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

Отчет по лабораторной работе №3
«ОБРАБОТКА РАЗРЕЖЕННЫХ МАТРИЦ»

Студент Нисуев Нису Феликсович

Группа ИУ7 – 32Б

Преподаватель Барышникова Марина Юрьевна

2023 г.

Оглавление

Описание условия задачи

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

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;
- вектор 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 хранящегося в форме вектора A и вектора, содержащего номера столбцов этих элементов, и матрицы, хранящейся в указанной форме, с получением результата в форме хранения вектора-строки.
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.

Описание технического задания

Входные данные:

1. **Номер команды**: целое число в диапазоне {-1} ∪ [1; 5].
2. **Дополнения к таблице:** строковое или целочисленное поле (в зависимости от команды)

Выходные данные:

1. Результаты произведений в разных формах
2. Введенные или сгенерированные вектор-строка и/или матрица в разных формах

Обращение к программе:

Запускается через терминал командой: ./build/app.exe

Аварийные ситуации:

1. Ввод некорректного пункта меню
2. Некорректные значения размера матрицы
3. Некорректное значение процента
4. Некорректные данные при вводе матрицы

НАБОР ТЕСТОВ

№	Название теста	Пользовательский ввод	Вывод
Негативные тесты			
1	Некорректные размеры	1 S S -1 1	ERROR: Incorrect input
2	Некорректные данные	1 S S 2 2 2 ф	ERROR: Incorrect input
3	Некорректная команда	1 S п	ERROR: Incorrect choice
4	Некорректный процент	2 2 2 -1	ERROR: Incorrect input
5	Некорректный процент	2 2 2 101	ERROR: Incorrect input
6	Некорректный пункт меню	10	ERROR: Incorrect action
Позитивные тесты			
1	Ввод матрицы и вектора-строки	1 {Корректный ввод}	Data successfully loaded
2	Генерация матрицы и вектора-строки	2	Data successfully generated

		2 2 75%	
3	Вывод стандартных матрицы и вектора-строки	6 A C C	{Классические матрица и вектор-строка}
4	Вывод разреженных матрицы и вектора-строки	6 A S S	{Разреженные матрица и вектор-строка}
5	Вывод замера	5	{Таблица замеров}
6	Стандартное умножение с выводом матрицы в стандартном виде	3 C	{Результирующий вектор-строка в классическом виде}
7	Разреженное умножение с выводом матрицы в разреженном виде	4 S	{Результирующий вектор-строка в разреженном виде}

Описание структуры данных

```c
/// @typedef структура матрицы
typedef struct {
    size_t rows; // Кол-во строк
    size_t cols; // Кол-во столбцов
    int **data;  // Элементы матрицы
} matrix_t;

/// @typedef структура вектора-строки
typedef struct {
    size_t size;   // Кол-во столбцов
    int *vector;  // Вектор-строка
} rvector_t;

/// @typedef структура разреженной матрицы
typedef struct {
    size_t rows;      // Кол-во строк
    size_t cols;      // Кол-во столбцов
    size_t els_cnt;   // Кол-во ненулевых элементов
    int *els;         // Элементы матрицы
    size_t *el_i;     // Номера строк элементов
    ssize_t  *el_j;   // Индекс первого ненулевого элемента в столбце
} msparce_t;

/// @typedef структура разреженного вектора-строки
typedef struct {
    size_t size;    // Кол-во столбцов
    size_t els_cnt; // Кол-во ненулевых элементов
    int *els;       // Элементы вектора-строки
    size_t *el_j;   // Номера столбцов элементов
} rvsparce_t;
```

Введенные данные:

```
Output [V:Vector, M:Matrix, A:All]: A

Matrix output
---

Print matrix
[C:Classical, S:Sparce]: C

Matrix sizes: 3 3
Matrix:
1    0    0
0    1    0
0    0    1


Vector output
---

Print vector
[C:Classical, S:Sparce]: C

Vector size: 3
Vector: 1    2    3
```

Результаты умножений:

разреженное

```
Multiply vector-column result
---
Print vector
[C:Classical, S:Sparce]: C

Vector size: 3
Vector: 1    2    3
```

стандартное

```
Multiply vector-column result
---
Print vector
[C:Classical, S:Sparce]: C

Vector size: 3
Vector: 1    2    3
```

Замерный эксперимент

Проводилось 1000 замеров для указанных размеров и процентов разреженности матриц. Измерялась память всех структур

Size	Sparceness, %	Measures	Std matrix	Sparce matrix
10x10	0%	Mem, b	592	1576
		Time, mcs	0	0
	25%	Mem, b	592	1252
		Time, mcs	0	0
	50%	Mem, b	592	916
		Time, mcs	0	0
	75%	Mem, b	592	532
		Time, mcs	0	0
	100%	Mem, b	592	136
		Time, mcs	0	0
50x50	0%	Mem, b	10832	31656
		Time, mcs	5	7
	25%	Mem, b	10832	24012
		Time, mcs	5	6
	50%	Mem, b	10832	16356
		Time, mcs	5	6
	75%	Mem, b	10832	8688
		Time, mcs	5	3
	100%	Mem, b	10832	456
		Time, mcs	5	0
100x100	0%	Mem, b	41632	123256
		Time, mcs	21	24
	25%	Mem, b	41632	92956
		Time, mcs	21	53
	50%	Mem, b	41632	62656
		Time, mcs	21	43
	75%	Mem, b	41632	32356
		Time, mcs	21	17
	100%	Mem, b	41632	856
		Time, mcs	24	0
500x500	0%	Mem, b	1008032	3016056
		Time, mcs	458	542
	25%	Mem, b	1008032	2264556
		Time, mcs	456	1821
	50%	Mem, b	1008032	1513056
		Time, mcs	451	1557
	75%	Mem, b	1008032	761556

		Time, mcs	441	797
	100%	Mem, b	1008032	4056
		Time, mcs	445	0
1000x1000	0%	Mem, b	4016032	12032056
		Time, mcs	2849	2207
	25%	Mem, b	4016032	9029056
		Time, mcs	2768	7697
	50%	Mem, b	4016032	6026056
		Time, mcs	1976	9921
	75%	Mem, b	4016032	3023056
		Time, mcs	4922	6158
	100%	Mem, b	4016032	8056
		Time, mcs	2828	0

Вывод: разреженные матрицы начинают занимать меньше памяти при разреженности более 50%. Также алгоритм произведения разреженных матриц начинает работать быстрее чем стандартный при разреженности менее 75%

Описание алгоритма

1. Ввод команды (возможные команды представлены в меню)
2. Пока пользователь не введет -1 (выход из программы), ему будет предложено вводить номера команд и выполнять действия по выбору

Ответы на контрольные вопросы

1. Что такое разреженная матрица, какие способы хранения вы знаете?

Разреженная матрица — матрица, содержащая большое кол-во нулевых элементов. Хранить такую можно как обычную матрицу, с помощью линейных связных списков, кольцевых связных списков, двунаправленных стеков и очередей.

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

Под обычную матрицу выделяется **N * M** ячеек памяти. Память под разреженную матрицу выделяется в зависимости от схемы хранения. Кроме того, память зависит от количества ненулевых элементов.

3. Каков принцип обработки разреженной матрицы?

Т.к. разреженные матрицы содержат большое кол-во нулей, то и хранятся они в таких структурах, которые «запоминают» только ненулевые элементы. Поэтому алгоритмы обработки оперируют лишь значащими данными, что даёт выигрыш по памяти и скорости по сравнению с алгоритмами обработки обычных матриц.

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

Разреженность матрицы следует учитывать только в том случае, если из этого можно извлечь выгоду за счёт игнорирования нулевых элементов. При достижении определенного процента наполнения ненулевыми элементами происходит значительное падение эффективности по времени.

Вывод

Использовать специальные структуры данных для (разреженных) матриц имеет смысл лишь при большом кол-ве элементов, т.к. тогда выигрыш по памяти будет существенен, и лишь при заполненности до 25% – иначе стандартные алгоритмы обработки матриц будут эффективнее во всех случаях, начиная с размерности 50x50, нежели те, которые реализуются для обработки разреженных матриц. Тем более, чем большая размерность у матриц, тем меньше процент заполненности ненулевыми элементами необходим для того, чтобы стандартный способ сложения матриц превосходил по скорости способ обработки разреженных.