How Many Zeros in the Sparse Matrix?

Koba Gelashvili

koba.gelashvili@tsu.ge

Department of Computer Sciences
Iv. Javakhishvili Tbilisi State University
13, University st., 0186, Tbilisi, Georgia

According to some definitions, a rectangular matrix in which the number of zeros exceeds the number of non-zero elements is called sparse. Otherwise, the matrix is called dense. According to some other definitions, square matrix with n rows is called sparse, if the number of non-zero elements in that matrix is $O(n)$. In [1] suggests: „A sparse matrix is defined, somewhat vaguely, as a matrix which has very few nonzero elements. But, in fact, a matrix can be termed sparse whenever special techniques can be utilized to take advantage of the large number of zero elements and their locations“. Presumably, this means that the "absolute" description of the sparse matrix doesn't exist, the sparseness of matrix is a relative category and depends on the algorithm, which uses the matrix's structure.

Report is devoted to determination of the percentage of zeros in matrix, which is used with conjugate gradient (briefly CG) algorithm, which identifies matrix as sparse.

In order to make use the number of zero-entries of original matrix, we use new storage format, so called jagged non-zero submatrix data structure or simply jnz-format. instead of the original matrix, it stores two jagged matrices, The first matrix contains non-zero entries of original matrix. Second matrix consists of integer numbers, representing column indexes of first matrix's elements (in original matrix). Second matrix has additional (first) row, the first element of which represents the number of rows in original matrix. Next elements are numbers of non-zero entries in rows of original matrix. As a result, column's indexes of non-zero entries of the i-th row of the original matrix are placed in (i+1)-th row of the index matrix.

The memory consumption of jnz-format is asymptotically the same as for currently used best formats.

To determine the percentage of nz-elements after which the CG-method is fairly faster using *jnz-format* data structure, in comparison with common rectangular representation of matrix, the following experiment has been conducted (see https://github.com/vakho10/Sparse-Storage-Formats). 15 matrices with different sizes (number of rows n changes from 100 to 1500 inclusively, with step of 100) have been used. For each n, 5 versions of symmetric matrices with randomly generated numbers have been constructed, with the different percentage of zeroes: 15, 20, 25, 30 and 35. In total 75 different matrices. Vectors of corresponding dimensions with random numbers have been generated for each matrix. For each pair (matrix+vector), CG-method was used. In case of 35% of zeros, for the absolute majority of tests, using the jnz-submatrix has sped up the solution process of the system. By increasing the number of zeros this difference becomes more significant.

Taking into account that *jnz-format* saves memory in case of 67% zero-entries, it seems to us that the notion of sparsity in the sense of run-time (unlike memory) is considerably weaker.

References

1. Yousef Saad. Iterative Methods for Sparse Linear Systems, 2nd Edition, 2003, SIAM