

Prediction by Partial Match

By Mrs. Shubhangi Bhosale

The "Prediction by Partial Match" method, originally developed by Cleary and Witten, with extension and an implementation by A. Moffat, is capable of very good compression on a wide variety of source data. The adaptive nature of the scheme and the flexibility afforded by arithmetic coding mean that an effective compression model will be built for any input file that is reasonably homogeneous. Cleary and Witten reported that their scheme was capable of representing English text in as little as 2.2 bits/ character.

The method is based on an encoder that maintains a statistical model of the text. The encoder inputs the next symbol S , assigns it a probability P , and sends S to an arithmetic encoder, to be encoded with probability P . The statistical model counts the number of times each symbol has occurred in the past and assigns the symbol a probability based on that. In a context based statistical model, the idea is to assign a probability to symbol S depending not just on the frequency of the symbol but on the contexts in which it has occurred so far. A static context based modeler always uses the same probabilities and offers the advantage of being simple and producing good result on average. However it may lead to considerable expansion in the case when certain input stream is statistically very different from the data originally used to prepare the table and when it encounters zero probabilities. The arithmetic encoder requires all symbols to have non-zero probabilities. Another reason why a symbol must have non-zero probability is that its entropy depends on $\log_2 P$, which is undefined for $P=0$.

An adaptive context-based modeler updates its probability table all the time as more data is being input, which adapts the probabilities to the particular data being compressed. Such a model is slower and more complex but produces better compression. An order N adaptive context based modeler reads the next symbol S

from the input stream and considers the N symbols preceding S the current order N context C of S . The model then estimates the probability P that S appears in the input data following the particular context C . Theoretically, the larger the N , the better the probability estimate. However, a larger context is difficult to manage.

A very long context retains information about the nature of old data. Experience shows that large data files contain different distributions of symbols in different parts. Better compression can therefore be achieved if the model assigns less importance to information collected from old data and more weight to fresh, recent data. Such an effect is achieved by a short context.

The central idea of PPM is to use this knowledge. It uses an adaptive model based on a variable length context. At each coding step the longest previously encountered context is used to predict the next character. If the symbol is novel to the context, an escape code is transmitted and the context shortened by dropping one symbol. The Process continues until the symbol is successfully transmitted. If the current symbol is novel even to the zero order context then a final escape is transmitted, and the symbol will be transmitted as an 8 bit code. The adaptive model then adds the current symbol to all applicable contexts.

For Example, Consider an 11 symbol string

xyzzxyxyzzx

has been input and encoded.

Let us assume the 12th symbol is z . various order statistics are:

Order 4	Order 3	Order 2	Order 1	Order 0
xyzz-x 2	xyz-z 2	xy - z 2	x- y 3	x 4
yzzx-y 1	yzx - x 2	xy - x 1	y- z 2	y 3
zzxy-x 1	zxx - y 1	yz - z 2	y - x 1	z 4

zxyx-y 1	zxy - x 1	zz - x 2	z- z 2	
xyxy-z 1	xyx - y 1	zx - y 1	z- x 2	
yxyz-z 1	yxy - z 1	yx - y 1		

The order 4 context is now yzxx, which earlier was seen followed by y but never by z. The encoder therefore switches to the order 3 context, which is zxx. The next lower order is zx and it also fail. Similarly order 1 also fails. Order 0 is checked next, where z has a frequency count of 4. Symbol z is thus sent to the adaptive arithmetic encoder, to be encoded with probability 4/11. The table is then updated.

Order 4	Order 3	Order 2	Order 1	Order 0
xyzz-x 2	xyz-z 2	xy - z 2	x- y 3	x 4
yzzx-y 1	yzz - x 2	xy - x 1	y- z 2	y 3
zzxy-x 1	zzx - y 1	yz - z 2	y - x 1	z 4
zxyx-y 1	zxy - x 1	zz - x 2	z- z 2	
xyxy-z 1	xyx - y 1	zx - y 1	z- x 2	
yxyz-z 1	yxy - z 1	yx - y 1		

Consider the PPM decoder. There is a fundamental difference between the way the PPM encoder and decoder work. The encoder decides to switch to a shorter context based on what the next symbol is. The decoder can not mirror this, since it does not know what the next symbol is. The PPM algorithm reserves one symbol of the alphabet as an escape symbol which makes it possible for the decoder to stay in lockstep with the encoder. When encoder switches to a shorter context, it first writes the escape symbol on the output stream.

The decoder decodes the escape sequence since it is encoded in the present context.

After decoding an escape, the decoder also switches to a shorter context. Initially escape probability should be high. It drops as more symbols are input and decoded,

and more information is collected by the modeler about contexts in the particular data being compressed.

Based on the different methods of assigning probabilities to escape symbol following variants of PPM are presented.

- PPMA
- PPMB
- PPMC

In PPMA, a group of symbols has total frequencies n (excluding escape symbol).

The escape symbol is assigned a probability $= 1/(n+1)$. This is equivalent to always assigning it a count of 1. Other members are still assigned their original probabilities (x/n) .

In PPMB, a symbol S following context C is assigned a probability only after S has been seen twice in context C . This is done by subtracting 1 from the frequency counts.

The subtracted 1's are added to the count of the Escape Symbol.

The way Escape probabilities are assigned in the three methods is based on intuition and experience, not on any underlying theory. Experience with the three variants of PPM shows that none is preferable. They produce compression ratios that normally differ by just a few percent. This shows that the basic PPM algorithm is robust, and does not depend on the precise way of assigning escape probabilities.

REFERENCES:

1. Moffat, A. (1990) "Implementing the PPM data Compression Scheme", IEEE Transactions on Communications Vol. 38 No. 11
2. Data Compression by Salomon.