Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ»

КАФЕДРА «ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ЭВМ И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ»

Лабораторная Работа №3 «Обработка разреженных матриц»

Вариант №3

Студент **Поляков Андрей Игоревич**

Группа **ИУ7-32Б**

Название предприятия **НУК ИУ МГТУ им. Н. Э. Баумана**

Студент **Поляков А.И.**

Оценка _____

Описание условия задачи

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

- вектор A содержит значения ненулевых элементов;
- вектор JA содержит номера столбцов для элементов вектора A;
- вектор IA, в элементе Nk которого находится номер компонент в A и JA, с которых начинается описание строки Nk матрицы A.

1. Смоделировать операцию умножения матрицы и вектора-столбца, хранящегося в форме вектора A и вектора, содержащего номера строк ненулевых элементов, с получением результата в форме хранения вектора-столбца.
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.

Техническое задание

Исходные данные:

Файл с матрицей в координатном виде.

Файл с вектором-столбцом в координатном виде.

Пункты меню, выраженные целыми числами 0-12. Внутри некоторых пунктов вводятся строки и числа.

1) Считать матрицу из файла
2) Ввести матрицу
3) Считать вектор-столбец из файла
4) Ввести вектор-столбец
5) Вывести матрицу
6) Вывести вектор-столбец
7) Произвести умножение матрицы на вектор-столбец, применяя алгоритм работы с разреженными матрицами
8) Произвести умножение матрицы на вектор-столбец, применяя стандартный алгоритм работы с матрицами
9) Сравнить алгоритмы
10) Сохранить матрицу в файл

11) Сохранить вектор-столбец в файл

12) Вывести информацию о программе

0) Выход

Результат:

Матрица, вектор-столбец, результат умножения матрицы на вектор – вектор-столбец, результаты замеров времени работы алгоритмов и оценки их эффективности

Описание задачи:

Чтение, сохранение, вывод, умножение, матрицы и вектора, оценка эффективности алгоритмов.

Способ обращения к программе:

Запуск с помощью ./app.exe

Аварийные ситуации и ошибки:

1. ERR_IO (Ошибка ввода): Введенные данные не соответствуют ожидаемому формату.

2. ERR_RANGE (Ошибка размера): Значение выходит за допустимые пределы.

3. ERR_MEM (Память): Ошибка выделения памяти.

4. ERR_ARGS (Ошибка аргументов командной строки): Ошибка в аргументах, переданных через командную строку.

5. ERR_NO_FILE (Файл отсутствует): Указанный файл не найден.

6. ERR_EXIT: Программа завершает выполнение.

7.ERR_WRONG_COMMAND (Некорректная команда): Введена некорректная команда.

8. ERR_NO_MTR (Нет матрицы) Не введена матрица.

9. ERR_NO_V (Нет вектора) Не введен вектор.

Описание внутренних структур данных

```
// Структура для представления элементов разреженной матрицы
typedef struct
{
    double *A;    // Массив значений ненулевых элементов
    size_t *JA;      // Массив номеров столбцов для элементов в A
    size_t *IA;      // Массив, указывающий на начало каждой строки в
A и JA
    size_t rows;     // Количество строк в матрице
    size_t cols;     // Количество столбцов в матрице
    size_t nz;       // Количество ненулевых элементов
} mtr_t;

// Структура для представления разреженного вектора-столбца
typedef struct
{
    double *A;    // Массив значений ненулевых элементов
    size_t *IA;      // Массив номеров строк для элементов в A
    size_t rows;     // Количество строк в векторе-столбце
    size_t nz;       // Количество ненулевых элементов
} vector_t;
```

1. mtr_t (Sparse Matrix Structure):
    - double *A: Указатель на массив значений ненулевых элементов матрицы.
    - size_t *JA: Указатель на массив номеров столбцов для соответствующих элементов в A.
    - size_t *IA: Указатель на массив, указывающий на начало каждой строки в A и JA.
    - size_t rows: Количество строк в матрице.
    - size_t cols: Количество столбцов в матрице.
    - size_t nz: Количество ненулевых элементов в матрице.

    Эта структура позволяет эффективно хранить разреженные матрицы, так как она хранит только ненулевые элементы и их позиции.

2. vector_t (Sparse Column Vector Structure):
    - double *A: Указатель на массив значений ненулевых элементов вектора-столбца.
    - size_t *IA: Указатель на массив номеров строк для соответствующих элементов в A.
    - size_t rows: Количество строк в векторе-столбце.
    - size_t nz: Количество ненулевых элементов в векторе-столбце.

Эта структура предназначена для представления разреженного вектора-столбца, где также хранятся только ненулевые элементы и их позиции.

```
double **mtr;

size_t n, m;
```

Матрица представлена в виде массива указателей на строки матрицы.

n и m – размеры матрицы

Описание алгоритма

1. Отобразить пользователю список команд и дождаться ввода номера нужной команды.
2. Команды считывания из файла выделяет память и считывает матрицу/вектор из файла.
3. Команды ввода вручную выделяет память и считывает матрицу/вектор из stdin.
4. Команды записи в файл записывает матрицу/вектор в файл.
5. Команды вывода выводят матрицу/вектор в консоль.
6. При выборе умножения, применяя алгоритм работы с разреженными матрицами:
    1. Проверка совместимости размеров.
    2. Выделение памяти для результата.
    3. Перебор строк матрицы.
    4. Для каждой строки, суммирование произведений ненулевых элементов этой строки и соответствующих элементов вектора.
    5. Если сумма ненулевая, она записывается в результирующий вектор.
    6. Возврат результата.
7. При выборе умножения, применяя стандартный алгоритм работы с матрицами:
    1. Перевод матрицы и вектора в типичное матричное представление.
    2. Проверка совместимости размеров.
    3. Выделение памяти для результата.
    4. Пермножение матриц.

5. Возврат результата.

8. При выборе замеров, произвести умножение двумя методами, измерив время. Оценить относительную эффективность программы (в процентах) по времени и по используемому объему памяти в зависимости от используемого алгоритма и от объема информации.
9. При выборе вывода информации о программе, вывести всю необходимую информацию.
10. При выборе выхода, закрыть файл и завершить программу.

Тестовые данные

Позитивные тесты

Тест	**Входные данные**	**Результат**
Ввести матрицу из файла	Имя файла	Разреженная матрица
Ввести матрицу вручную	Элементы матрицы	Разреженная матрица
Ввести столбец из файла	Имя файла	Разреженный столбец
Ввести столбец вручную	Элементы столбца	Разреженный столбец
Вывести матрицу	Разреженная матрица	Матрица в координатном виде в консоли
Вывести матрицу в файл	Разреженная матрица	Матрица в координатном виде в файле
Вывести столбец	Разреженный столбец	Столбец в координатном виде в консоли
Вывести столбец в файл	Разреженный столбец	Столбец в координатном виде в файле
Произвести умножение матрицы и столбца корректных размерностей, применяя алгоритм работы с разреженными матрицами	Разреженные матрица и столбец	Вектор-столбец – результат умножения
Произвести умножение матрицы и столбца корректных размерностей, применяя стандартный алгоритм	Матрица и столбец	Вектор-столбец – результат умножения

работы с матрицами		
Произвести умножения двумя методами и сравнить их эффективность	Матрица и столбец	Результаты выполнения замеров и сравнения алгоритмов

Негативные тесты

Неверная команда в меню	23	Некорректная комманда
Слишком длинная строка	Имя файла длины 60	Ошибка размера
Файл не существует	Попытка прочитаь не существующий файл	Файл отсутствует
Ввод нецелочисленных данных для матрицы	Нецелые числа	Ошибка ввода
Ввод некорректных данных для размерности матрицы	Некорректная размерность матрицы	Ошибка размера
Ввод некорректных данных для размерности вектора	Некорректная размерность вектора	Ошибка размера
Попытка умножения разреженной матрицы на вектор с неправильными размерами	m матрицы не равно n вектора	Ошибка размера
Ввод матрицы с нулевой размерностью	m матрицы = 0	Ошибка размера

Замеры

Программа замеряет время умножения разными алгоритмами. Для каждого из тестов программа выполняет многочисленные замеры, пока их RSE не становится <5. Затем все тестовые случаи сравниваются и оценивается их эффективность.

Размер матрицы: 1000x1000 = **1000000** элементов
1 % матрицы - ненулевые элементы

Умножение матрицы на вектор-столбец, применяя алгоритм работы с разреженными матрицами:

Кол-во элементов	Время, нс	Кол-во итераций	RSE
1000000	966994.45	20	4.52

Занимаемая паямть - 168056 байт

Умножение матрицы на вектор-столбец, применяя стандартный алгоритм:

Кол-во элементов	Время, нс	Кол-во итераций	RSE
1000000	3482853.50	10	4.19

Занимаемая паямть - 8008000 байт
Разность = 2515859.05 нс

Обработка разреженной матрицы быстрее на **72.235569 %**

Размер матрицы: 1000x1000 = 1000000 элементов
2.984300% матрицы - ненулевые элементы

Умножение матрицы на вектор-столбец, применяя алгоритм работы с разреженными матрицами:

Кол-во элементов	Время, нс	Кол-во итераций	RSE
1000000	2730518.40	10	4.24

Занимаемая паямть - 485544 байт
Умножение матрицы на вектор-столбец, применяя стандартный алгоритм:

Кол-во элементов	Время, нс	Кол-во итераций	RSE
1000000	3436449.60	10	0.51

Занимаемая паямть - 8008000 байт
Разность = 705931.20 нс

Обработка разреженной матрицы быстрее на 20.542458 %

Размер матрицы: 1000x1000 = 1000000 элементов
4.976500% матрицы - ненулевые элементы

Умножение матрицы на вектор-столбец, применяя алгоритм работы с разреженными матрицами:

Кол-во элементов	Время, нс	Кол-во итераций	RSE
1000000	4404824.50	10	1.64

Занимаемая паямть - 804296 байт
Умножение матрицы на вектор-столбец, применяя стандартный алгоритм:

Кол-во элементов	Время, нс	Кол-во итераций	RSE
1000000	3454972.50	10	0.41

Занимаемая паямть - 8008000 байт

Разность = 949852.00 нс

Обработка обычной матрицы быстрее на 21.563901 %

Размер матрицы: 1000x1000 = 1000000 элементов
14.929500% матрицы - ненулевые элементы

Умножение матрицы на вектор-столбец, применяя алгоритм работы с разреженными матрицами:

Кол-во элементов	Время, нс	Кол-во итераций	RSE
1000000	13034288.30	10	0.35

Занимаемая паямть - 2396776 байт
Умножение матрицы на вектор-столбец, применяя стандартный алгоритм:

Кол-во элементов	Время, нс	Кол-во итераций	RSE
1000000	3467983.80	10	0.65

Занимаемая паямть - 8008000 байт
Разность = 9566304.50 нс

Обработка обычной матрицы быстрее на 73.393378 %

Размер матрицы: 1000x1000 = 1000000 элементов
49.750000% матрицы - ненулевые элементы

Умножение матрицы на вектор-столбец, применяя алгоритм работы с разреженными матрицами:

Кол-во элементов	Время, нс	Кол-во итераций	RSE
1000000	43615882.10	10	0.46

Занимаемая паямть - 7968056 байт
Умножение матрицы на вектор-столбец, применяя стандартный алгоритм:

Кол-во элементов	Время, нс	Кол-во итераций	RSE
1000000	3714887.40	10	2.66

Занимаемая паямть - 8008000 байт
Разность = 39900994.70 нс

Обработка обычной матрицы быстрее на 91.482719 %

Размер матрицы: 1000x1000 = **1000000** элементов
69.646900% матрицы - ненулевые элементы

Умножение матрицы на вектор-столбец, применяя алгоритм работы с разреженными матрицами:

Кол-во элементов	Время, нс	Кол-во итераций	RSE

1000000	60653573.20	10	0.40

Занимаемая паямть - 11151560 байт

Умножение матрицы на вектор-столбец, применяя стандартный алгоритм:

Кол-во элементов	Время, нс	Кол-во итераций	RSE
1000000	3437005.40	10	0. 21

Занимаемая паямть - 8008000 байт
Разность = 57216567.80 нс
Обработка обычной матрицы быстрее на **94.333383 %**

Ответы на вопросы

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

Разреженная матрица содержит большинство нулевых элементов. Она применяется в задачах обработки изображений, анализа графов и решения систем уравнений.

Существуют разные методы хранения таких матриц, например, линейный связный список, диагональная схема для симметричных матриц, а также различные связные схемы.

Наиболее распространенный метод - "разреженный строчный формат" Чанга и Густавсона. Этот метод требует минимальное количество памяти и подходит для различных операций, таких как сложение, умножение матриц, перестановка строк и столбцов, транспонирование и решение систем уравнений.

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

Если матрица достаточно большая, разреженная матрица требует меньше памяти, так как хранит только ненулевые элементы, в отличие от обычной матрицы, которая выделяет память для всех элементов независимо от их значения. Размер памяти для разреженной матрицы зависит от выбранной схемы хранения и структуры матрицы.

3. Каков принцип обработки разреженной матрицы?

Работа с разреженной матрицей отличается от работы с обычной. Главный подход в обработке разреженной матрицы заключается в исключении операций с нулевыми элементами, что снижает вычислительные и памятные затраты.

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

Эффективнее всего применять стандартные алгоритмы обработки матриц, когда они имеют большинство ненулевых элементов и небольшие размеры. Это зависит от того, насколько "плотной" является разреженная матрица. Чем больше ненулевых элементов, тем менее эффективно использовать схему для хранения разреженной матрицы. В случае плотной матрицы или когда ненулевых элементов мало, применение стандартных алгоритмов может оказаться более эффективным.

Вывод

В ходе лабораторной работы мы провели сравнение времени выполнения операций умножения и объема памяти, занятого разреженными и стандартными матрицами. Мы исследовали различные уровни разреженности матриц и оценили их влияние на эффективность разреженных матриц.

Время выполнения операций над разреженными и стандартными матрицами зависит от размеров матрицы и её степени разреженности. Разреженные матрицы требуют значительно меньше памяти, особенно при высокой степени разреженности. Однако, эффективность разреженных матриц может снижаться с увеличением размеров матрицы или степени её разреженности.
В случае реализованного мною алгоритма, обработка разреженных матриц выигрывает до 3% разреженности. А по памяти хранение матрицы в разреженном виде выигрывает примерно до 50% разреженности.