Лабораторная работа №3

1. Задание:

Разреженная (содержащая много нулей) матрица хранится в форме 3-х объектов: - вектор A содержит значения ненулевых элементов; - вектор JA содержит номера строк для элементов вектора A; - связный список IA, в элементе Nk которого находится номер компонент в A и JA, с которых начинается описание строки Nk матрицы A.

1. Смоделировать операцию сложения двух матриц, хранящихся в этой форме, с получением результата в той же форме.
2. Произвести операцию сложения, применяя стандартный алгоритм работы с матрицами.
3. Сравнить время выполнения операций и объем памяти при использовании этих 2-х алгоритмов при различном проценте заполнения матриц.

2. ТЗ:

Цель работы – реализация алгоритмов обработки разреженных матриц, сравнение этих алгоритмов со стандартными алгоритмами обработки матриц.

Входные данные.

Изначально пользователь получает 3 пункта меню: запуск программы со случайно созданными матрицами (пункт 1), запуск программы с веденными вручную матрицами (пункт 2), выход (пункт 0). В случае запуска программы (1 или 2), пользователю предлагают ввести размер матриц: по очереди вводятся размер матриц по строке и по столбику соответственно. Далее, в случае выбора пользователем пункта 1 (случайно генерируемые матрицы), пользователю предлагается ввести процент заполнение каждой из 2 матриц (вводится по очереди, сначала для 1 матрицы, потом для 2). В случае выбора пункта 2 (матрицы вводятся в ручную), пользователю предлагается ввести количество ненулевых элементов (сначала для 1 матрицы, потом для 2), а также сами элементы (ввод осуществляется за 3 этапа: значение элемента, координата по строке, координата по столбику).

Выходные данные.

По ходу выполнения, программа выводит пользователю созданные матрицы (и в случае с рандомной генерацией, и в случае с вводом вручную). В результате работы программа выводит результирующую матрицу - сумму 2 заданных матриц, а также время, затраченное на каждый из 2 алгоритмов суммы: стандартной суммы и суммы разреженных матриц.

Обработка ошибок.

В случае неправильного ввода команды старта, программа говорит пользователю об ошибке и предлагает ввести команду еще раз. В случае некорректно заданного процента заполнения матрицы (<0) программа говорит об ошибке и просит пользователя ввести процент заново).

3. СД:

В программе была использована структура матриц, хранящая в себе массив элементов, массив индексов по I, и также список, в котором хранится индекс первого элемента по строке.

```
struct ElementNode {
    int index;
    struct ElementNode *next;
};

struct SparseMatrix {
    int *A;
    int *IA;
    struct ElementNode *JA;
    int size;
    int rows;
    int columns;
};
```

4. Тесты:

Входные данные:	Время:
5x5, 10%, 10%	Time of standart sum = 257 Time of sparse sum = 78
5x5, 50%, 50%	Time of standart sum = 274 Time of sparse sum = 360
10x10, 10%, 10%	Time of standart sum = 800 Time of sparse sum = 256
10x10, 50%, 50%	Time of standart sum = 469 Time of sparse sum = 693
100x100, 10%, 10%	Time of standart sum = 36200 Time of sparse sum = 12176
100x100, 50%, 50%	Time of standart sum = 32612 Time of sparse sum = 44842 Percent (standart of sparse) = 72
500x500, 10%, 10%	Time of standart sum = 1047706 Time of sparse sum = 697936
500x500, 50%, 50%	Time of standart sum = 726856 Time of sparse sum = 2030761

5. Выводы:

В результате работы были освоены алгоритмы работы с разреженными матрицами. Также был проведен анализ реализованных алгоритмов и сравнение их времени работы с временем работы стандартных функций. При заполнении матрицы более чем 40% не нулевыми элементами , происходит выигрыш стандартного хранения над использованием схемы разреженного построчного хранения матрицы.

6. Контрольные вопросы:

1) Что такое разреженная матрица, какие схемы хранения таких матриц Вы знаете?

- Разреженная матрица - матрица, в которой есть определенный процент не нулевых элементов, а все остальные элементы = 0. Наиболее широко используемая схема хранения разреженных матриц - это схема, предложенная Чангом и Густавсоном, называемая: "разреженный строчный формат". В этом случае значения ненулевых элементов хранятся в массиве AN, соответствующие им столбцовые индексы - в массиве JA. Кроме того, используется массив указателей, например IA, отмечающих позиции AN и JA, с которых начинаются описание очередной строки. Дополнительная компонента в IA содержит указатель первой свободной позиции в JA и AN.

2) Каким образом и сколько памяти выделяется под хранение разреженной и обычной матрицы?

- При хранении обычной матрицы память выделяется под каждый элемент (включая нулевые). В случае с разреженной матрицей, память выделяется только под не нулевые элементы. Значения ненулевых элементов хранятся в массиве AN, соответствующие им столбцовые индексы - в массиве JA. Кроме того, используется массив указателей, например IA, отмечающих позиции AN и JA, с которых начинаются описание очередной строки. Дополнительная компонента в IA содержит указатель первой свободной позиции в JA и A. Поэтому при процентном заполнении матрицы = 100%, хранение матрицы в разреженном виде займет больше места, чем хранение матрицы в обычном виде.

3) Каков принцип обработки разреженной матрицы?

- Нет необходимости пробегать по всей матрице, проверяя каждый ее элемент. Нам заранее известны ненулевые элементы и их позиции, поэтому работая с разреженной матрицей мы работаем только с ненулевыми элементами (это значительно ускоряет работу программы).

4) В каком случае для матриц эффективнее применять стандартные алгоритмы обработки матриц? От чего это зависит?

- Стандартные алгоритмы обработки эффективно применять в случае большого процента заполнения или маленького размера матриц. В случае, если заданы большие матрицы с низким процентом ненулевых элементов, выгодно использовать разреженные матрицы и алгоритмы работы с ними.