Отчёт

1. **Дисциплина: Типы и структуры данных**
2. **Лабораторная работа №3**
3. **по теме: «Обработка разреженных матриц»**
4. Работу выполнил:
5. студент группы ИУ7-32Б
6. Иванов Всеволод

# Описание условия задачи

Разработать программу умножения или сложения разреженных матриц. Предусмотреть возможность ввода данных, как с клавиатуры, так и использования заранее подготовленных данных. Матрицы хранятся и выводятся в форме трех объектов. Для небольших матриц можно дополнительно вывести матрицу в виде матрицы. Величина матриц - любая.

Сравнить эффективность (по памяти и по времени выполнения) стандартных алгоритмов обработки матриц с алгоритмами обработки разреженных матриц при различной степени разреженности матриц и различной размерности матриц.

# Описание технического задания

Программа для ввода, вывода и умножения матриц и столбцов-векторов в стандартном (массив массивов) и сжатом (массив А, JA, IA) виде.

Область применения – матрицы целых чисел любого размера и заполненности.

Сроки выполнения – 2 недели.

Основой для разработки программы послужила потребность в хранении и обработки матриц с большим количеством нулевых элементов. Для этой программы в качестве обработки было выбрано умножение матрицы на вектор-столбец.

Для реализации этой задачи использовался тип запись, содержащий тип запись, содержащий информацию о матрице с сжатом виде (элементы, позиции их столбцов, элементы, с которых начинаются ряды)

Программа должна:

* Содержать указание формата и диапазона данных при вводе
* Содержать указание операций, производимых программой
* Иметь пояснения при выводе результата
* Иметь возможность заполнения матриц вручную, по позиции и значению, по проценту нулевых элементов
* Иметь возможность вывода матриц в обычном (для небольших матриц) и в сжатом виде.
* Контролировать правильность ввода данных
* Выдавать сообщения в случае нулевой матрицы
* Отображать результаты сравнения времени и памяти работы программы при использовании алгоритма обычного и разряженного умножения на матрицах различной заполненности и различного размера.

# Этап постановки

Исходными данными являются матрица и вектор-столбец. Результатом является вектор-столбец, равный произведению исходных матриц, и сравнение алгоритмов произведения.

Обращение к программе происходит выбором функций программы путём введения их кодов в консоли. Вывод результатов работы программы также производится в консоль.

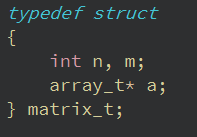
Возможными ошибками являются:

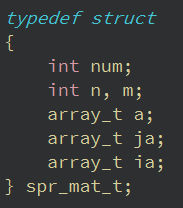
1. Попытка вывести нулевую матрицу в сжатом виде
2. Ввод некорректных размеров матриц
3. Ввод некорректных элементов матриц
4. Ввод символов
5. Ввод недопустимого процента нулевых элементов
6. Ввод несуществующий позиции в режиме ввода по позиции и значению.

Особых требований для технических средств, аппаратной совместимости, программных средств не имеется.

# Описание СД

Для хранения матриц в стандартном виде был создан тип записи matrix\_t, содержащий:

* Целые числа n, m, хранящие размер матрицы (количество рядов и столбцов соотв.)
* Двухмерного динамического массива целых чисел a, содержащего элементы матрицы

Для хранения матриц в разряженном виде был создан тип записи spr\_mat\_t, содержащий:

* Целое число num, содержащий количество ненулевых элементов матрицы
* Целые числа n, m, хранящие размер матрицы (количество рядов и столбцов соотв.)
* Динамического массива целых чисел a, хранящего ненулевые элементы матрицы их обхода по рядам
* Динамического массива целых чисел ja, хранящего позиции по столбцам соотв. элементов массива а
* Динамического массива целых чисел ia, хранящего позиции элементов в массиве а, с которых начинается строка (номер строки соотв. позиции элемента в ja)

Динамические массивы целых чисел представлены типом array\_t

# Описание алгоритма

Для умножения матрицы и вектора-столбца в обычном виде был использован следующий алгоритм. При проходе по каждой строке итогового вектора, для него вычисляется сумма поэлементного произведения ряда матрицы на вектор-столбец.

Для умножения матрицы и вектора-столбца в разряженном виде был использован следующий алгоритм. При проходе по каждой строке итогового вектора, для i-й строки:

1. В ia[i] записывается количество считанных ненулевых элементов
2. Вычисляется сумма поэлементного произведения ряда i матрицы на вектор-столбец. Производится проход по элементам вектора-столбца, и если в матрице содержится соответствующий элемент, то их произведение прибавляется к значению элемента
3. Если итоговый элемент ненулевой, то производится его добавление в вектор ответа

Оценка времени работы сортировок производится путём замера времени многократного повторения операций умножения при различных размерностях и заполненностях. Ввиду того, что функция clock() не способна зарегистрировать время выполнения одной операции, она производится 1000 раз, а потом вычисляется среднее время.

# Набор тестов

Негативные тесты

|  |  |  |
| --- | --- | --- |
| Ввод | Проверяемый случай | Вывод |
| 50 | Использование несуществующей команды | Ошибка: команда не существует |
| afa | Ввод символов | Ошибка: неправильный формат ввода |
| 0 1 | Ввод некорректных размеров матрицы | Ошибка: некорректный размер матрицы |
| \*команда обычной печати\* | Вызов печати матрицы с 30+ столбцами | Ошибка: матрица содержит более 30 столбцов |
| \*команда обычной печати\* | Вызов печати не введённой матрицы | Ошибка: матрица пуста |
| \*команда разряженной печати\* | Вызов печати не введённой или пустой разряженной матрицы | Ошибка: матрица пуста |
| -1 0 3 | Ввод элемента на несуществующую позицию (в режиме ввода по позиции и значению) | Ошибка: неправильная позиция элемента |
| 1 1 10  1 1 5 | Ввод элемента на ненулевой элемент (в режиме ввода по позиции и значению) | Ошибка: неправильная позиция элемента |
| 120 | Ввод некорректного процента заполненности | Ошибка: некорректный процент |

Тепличные тесты

|  |  |  |
| --- | --- | --- |
| Ввод | Проверяемый случай | Вывод |
| Команда: 00 | Выход из программы | Выход из программы  \*завершение работы\* |
| Команда: 01 | Вывод функций | Таблица доступных функций |
| Команда: 10  text.txt | Ввод матрицы стандартным методом | Введите размерs матрицы: \_\_ \_\_  Введите матрицу:  …  Матрица введена успешно |
| Команда: 11 | Ввод ненулевых элементов матрицы по позиции и значению | Введите размер матрицы: \_\_ \_\_  Введите число ненулевых элементов: \_  Введите элементы в формате позиция, значение:  …  Матрица введена успешно |
| Команда: 12 | Ввод матрицы по проценту заполненности | Введите размер матрицы: \_\_ \_\_  Введите процент нулевых элементов:\_\_  …  Матрица введена успешно |
| Команда: 20 | Обычная печать матрицы | \*Матрица в стандартном виде\* |
| Команда: 21 | Печать разряженной матрицы | \*Матрица в сжатом виде\* |
| Команда: 30, 31 | Произведение матрицы на вектор-столбец (стандартное и в сжатом виде) | Введите способ ввода столбца: \_\_  Введите вектор-столбец:  \*Вывод вектора-столбца\*  \*Вывод матрицы произведения\* |
| Команда: 40 | Сравнение затраченного времени и памяти алгоритмов | Время и память, затраченные на умножение матриц  Сравнение в процентах эффективности алгоритмов умножения и матриц |

Граничные тесты

|  |  |  |
| --- | --- | --- |
| Ввод | Проверяемый случай | Вывод |
| Команда: 20 | Стандартный вывод матрицы с 30 столбцами | \*Вывод матрицы\* |
| Команда: 21 | Вывод разряженной матрицы с одним элементом | \*Вывод матрицы\* |
| Команда: 12  0 | Ввод матрицы без нулевых элементов с помощью случайной генерации | Матрица введена успешно |
| Команда: 12  100 | Ввод матрицы без ненулевых элементов с помощью случайной генерации | Матрица введена успешно |

# Сравнение эффективностей

(Умножение производилось на вектор-столбец с 90% значащих цифр)

Умножение разряженных матриц по сравнению с стандартным умножением дало выигрыш в:

Матрица 100х100:

|  |  |  |
| --- | --- | --- |
| Заполненность | Время | Память |
| 1 элемент | 105 раз | 100 раз |
| 10% | 7 раз | 5 раз |
| 50% | +10% | Одинаково |

Матрица 1000х1000:

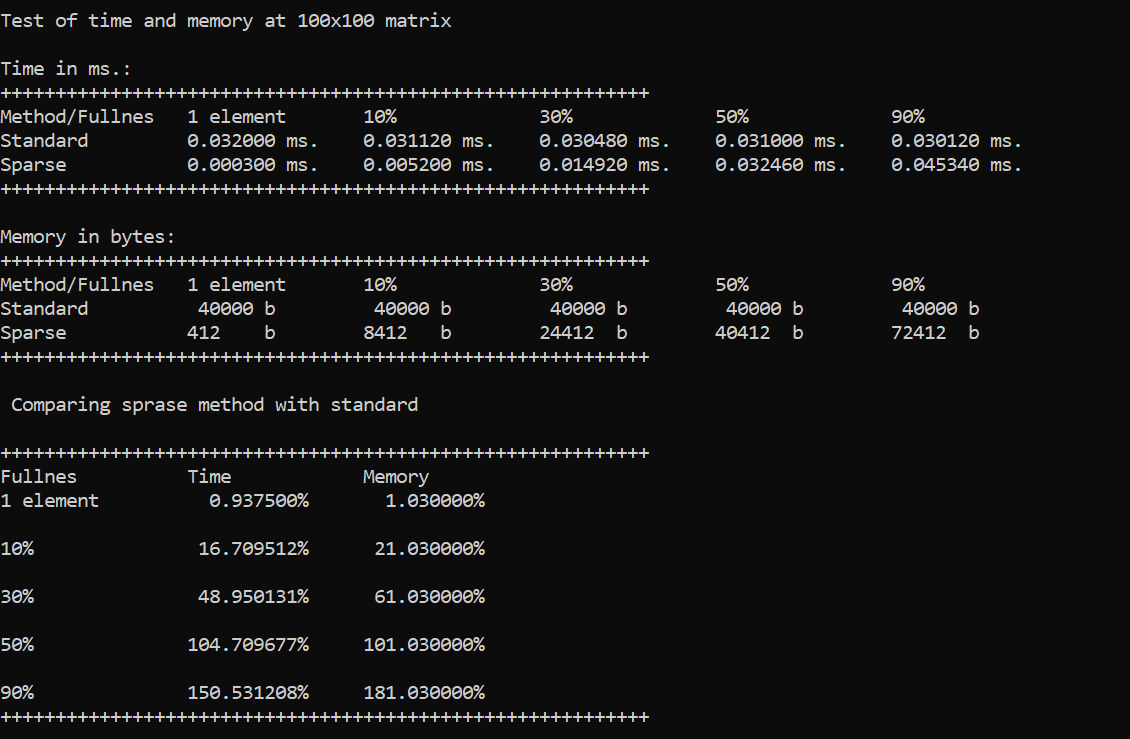
|  |  |  |
| --- | --- | --- |
| Заполненность | Время | Память |
| 1 элемент | 700 раз | 1000 раз |
| 10% | 7 раз | 5 раз |
| 60% | -20% | +20% |

Матрица 2000х2000:

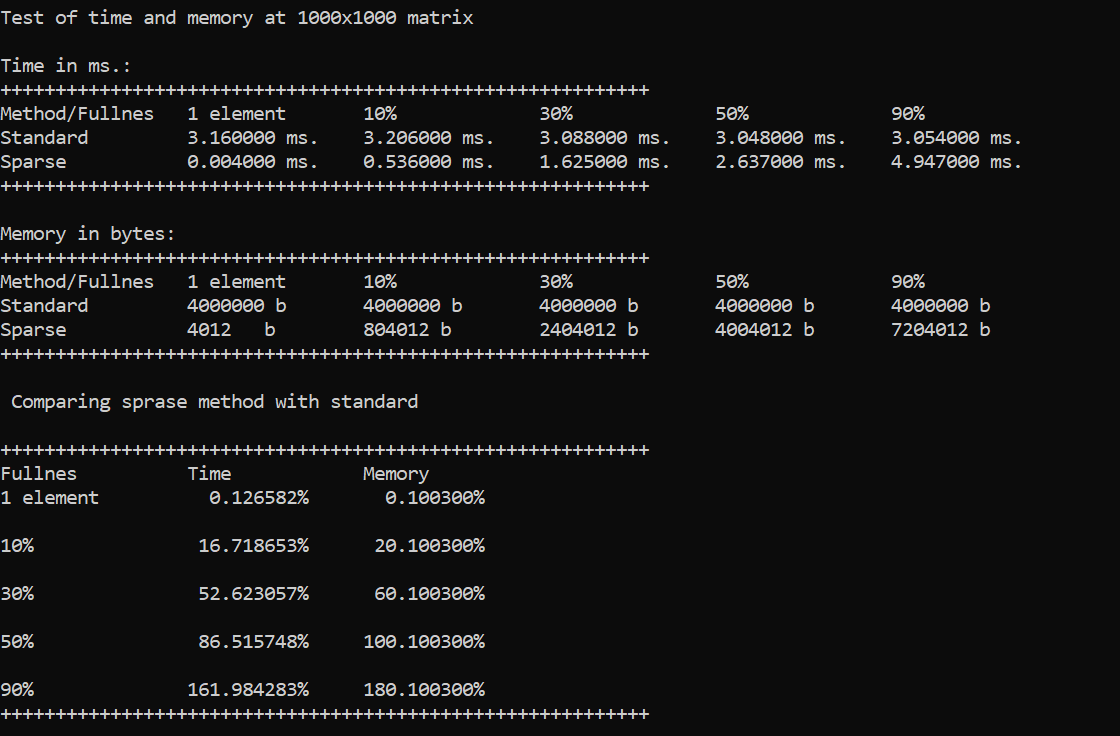
|  |  |  |
| --- | --- | --- |
| Заполненность | Время | Память |
| 1 элемент | 1200 раз | 2000 раз |
| 10% | 6.6 раз | 5 раз |
| 60% | Одинаково | +20% |

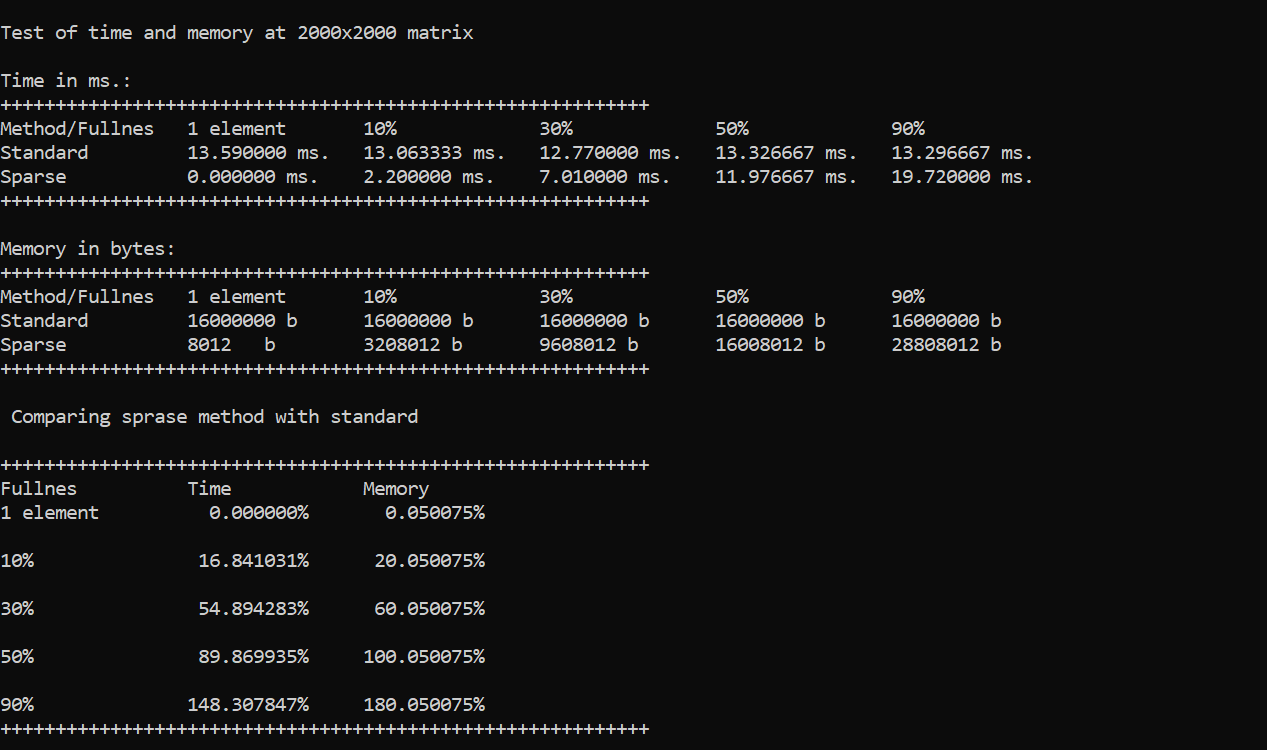
Данные анализа:

Приложение №1 (оценка эффективности при матрице 100x100)



Приложение №2 (оценка эффективности при матрице 1000x1000)



Приложение №3 (оценка эффективности при матрице 2000x2000) 

# Ответы на вопросы

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

Разряженная матрица – матрица порядка n, которая содержит ≤ n1+(0.3±0.1)

ненулевых элементов. То есть это такая матрица, в которой существенное количество элементов являются нулевыми.

Схемы хранения таких матриц основаны на том, что в памяти хранятся только ненулевые элементы и вспомогательные массивы для однозначного определения их позиции. Такими схемами являются:

* Схема хранения Кнута (A, I, J, NR, NC, JR, JC)
* Координатная схема хранения (A, I, J)
* Схема хранения в разряженном строчном формате (A, JA, IA)
* Схема хранения Деннил (для оболочной матрицы) (A, AN, IA)

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

(N, M – размеры матрицы, NUM – количество ненулевых элементов)

Под хранение обычной матрицы выделяется память под N\*M элементов в виде двухмерного массива элементов.

Объём памяти, выделяемый на хранение разряженной матрицы, зависит от метода хранения. В основном выделяется память под:

NUM ненулевых элементов матрицы, в виде массива элементов A

NUM позиций соотв. элементов матрицы, в виде массива (массивов позиций может быть несколько)

N и/или M позиций элементов в A, с которых начинается/заканчивается строка или столбец, в виде связанного списка или массива.

3. Каков принцип обработки разреженной матрицы?

Принцип обработки разряженной матрицы заключается в поиске для каждого элемента первой матрицы ненулевых элементов во второй матрице и выполнения с ними определённых действий. Для ускорения поиска может быть использован массив IP, хранящий указатели на элементы с соответствующим индексом.

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

Стандартные алгоритмы обработки матриц эффективнее применять для неразреженных матриц. Потому в таком случае алгоритмы обработки разреженных матриц будут работать дольше и требовать больше памяти.

Эффективность использования того или иного алгоритма зависит в основном от степени разреженности матриц.

# Выводы по проделанной работе

Из проведённого анализа следует, что использование данной схемы хранения и умножения матриц дают ощутимый выигрыш по памяти и времени для матриц ≤40% заполненности. Для матриц 40-60% заполненности время память примерно одинаковые. Для матриц >60% заполненности такая схема хранения и обработки проигрывает как в памяти, так и во времени обработки. Из чего следует вывод, что выбранные средства хранения и обработки эффективны для матриц с <40% ненулевых элементов.