HASHING THE SUBSCRIPTS OF A SPARSE MATRIX

E. V. D. S. DE VILLIERS and L. B. WILSON*

Abstract.

It has been suggested that key transformation techniques might be a very effective way of manipulating sparse matrices particularly if the operations on the matrix access the elements in an unsystematic way. The purpose of the present paper is to investigate methods of hashing the subscripts of a matrix to give a suitable address in the scatter storage table. Various different types of sparse matrices are considered.

Introduction.

Key transformation techniques are often a very efficient way of storing and searching for records. In these methods the key is used to calculate an address in the scatter storage table (SST) and there are basically two aspects.

(a) The key to address transformation—usually called the hashing method.
(b) The resolution of collisions i.e. the technique used when two or more keys give the same hash address.

Key transformation techniques have been mainly used in symbol tables for compilers or assemblers. However in his general review of the subject Morris [1] remarked that they represent a powerful but little known method for handling sparse arrays and are likely to be effective when the array elements are accessed in an unpredictable order.

In this paper we intend to examine hashing methods suitable for sparse matrices. The problem of collision handling is not investigated here and in fact the chaining method using an overflow table is used throughout. It is thought that this method has some advantages for sparse matrices. Several methods of hashing the keys, which are the row and column subscripts of the non-zero elements of the matrix, are examined and compared both for different kinds of sparse matrix and for different loading factors in the scatter storage and overflow tables. A performance study

Received January 9, 1974. Revised May 4, 1974.

* At present visiting University of Waterloo, Ontario, Canada.
of key-to-address transformation techniques has been given by Lum, Yuen and Dodd [2]. This was a useful starting point for our work although their study does not relate to keys which are the subscripts of sparse matrices. It is hoped in a later paper to compare the key transformation method of manipulating a sparse matrix with the more traditional methods.

2. The principal factors

A hashing method used in conjunction with a scatter storage table must satisfy two criteria.

Firstly it must be simple and so take very little computational time and secondly it must spread the addresses evenly over the scatter storage table so that collisions are kept at a minimum. In our particular case, where the keys being hashed are the subscripts of sparse matrices, the principal factors affecting the efficiency of the hashing method are

(a) the type of sparse matrix,
(b) the hashing method,
(c) the loading factor in the scatter storage table,
(d) the collision handling method,
(e) the computer, operating system and compiler.

2.1. The Sparse Matrices.

A matrix is treated as sparse if it has so few non-zero elements that it is worthwhile adopting special methods for its storage and manipulation. In practice sparse matrices usually have more than 90% of their elements zero. There are many types of sparse matrix arising from different practical problems and in this paper the types considered are

(i) randomly sparse matrices,
(ii) randomly sparse symmetric matrices,
(iii) Banded matrices. Two kinds are considered, the first kind have the non-zero elements randomly distributed across the band, and the second kind have the sort of pattern which is obtained from finite difference methods of solving ordinary or partial differential equations,
(iv) transition matrices of the type used in a parser.

2.2. The Hashing Methods.

Many hashing methods for transforming a general key into an address have been proposed (see for example the articles by Lum, Yuen and