**Отчет**

**Лабораторная работа №3**

**“Обработка разреженных матриц”**

Выполнил: Елгин Илья

Группа: ИУ7-34

Вариант 6

**Задача:**

Создать разреженную матрицу, которая 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. Обеспечить возможность ввода матриц вручную. Сравнить время выполнения стандартного алгоритма обработки матриц и алгоритма обработки разреженных матриц при различной заполненности матриц (от 1 элемента до 100% заполненности). Сравнить объем требуемой памяти для реализации стандартного алгоритма обработки матриц и алгоритма обработки разреженных матриц при различном проценте заполнения матриц и при различном их размере. Смоделировать операцию умножения вектора-строки и матрицы, применяя разные алгоритмы.

**Входные данные:**

Файл с записанной заранее матрицей.

Строка (имя файла).

Пункт меню:

1 - Умножение вектора-строки на матрицу, хранящихся в разреженной форме, с получением результата в той же форме.

2 - Умножение стандартным алгоритмом работы с матрицами.

3 - Сравние времени выполнения операций и объем памяти при использовании этих 2-х алгоритмов при различном проценте заполнения матриц.

4 - Вывести матрицу в разреженной форме.

5 - Вывести матрицу в простой форме.

0 – Выход

Числа, введённые с консоли.

Матрица в координатной форме.

**Выходные данные:**

Матрица в простом или разреженном виде.

Вектор в простом или разреженном виде.

Результаты измерений времени и памяти.

**Способ обращения к программе:**

**Обращение к программе**:

Исполняемый файл app.exe . Запускается из командной строки, аргументы строки не требуются.

**Аварийные ситуации и ошибки пользователя:**

|  |  |
| --- | --- |
| Ошибка | Результат |
| Ошибка ввода опции меню | “Некорректный ввод. Повторите. ” |
| Ввод строки вместо числа | “Ошибка ввода” |
| В результате операции получилась нулевая матрица | “В матрице все элементы нулевые” |
| Ввод числа, выходящего за указанные пределы | “Неверный размер” |
| Произошла ошибка при выделении динамической памяти | Завершение программы с сообщением: “Ошибка при выделении памяти (программа завершена)” |

**Внутренние структуры данных:**

Сначала создается матрица указанного размера, затем по матрице создаются массивы A, IA и список JA. При умножении создаются аналогичные массивы для вектора-строки и результирующего вектора-строки.

Матрица в обычном формате

int \*\*matrix;

Матрица в виде векторов.

int \*A = malloc(count \* sizeof(int));

int \*IA = malloc(count \* sizeof(int));

struct ja JA;

struct ja

{

int num;

struct ja \*next;

};

**Алгоритм:**

Выбор пользователем способа ввода матрицы:

1: ввод матрицы вручную

2: считывание матрицу из файла

3: генерация случайной матрицы по размеру и разрежености

создание и заполнение векторов A, IA и JA

пока не выбран 0:

1: если матрица не нулевая

Умножение матрицы в разряженном виде на вектор-строку

иначе вывести сообщение

2: если матрица не нулевая

умножение матрицы обычным способом

иначе вывести сообщение

3: вывод таблицы сравнения времени и памяти

4: если матрица не нулевая

вывод матрицы в разреженной форме

иначе вывести сообщение

5: вывод матрицы в порстой форме

Алгоритм умножения матрицы на вектор в разреженном виде:

Создаём результирующий нулевой вектор AR

Проходим по индексам столбцов матрицы

если JA[i] <= JA[i+1](столбец не пустой) то

проходимся по строкам

если IA[k] равно номер строки (есть элемент строки)

если JAS[номер строки] не= JAS[номер строки+1](есть соответствующий элемент вектора) то

AR[r] += AS[JAS[номер строки]] \* A[k];

k = k + 1

Удаляем все нулевые элементы из AR и выводим его

Алгоритм умножения матрицы на вектор в простом виде:

Создаём результирующий нулевой вектор AR

Проходим по строкам матрицы

Проходим по столбцам матрицы

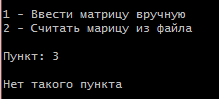
Если элемент M[i][j] не = 0 и соответствующий элемент вектора не = 0 то

AR[r] += AS[JAS[номер строки]] \* A[k];

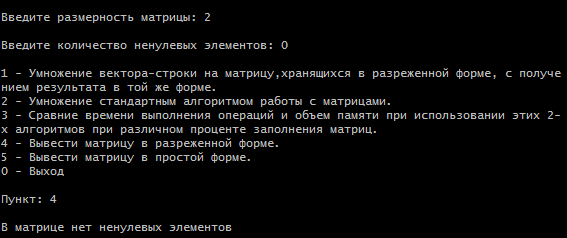
Вывод AR

**Тесты:**

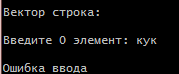
1. Ошибка ввода опций меню



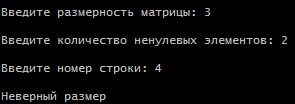
1. Вывод нулевой матрицы в разреженной форме



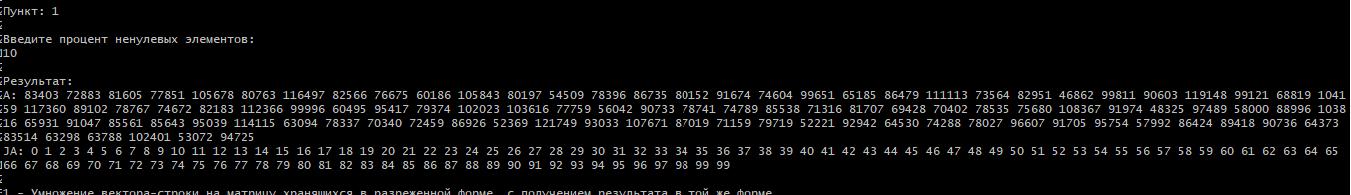
1. Ввод строки вместо числа



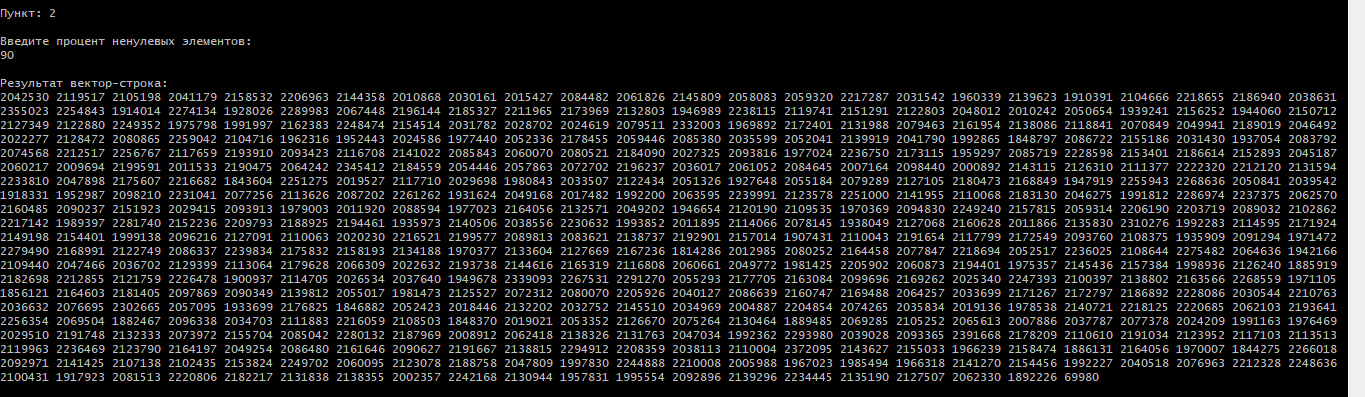
4) Ввод числа (индекса элемента), выходящего за пределы



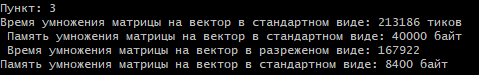
5)Умножение в разреженной форме



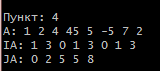
6)Умножение в простой форме



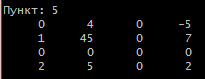
7) Вывод сравнения времени и памяти



8) Вывод матрицы в разреженной форме



9) Вывод матрицы в простой форме



**Оценка эффективности:**

|Размерность|Заполнение| Время, тики | Память, байт |

| | |----------------------------------------

| матрицы | % |Простая |Разреженная |Простая|Разреженная|

----------------------------------------------------------------

| | 1 | 2728 | 950 | | 136 |

| 10 | 5 | 3098 | 1148 | 440 | 168 |

| | 20 | 2622 | 3656 | | 288 |

| | 50 | 2438 | 4266 | | 528 |

| | 100 | 2736 | 11872 | | 929 |

-----------------------------------------------------------------

| | 1 | 233266 | 49634 | | 2008 |

| 100 | 5 | 221750 | 69206 | 40400 | 5208 |

| | 20 | 241552 | 247701 | | 17208 |

| | 50 | 261890 | 382578 | | 41208 |

| | 100 | 287758 | 592876 | | 81208 |

----------------------------------------------------------------

| | 1 | 3442866 | 1990853 | | 26008 |

| 500 | 5 | 3701166 | 2330358 |1002000| 106008 |

| | 20 | 3891272 | 4125086 | | 406008 |

| | 50 | 3887616 | 7917034 | | 1006008 |

| | 100 | 4399747 | 12243552 | | 2006008 |

----------------------------------------------------------------

| | 1 | 19010867 | 5534724 | | 92008 |

| 1000 | 5 | 19275342 | 7228768 |4004004| 412008 |

| | 20 | 19510837 | 19899493 | | 1612008 |

| | 50 | 20013059 | 30859038 | | 4012008 |

| | 100 | 21388351 | 50426903 | | 8012028 |

----------------------------------------------------------------

Таким образом, использование разреженной матрицы выгодно при малом проценте заполнения и большой размерности матрицы. Мы получаем выигрыш по памяти примерно при заполнении матрицы до 20-40% и по времени при заполнении до 10%.

**Ответы на вопросы.**

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

Разреженная матрица – матрица с малым количеством ненулевых элементов.

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

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

Для обычной матрицы выделяется память N\*M\*sizeof(element), где N\*M – размер матрицы. В разреженной же таблице количество выделяемой памяти зависит от метода её хранения. Зачастую (количество ненулевых элементов \* (sizeof(element) + индекс столбца + индекс строки))

3. Каков принцип обработки разреженной матрицы?

Алгоритмы обработки разреженных матриц предусматривают действия только с ненулевыми элементами и, таким образом, количество операций будет пропорционально количеству ненулевых элементов.

4. В каком случае для матриц эффективнее применять стандартные

алгоритмы обработки матриц? От чего это зависит?

Когда процент заполнения высок, эффективнее применять стандартные алгоритмы обработки матриц, т.к. при использовании разряженного алгоритма увеличивается память и время обработки.

**Вывод:**

В процессе выполнения работы выяснено, что использование разреженной матрицы выгодно при малом проценте заполнения и большой размерности матрицы. Мы получаем выигрыш по памяти примерно при заполнении матрицы до 20-40%, и по времени при заполнении до 10%. При заполнении матрицы менее, чем на 7% мы используем памяти до 85% меньше, а также уменьшаем время выполнения операций.