SOFTWARE—PRACTICE AND EXPERIENCE, VOL. 24(3), 327–336 (MARCH 1994)

A New Data Structure for Cumulative Frequency Tables

peter m. fenwick
Department of Computer Science, University of Auckland, Private Bag 92019, Auckland, New Zealand (email: p_fenwick@cs.auckland.ac.nz)

SUMMARY

A new method (the 'binary indexed tree') is presented for maintaining the cumulative frequencies which are needed to support dynamic arithmetic data compression. It is based on a decomposition of the cumulative frequencies into portions which parallel the binary representation of the index of the table element (or symbol). The operations to traverse the data structure are based on the binary coding of the index. In comparison with previous methods, the binary indexed tree is faster, using more compact data and simpler code. The access time for all operations is either constant or proportional to the logarithm of the table size. In conjunction with the compact data structure, this makes the new method particularly suitable for large symbol alphabets.

key words: Binary indexed tree Arithmetic coding Cumulative frequencies

INTRODUCTION

A major cost in adaptive arithmetic data compression is the maintenance of the table of cumulative frequencies which is needed in reducing the range for successive symbols. Witten, Neal and Cleary[1] ease the problem by providing a move-to-front mapping of the symbols which ensures that the most frequent symbols are kept near the front of the search space. It works well for highly skewed alphabets (which may be expected to compress well) but is much less efficient for more uniform distributions of symbol frequency. Moffat[2] describes a tree structure (actually a heap) which provides a linear-time access to all symbols. Jones[3] uses splay trees to provide an optimized data structure for handling the frequency tables. The three techniques will be referred to in this paper as MTF, HEAP and SPLAY, respectively. In all cases they attempt to keep frequently used symbols in quickly-referenced positions within the data structure, but at the cost of sometimes extensive data reorganization.

This current paper describes a new method which uses only a single array to store the frequencies, but stores them in a carefully chosen pattern to suit a novel search technique whose cost is proportional to the number of 1 bits in the element index. This cost applies to both updating and interrogating the table. In comparison with the other methods it is simple, compact and fast and involves no reorganization or movement of the data.

CCC 0038–0644/94/030327–10

Received 7 June 1993
Revised 6 October 1993