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

Отчёт по лабораторной работе №3 «Обработка разреженных матриц.»

Студент Шелия София Малхазовна

Группа ИУ7 – 35Б

2020 г.

Цель работы: реализация алгоритмов обработки разреженных матриц, сравнение эффективности использования этих алгоритмов со стандартными алгоритмами обработки матриц при различном размере матриц и степени их разреженности.

1. Описание условия задачи.

Вариант 1
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; - 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. Техническое задание

Исходные данные и результат

Ввод:

1. Пользователь вводит цифру (1 или 2), в зависимости от выбранного варианта заполнения матрицы (1 – ручное заполнение, 2 – автоматическое заполнение)
2. Пользователь вводит целое положительное число - количество строк матрицы.
3. Пользователь вводит целое положительное число – количество столбцов матрицы.
4. Если выбрано автоматическое заполнение, то пользователь вводит целое неотрицательное число <=100 – процент заполнения матрицы
5. Пользователь вводит цифру от 1 до 4, где
    - 1 – Вывод матриц в обычном виде
    - 2 – Вывод матриц в виде массивов
    - 3 – Вывод матриц обоими способами
    - 4 – Завершение программы, без вывода матриц

Вывод:

1. Программа выводит время выполнения сложения матриц с помощью стандартного алгоритма и специального.
2. В зависимости от выбора пользователя (пункт 5 ввод) выводит или не выводит матрицы.

Описание задачи, реализуемой программой

1. При автоматическом заполнении программа создает две матрицы с заданным процентном заполнения, при ручном – заполняет матрицы, введенными значениями.

2. Выполняет сложение матриц стандартным алгоритмом.
3. Выполняет сложение матриц специальным алгоритмом.
4. Выводит время сложения матриц для разных алгоритмов.
5. Выводит или не выводит матрицы (в зависимости от выбора пользователя)

Способ обращения к программе

Запуск приложения возможен через терминал MSYS2, а именно.

```
1. gcc -std=c99 -Wall -Werror -c *.c
2. gcc -o main.exe *.o
3. ./main.exe
```

Описание возможных аварийных ситуаций и ошибок пользователя

Ошибки пользователя при выборе пункта меню

- Пустой ввод
- Вводятся посторонние символы (например, буквы)
- Вводится отрицательное число
- Вводится не целое число
- Вводится отсутствующий пункт меню

Ошибки пользователя при вводе количества строк и столбцов

- Пустой ввод
- Вводятся посторонние символы (например, буквы)
- Вводится отрицательно число
- Вводится не целое число

Ошибки пользователя при вводе процента заполнения матриц

- Пустой ввод
- Вводятся посторонние символы (например, буквы)
- Вводится отрицательно число
- Вводится не целое число
- Вводится число больше 100

Ошибки пользователя при вводе элементов матрицы

- Пустой ввод
- Вводятся посторонние символы (например, буквы)
- Вводится не целое число

Аварийные ситуации.

- Сложение двух пустых матриц

3. Описание внутренних структур данных

Для хранения матрицы в обычном виде я использую следующую структуру:

```
struct usual_matrix{
    int **matrix; //Элементы матрицы
    int lines; //Количество строк матрицы
    int columns; //Количество столбцов матрицы

};
```

Для хранения матрицы в специальном виде я использую следующую структуру:

```
struct sparse_matrix{
    int lines; //Количество строк матрицы
    int columns; //Количество столбцов матрицы
    int num_non_zero_elem; //Количество ненулевых элементов
матрицы

    int *elements_value; //Значения ненулевых элементов матрицы
    int *column_index; //Индексы столбцов соответствующих
элементов
    int *line_index; //Индексы элементов, с которых начинаются
строки матрицы
};
```

4. Описание алгоритма

Для решения задачи нужно было реализовать два алгоритма сложения матриц.

1. Обычное сложение матриц

Программа циклически проходит по столбцам и строкам матрицы, выполняя сложение соответствующих элементов. Сложность $O(N^2)$

```
//Обычное сложение матриц
void addition_of_ordinary_matrices(struct usual_matrix *matrix_1, struct usual_matrix
*matrix_2, struct usual_matrix *result_matrix)
{
    for (int i = 0; i < result_matrix->lines; i++)
    {
        for (int j = 0; j < result_matrix->columns; j++)
            result_matrix->matrix[i][j] = matrix_1->matrix[i][j] + matrix_2-
>matrix[i][j];

    }
}
```

2. Сложение матриц, представленных с помощью массивов.

Для реализации данного алгоритма мною были использованы несколько дополнительных функций.

Функция добавления числа в матрицу

```
void add_items(struct sparse_matrix *result_matrix, struct sparse_matrix *matrix, int *result_count, int *count, int *line_index_result_count)
```

Функция поиска “пары” для элемента

```
void find_the_pairs(struct sparse_matrix *matrix_1, struct sparse_matrix *matrix_2, struct sparse_matrix *result_matrix, int *result_count, int *count_matrix_1, int *count_matrix_2, int *second_matrix_line_end, int *line_index_result_count)
```

Функция прохода по строке матрицы

```
void passage_of_line(struct sparse_matrix *matrix_1, struct sparse_matrix *matrix_2, struct sparse_matrix *result_matrix, int *count_matrix_1, int *count_matrix_2, int *first_matrix_line_end, int *second_matrix_line_end, int *result_count, int *line_index_result_count)
```

Функция сложения двух матриц

```
int addition_of_sparse_matrices(struct sparse_matrix *matrix_1, struct sparse_matrix *matrix_2, struct sparse_matrix *result_matrix, int size)
```

Сам алгоритм:

- *Запускаем цикл, которые завершается, когда программа дойдет до последнего элемента массива значений элементов матриц*
- В цикле осуществляем проход по N строке
- count_matrix_1 и count_matrix_2 равны индексам элементов, с которых начинается N строка соответствующих матриц.
- Проверяем, есть ли в данной строке первой и второй матрицы элементы, неравные 0. Если все элементы 0 хотя бы в одной из матриц, то просто записываем в матрицу-результат значения элементов матрицы, у которой в данной строке есть элементы, неравные 0.
- Если в обоих матрицах в данной строке есть элементы, неравные 0, то мы переходим в функцию прохода по строке.
- *В функции запускается цикл, который завершается при окончании обработки строки*
    - Сравниваем значения столбцов элементов с индексами count_matrix_1 и count_matrix_2, если они равны (то есть элементы находятся в одной строке и в одном столбце), то добавляем их сумму в матрицу результат и увеличиваем count_matrix_1 и count_matrix_2 (переходим на следующие элементы)
    - Если значения неравны, то переходим в функцию поиска пары.

    В данной функции с помощью счетчика проходимся по строке матрицы, где был элемент с наименьшим значением, параллельно добавляя значения из неё в матрицу – результат (так как у данных элементов точно не будет пары). Если счетчик дошел до конца и пары не нашлось, то просто добавляем значение в матрицу – результат, иначе добавляем сумму.

5. Тесты

Негативные тесты

Входные данные	Результат	Условие проверки
Для выбора пункта меню		
Выберите вариант заполнения матриц: 1 - ручное заполнение 2 - автоматическое заполнение Выбор:	Некорректный ввод.	Пустой ввод

Выберите вариант заполнения матриц: 1 - ручное заполнение 2 - автоматическое заполнение Выбор: авв	Некорректный ввод.	Вводятся посторонние символы (например, буквы)
Выберите вариант заполнения матриц: 1 - ручное заполнение 2 - автоматическое заполнение Выбор:-5	Некорректный ввод.	Вводится отрицательное число
Выберите вариант заполнения матриц: 1 - ручное заполнение 2 - автоматическое заполнение Выбор:45.1	Некорректный ввод.	Вводится не целое число
Выберите вариант заполнения матриц: 1 - ручное заполнение 2 - автоматическое заполнение Выбор:12	Некорректный ввод.	Вводится отсутствующий пункт меню
Для ввода количества строк и столбцов		
Введите количество строк:	Некорреткный ввод. Введите количество строк:	Пустой ввод
Введите количество строк: 5 Введите количество столбцов: уау	Некорреткный ввод. Введите количество столбцов:	Вводятся посторонние символы (например, буквы)
Введите количество строк:10 Введите количество столбцов:-4	Некорреткный ввод. Введите количество столбцов:	Вводится отрицательно число
Введите количество строк:34.6	Некорреткный ввод. Введите количество строк:	Вводится не целое число
Для ввода процента заполнения матриц		
Введите приблизительный процент заполнения (целое неотрицательное число <=100):	Некорреткный ввод. Введите приблизительный процент заполнения (целое неотрицательное число <=100):	Пустой ввод
Введите приблизительный процент заполнения (целое неотрицательное число <=100): па	Некорреткный ввод. Введите приблизительный процент заполнения (целое неотрицательное число <=100):	Вводятся посторонние символы (например, буквы)
Введите приблизительный процент заполнения (целое неотрицательное число <=100): -6	Некорреткный ввод. Введите приблизительный процент заполнения (целое неотрицательное число <=100):	Вводится отрицательно число
Введите приблизительный процент заполнения (целое неотрицательное число <=100): 7.2	Некорреткный ввод. Введите приблизительный процент заполнения (целое неотрицательное число <=100):	Вводится не целое число
Введите приблизительный процент заполнения (целое неотрицательное число <=100): 101	Некорреткный ввод. Введите приблизительный процент заполнения (целое неотрицательное число <=100):	Вводится число больше 100
Для ввода элементов матрицы		

Введите элемент строки 1, столбца 1 (целое число):	Некорреткный ввод. Введите значение повторно:	Пустой ввод
Введите элемент строки 1, столбца 1 (целое число): пап	Некорреткный ввод. Введите значение повторно:	Вводятся посторонние символы (например, буквы)
Введите элемент строки 1, столбца 1 (целое число): 45.6	Некорреткный ввод. Введите значение повторно:	Вводится не целое число

Позитивные тесты

Входные данные	**Результат**	**Условие проверки**
Количество строк – 5 Количество столбцов - 5	Матрица 1: 0 0 56 103 0 0 28 0 0 0 0 0 0 0 0 85 0 32 0 0 0 0 0 0 0 Матрица 2: 0 95 0 0 103 0 0 0 0 53 0 75 0 0 0 0 0 0 0 0 0 0 84 0 0 Матрица результат: 0 95 56 103 103 0 28 0 0 53 0 75 0 0 0 85 0 32 0 0 0 0 84 0 0 Значение ненулевых элементов первой матрицы: 56 103 28 85 32 Соответствующие индексы столбцов ненулевых элементов первой матрицы: 2 3 1 0 2 Индексы элементов, с которых начинаются строки: 0 2 -1 3 -1 Значение ненулевых элементов второй матрицы: 95 103 53 75 84 Соответствующие индексы столбцов ненулевых элементов второй матрицы: 1 4 4 1 2	Автоматическое заполнение матриц (30% заполнения)

	Индексы элементов, с которых начинаются строки: 0 2 3 -1 4 Значение ненулевых элементов в результате: 95 56 103 103 28 53 75 85 32 84 Соответствующие индексы столбцов ненулевых элементов в результате: 1 2 3 4 1 4 1 0 2 2 Индексы элементов, с которых начинаются строки: 0 4 5 7 9	

6. Сравнение двух способов сложения матриц

Время

0% Заполнения

Размеры матрицы	Разреженная матрица	Обычная матрицы
10 x 10	0.0000000200	0.0000000750
100 x 100	0.0000001470	0.0000059220
1000 x 1000	0.0000015000	0.0007125000

5% Заполнения

Размеры матрицы	Разреженная матрица	Обычная матрицы
10 x 10	0.0000000460	0.0000000770
100 x 100	0.0000047500	0.0000059680
1000 x 1000	0.0005100000	0.0006530000

10% Заполнения

Размеры матрицы	Разреженная матрица	Обычная матрицы
10 x 10	0.0000001300	0.0000000800
100 x 100	0.0000095900	0.0000059600
1000 x 1000	0.0016299000	0.0006844000

20% Заполнения

Размеры матрицы	Разреженная матрица	Обычная матрицы
10 x 10	0.0000002280	0.0000000760
100 x 100	0.0000247650	0.0000060840
1000 x 1000	0.0032468000	0.0007353000

30% Заполнения

Размеры матрицы	Разреженная матрица	Обычная матрицы
10 x 10	0.0000002460	0.0000000770
100 x 100	0.0000359410	0.0000059670
1000 x 1000	0.005090000	0.000757000

40% Заполнения

Размеры матрицы	Разреженная матрица	Обычная матрицы
10 x 10	0.000000220	0.000000119
100 x 100	0.000060999	0.000007999
1000 x 1000	0.006634999	0.0007169999

50% Заполнения

Размеры матрицы	Разреженная матрица	Обычная матрицы
10 x 10	0.000000440	0.000000130
100 x 100	0.000066000	0.000009000
1000 x 1000	0.0079849995	0.0007779999

100% Заполнения

Размеры матрицы	Разреженная матрица	Обычная матрицы
10 x 10	0.0000004799	0.0000000900
100 x 100	0.0000430000	0.0000110000
1000 x 1000	0.0044180001	0.0007070000

Занимаемая память

5% Заполнения

Размеры матрицы	Разреженная матрица	Обычная матрицы
10 x 10	292	808
100 x 100	16836	80008
1000 x 1000	1567812	8000008

10% Заполнения

Размеры матрицы	Разреженная матрица	Обычная матрицы
10 x 10	452	808
100 x 100	31108	80008
1000 x 1000	3042756	8000008

20% Заполнения

Размеры матрицы	Разреженная матрица	Обычная матрицы
10 x 10	596	808

100 x 100	57364	80008
1000 x 1000	5764356	8000008

30% Заполнения

Размеры матрицы	Разреженная матрица	Обычная матрицы
10 x 10	772	808
100 x 100	72612	80008
1000 x 1000	7159732	8000008

50% Заполнения

Размеры матрицы	Разреженная матрица	Обычная матрицы
10 x 10	1332	808
100 x 100	121956	80008
1000 x 1000	12014068	8000008

100% Заполнения

Размеры матрицы	Разреженная матрица	Обычная матрицы
10 x 10	1716	808
100 x 100	160836	80008
1000 x 1000	16008036	8000008

Выводы по проделанной работе

Использование специального способа обработки и хранения разреженных матриц проигрывает по времени обычному способу в несколько раз, но выигрывает по памяти примерно в два раза, если процент заполнения <=40. Отсюда можно сделать вывод, что для сложения разреженных матриц данный способ является неэффективным.

Ответы на контрольные вопросы

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

Разреженная матрица – это матрица, которая содержит большое количество нулей.

Способы хранения: линейный связный список, связная схема хранения, двунаправленные стеки и очереди, строчный формат, кольцевой связный список.

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

Для обычной матрицы требуется N*M ячеек памяти (N – кол-во строк M – кол-во столбцов). Для разреженной матрицы количество ячеек памяти зависит от выбора способа ее

хранения. Если рассматривать относительно лабораторной работы, то требуется 2*V (V – количество ненулевых элементов) + T (количество строк в матрице) ячеек памяти.

3.Каков принцип обработки разреженной матрицы?

Данные алгоритм подразумевает под собой обработку только ненулевых элементов матрицы, он заключается в постоянном поиске “пары” для элемента (пара – элемент с совпадающим столбцом и строкой) Вероятность того, что алгоритм по итогу получится линейным очень мала, так как для этого, нужно чтобы ненулевые элементы в двух матрицах стояли на одним и тех же позициях, то есть не требовалось осуществление дополнительного поиска.

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

Если говорить о сложении матриц, то специальный алгоритм оказался крайне неэффективным, однако при небольшом проценте заполнения матриц, можно использовать данный алгоритм, если требуется сокращение расходов по памяти.