Отчет по 3 лабораторной работе

По дисциплине «Типы и структуры данных»

Подготовил Чалый Андрей

Группа ИУ7-32Б

Лабораторная работа № 3 «Обработка разреженных матриц»

**Цель работы** - реализовать алгоритмы обработки разреженных матриц, сравнить эффективность использования этих алгоритмов (по времени выполнения и по требуемой памяти) со стандартными алгоритмами обработки матриц при различном процентном заполнении матриц ненулевыми значениями и при различных размерах матриц.

**Условие задачи:**

Разреженная (содержащая много нулей) матрица хранится в форме 3-х объектов:

- вектор A содержит значения ненулевых элементов;

- вектор IA содержит номера строк для элементов вектора A;

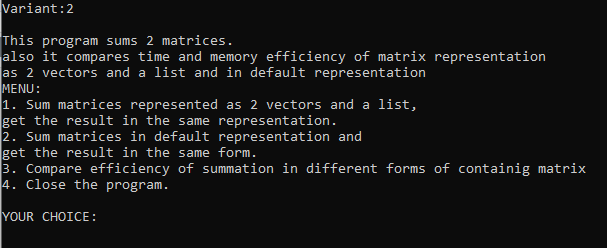
- связный список JA, в элементе Nk которого находится номер компонент в A и IA, с которых начинается описание столбца Nk матрицы A.

1. Смоделировать операцию умножения вектора-строки и матрицы, хранящихся в этой форме, с получением результата в той же форме.

2. Произвести операцию умножения, 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.

**Интерфейс:**



При запуске программы появляется меню, которое поясняет работу программы и предлагает пользователю выбрать один из 4х вариантов работы.

**Входные данные:**

Размер первой матрицы, выбор заполнения вручную или автоматически, если вручную то количество ненулевых элементов, индексы элемента в матрице и сам элемент, если автоматически то процент заполнения, размер второй матрицы, выбор заполнения вручную или автоматически, если вручную то количество ненулевых элементов, индексы элемента в матрице и сам элемент, если автоматически, то процент заполнения.

**Возможные ошибки пользователя:**

1) При некорректном вводе или выборе меню пользователь получит следующее сообщение об ошибке "Ошибочный ввод"

2) При превышении размера матрицы 300 х 300: "Введите допустимый размер

**Аварийные ситуации:**

Программа защищена от ввода нулевого количества элементов матрицы, в подобном случае она выводит повторное сообщения о требованиях к корректному вводу.

**Структуры хранения данных:**

Обычная матрица представлена в виде двумерного массива типа int

Разреженная матрица представлена в виде трех массивов:

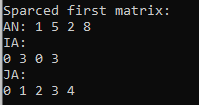
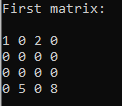
1) массив AN типа int, в котором хранятся значения ненулевых элементов.

2) массив IA типа int, который содержит номера строк для элементов массива AN.

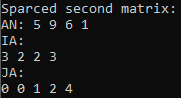
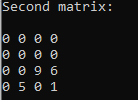
3) массив JA типа int, в элементе Nk которого находится номер компонент в AN и IA, с которых начинается описание столбца Nk матрицы A

**Представления матрицы в виде разреженной:**

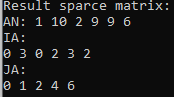
Первая Матрица:



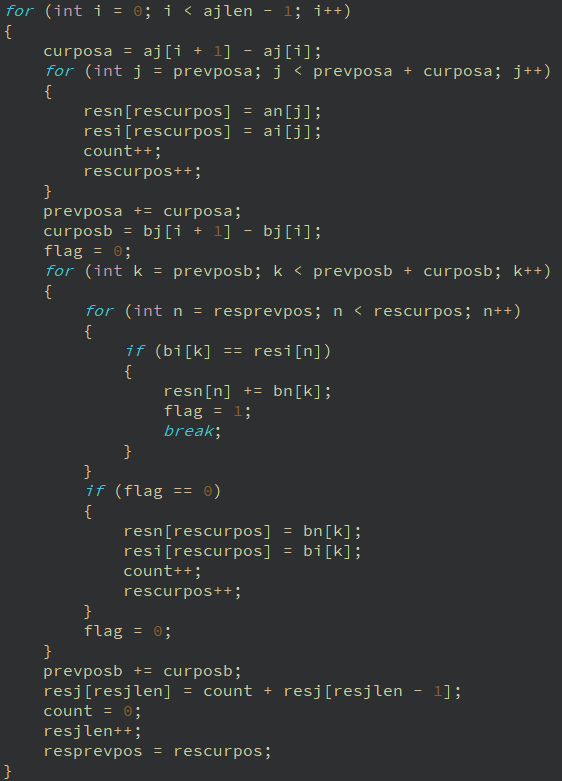
Вторая матрица:



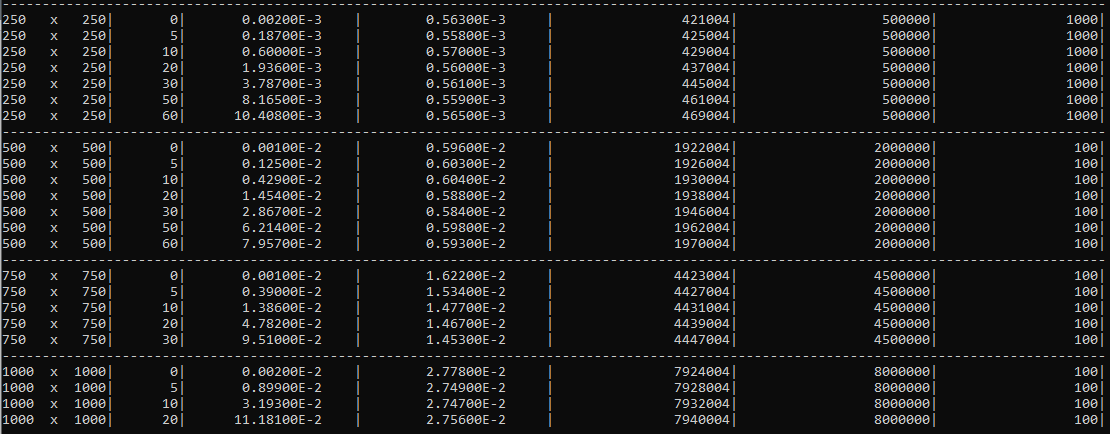
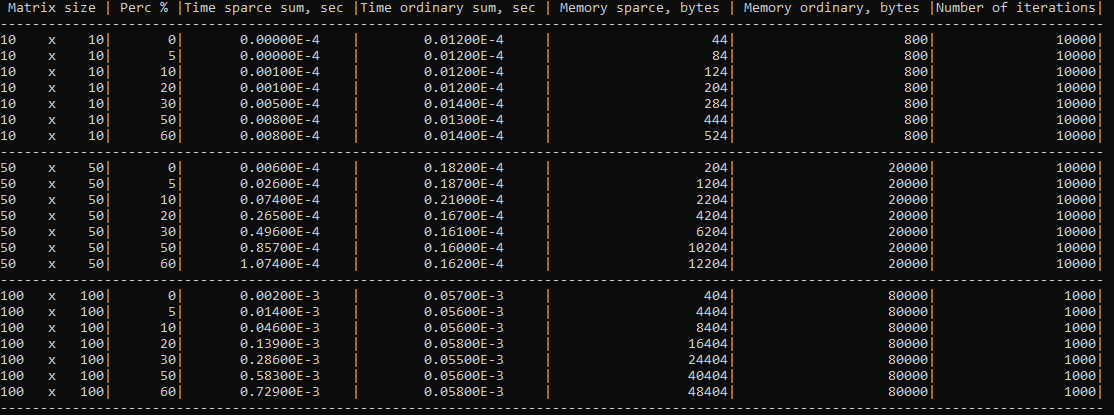
**Результат сложения матриц:**



**Алгоритм умножения:**



**Оценка эффективности:**



Таким образом,

при размерах 10x10 эффективность по времени наблюдается до 60% ненулевых элементов, эффективность по памяти 65-95%, при данном процентном заполнении

при размерах 50x50 эффективность по времени наблюдается до 10% ненулевых элементов, эффективность по памяти 39-99% при данном процентном заполнении

при размерах 100x100 эффективность по времени наблюдается до 11-12% ненулевых элементов, эффективность по памяти 40-99% при данном процентном заполнении

при размерах 250x250 эффективность по времени наблюдается до 9-10% ненулевых элементов, эффективность по памяти 35-95% при данном процентном заполнении

при размерах 500x500 эффективность по времени наблюдается до 14-15% ненулевых элементов, эффективность по памяти 40-92% при данном процентном заполнении

при размерах 1000x1000 эффективность по времени наблюдается до 8-9% ненулевых элементов, эффективность по памяти 38-90% при данном процентном заполнении

**Вывод**

Можно сделать вывод, что структуры для хранения матриц целесообразно применять только тогда, когда они разрежены – тогда это дает значительный выигрыш и по времени, и по памяти. Иначе лучше применять обычный способ хранения.

Так, если мы имеем матрицу, 20% или менее элементов которой ненулевые, которая имеет большую размерность, то с ней будет целесообразно работать. Во всех остальных случаях к этому способу прибегать не стоит, так как программа может работать очень медленно из-за сложного обращения к элементам через три структуры, может не давать выигрыша по памяти на больших размерностях.

**Вопросы**

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

Разреженная – матрица, содержащая достаточно большое количество элементов, из которых лишь малая часть является ненулевыми (n^(1+g) для матрицы размерности n, g<1).

Простейшая схема хранения разреженной матрицы: хранить массив ненулевых элементов (AN), и два массива их «координат» (I, J).

Кнут предложил хранить дополнительно массивы NR (содержит номер из AN следующего ненулевого j элемента, расположенного в матрице по строке) и NC (номера –‘’- по столбцу), а также массивы JR и JC (указатели для входа в строку и столбец). Чанг и Густавсон предложили схему разреженного строчного формата: 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?

Для хранения обычной матрицы: M \* N \* sizeof(list\_t). Память под разреженную матрицу выделяется в зависимости от схемы хранения. Память выделяется по мере наполнения ненулевыми элементами.

3. Каков принцип обработки разреженной матрицы?

Предполагает работу только с ненулевыми элементами.

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

Чем больше ненулевых элементов в матрице, тем менее эффективно использовать разряженные алгоритмы. При достижении определенного процента наполнения наблюдается значительная деградация разряженного алгоритма по времени работы