Witten in 1984. It has not been as popular as the various Ziv-Lempel-based algorithms  
mainly because of the faster execution speeds of the latter algorithms. Lately, with the  
development of more efficient variants, ppm-based algorithms are becoming increasingly  
more popular. The idea of the ppm algorithm is elegantly simple. We would like to use large contexts to determine the probability of the symbol being encoded. However, the use of large contexts would require us to estimate and store an extremely large number of conditional probabilities, which might not be feasible. Instead of estimating these probabilities ahead of time, we can reduce the burden by estimating the probabilities as the coding proceeds. This way we only  
need to store those contexts that have occurred in the sequence being encoded. This is a  
much smaller number than the number of all possible contexts. While this mitigates the  
problem of storage, it also means that, especially at the beginning of an encoding, we will  
need to code letters that have not occurred previously in this context. In order to handle  
this situation, the source coder alphabet always contains an escape symbol, which is used to  
signal that the letter to be encoded has not been seen in this context.

The basic algorithm initially attempts to use the largest context. The size of the largest  
context is predetermined. If the symbol to be encoded has not previously been encountered in  
this context, an escape symbol is encoded and the algorithm attempts to use the next smaller  
context. If the symbol has not occurred in this context either, the size of the context is further  
reduced. This process continues until either we obtain a context that has previously been  
encountered with this symbol, or we arrive at the conclusion that the symbol has not been  
encountered previously in any context. In this case, we use a probability of 1/M to encode  
the symbol, where M is the size of the source alphabet. For example, when coding the a  
of probability, we would first attempt to see if the string proba has previously occurred—  
that is, if a had previously occurred in the context of prob. If not, we would encode an escape and see if a had occurred in the context of rob. If the string roba had not occurred  
previously, we would again send an escape symbol and try the context ob. Continuing in  
this manner, we would try the context b, and failing that, we would see if the letter a (with a  
zero-order context) had occurred previously. If a was being encountered for the first time,  
we would use a model in which all letters occur with equal probability to encode a. This  
equiprobable model is sometimes referred to as the context of order −1

≈16000 symbols