CMPE160 PROJECT#4

Deadline: May 21st, 2004, 17:00

1 Project Description

In this project you will implement a sparse matrix abstract data type. When matrices are sparse (there are many zero terms), we can reduce time and space complexity of matrix operations by retaining only the nonzero terms. If the nonzero terms in the matrix form a pattern (e.g. a triangle or a band), a sequential scheme, in which terms are represented by nodes with *row, column* and *value* fields, may be enough. But for unstructured sparse matrices, a different representation should be used.

In this representation, each row and column of the matrix is represented by a circular linked list with head nodes. Each node has a *head* field which is used to discriminate between a head node and a typical node. Each head node has three additional fields: *down, right* and *next*. *down* is a link into a column list and *right* into a row list. The head node for row i is also a head node for column i. *next* links the head nodes together.

Entry nodes have six fields: *head, row, col, down, right* and *value* (Figure 1). *down* is a link to the next nonzero term in the same column and *right* to the next nonzero term in the same row. *row, col* and *value* fields keep the row number, column number and value of the term, respectively. Each entry node is simultaneously in two different lists.

Head nodes are in three lists: a row list, a column list and the list of head nodes. There is a head node for the list of head nodes and it is identical to the entry nodes. *row* and *col* fields of this node stores the matrix dimensions.

The linked structure for sparse matrix A, Figure 2, is shown in Figure 3.

In this project, you will implement the **SparseMatrix** class according to the above representation. It should contain the following methods:

Input: Read in a matrix and set up its linked representation. First, read in the matrix dimensions and the number of nonzero terms. Then, read each term as (row number, column number, value) triples. Overload >> operator.

Output: Display the whole matrix in a nice way. Overload << operator.

Add: Add two matrices, return the result as a new matrix. Overload **+** operator.

Subtract: Subtract two matrices, return the result as a new matrix. Overload **-** operator.

Multiply: Multiply two matrices, return the result as a new matrix. Overload ***** operator.

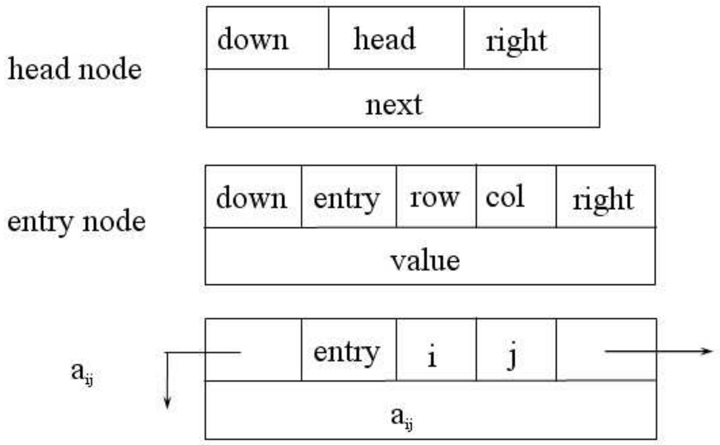

Figure 1: Node structure for sparse matrices

$$\begin{bmatrix} 0 & 0 & 11 & 0 \\ 12 & 0 & 0 & 0 \\ 0 & -4 & 0 & 0 \\ 0 & 0 & 0 & -15 \end{bmatrix}$$

Figure 2: 4x4 sparse matrix A

You will also write a main program that uses the sparse matrix class. It will read two matrices (A and B) from two files and then calculate $C = A + B$, $D = A - B$, $E = C * D$ and display matrix E.

2 Material to Submit

You will prepare the SDD of your project and submit its printout until May 14th, 2004, 17:00. You will submit the printout of the source listing of your program and a diskette containing the source code and the executable (.cpp and .exe). Send an e-mail with your source codes attached to dikmen@cmpe.boun.edu.tr with the name “yournumber.c” (e.g. 97022425.cpp or 01002425.cpp). The subject of the e-mail should be “cmpe160 project4”. Deadline is May 21st, 2004, 17:00.

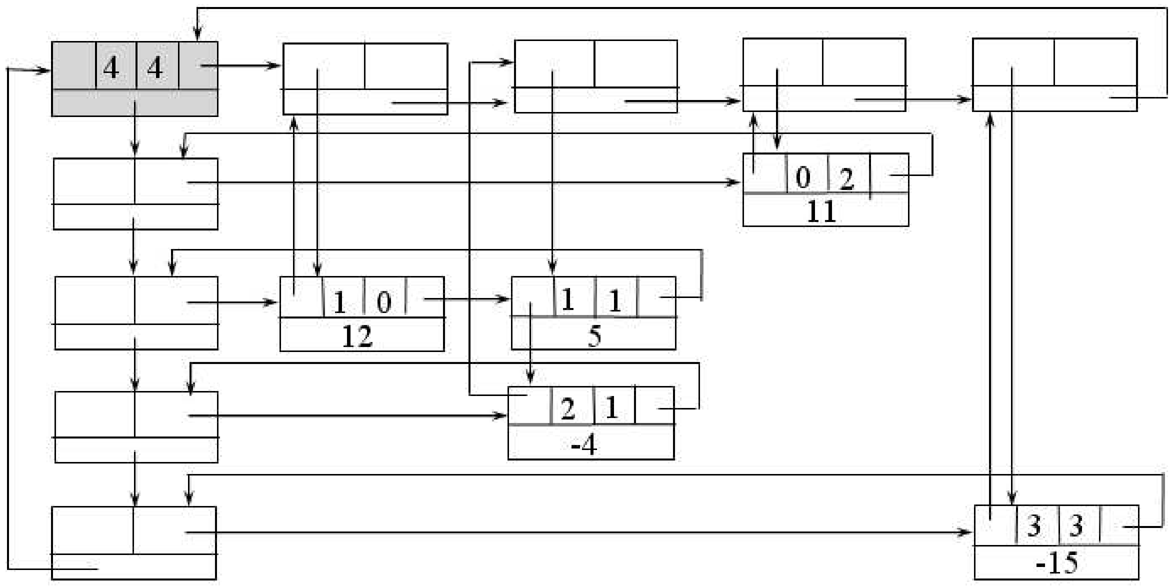

Figure 3: Linked representation of the sparse matrix A