PPM*-Style Context Sorting Compression Method Using a Prefix List
Suguru Itagaki and Hidetoshi Yokoo
Department of Computer Science, Gunma University
Kiryu 376-8515, JAPAN
Email: suguru@isl.cs.gunma-u.ac.jp
yokoo@cs.gunma-u.ac.jp

We present a new implementation of the context sorting data compression method [2],
which is an on-line adaptive algorithm for text compression. Our key idea is to utilize
a new data structure called a prefix list [3]. The original context sorting compression
method, which uses neither explicit modeling nor arithmetic coding, can be viewed
as a symbol ranking text compressor. In the method presented here, in contrast, we
form a context model with the frequency distribution to predict the current symbol.
Our context model can exploit contexts of unlimited length, and it is combined with
arithmetic coding. In these respects, the proposed method can also be viewed as
giving an implementation of PPM*[1]. Our space requirement is linear in the string
length without depending on the context order.

The prefix list [3] is a dynamic data structure, which was proposed primarily to
maintain a set of contexts in reverse lexicographic order. We can easily gather pre-
vious contexts according to their similarity to the current context. Predicted symbols
can also be enumerated as the following symbols in those contexts. While enumerat-
ing those predicted symbols, we can completely simulate PPM*. If a set of contexts
whose similarities to the current context are $d$ or larger gives only one prediction, then
their common $d$-symbol suffix is said to be a deterministic context. In our method,
we begin the PPM mechanism with the shortest deterministic context if any, or with
the contexts most similar to the current one otherwise. An escape symbol is emitted
each time the similarity between the current context and the existing context pointed
to by a pointer in the prefix list decreases. This process is repeated with progressively
shorter contexts until the same predicted symbol as the current symbol is reached.

Most existing context-based models use tree structures to store data statistics
because the tree structure is the most natural representation of a context set. A
typical tree holds statistic information, e.g., symbol frequencies, explicitly in each
node. On the other hand, the prefix list contains no such information in itself; it
should be traversed whenever it is required to produce such statistics. However,
this is not necessarily a drawback. We can show that our method encodes a string
generated from a stationary ergodic source in expected linear time.

References
[3] H. Yokoo, A dynamic data structure for reverse lexicographically sorted prefixes, In M.