Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский государственный технический университет имени Н.Э. Баумана

ФАКУЛЬТЕТ Информатика и системы управления
КАФЕДРА Программное обеспечение ЭВМ и информационные технологии

Отчет по дисциплине «Типы и структуры данных»
Лабораторная работа №3

Вариант 4

Студент Голикова С.М.
Группа ИУ7-35Б

Москва, 2021

1. Условие задачи

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;
- связный список JA, в элементе Nk которого находится номер компонент в A и IA, с которых начинается описание столбца Nk матрицы A.

1. Смоделировать операцию умножения матрицы и вектора-столбца, хранящихся в этой форме, с получением результата в той же форме.

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.

2. Внешняя спецификация

а) Исходные данные и результаты

Исходные данные:

- Номер команды из меню (целое число от 0 до 9)
- Командно-зависимые данные:
    - размеры матрицы
    - размер вектора-столбца
    - количество ненулевых элементов
    - значения элементов
    - индексы ненулевых элементов
    - процент заполненности матрицы

Допущения:

1) Матрица и вектор-столбец содержат целочисленные элементы

2) Максимальное значение элемента в матрице / векторе-столбце 99999
3) Максимальный размер для вывода матрицы в стандартном виде 50x50

Выходные данные:

В зависимости от пункта меню:

- исходная матрица в стандартном виде / в форме 3-х объектов
- исходный вектор-столбец в стандартном виде / в форме 3-х объектов
- результат умножения в стандартном виде / в форме 3-х объектов
- результаты сравнения времени и памяти выполнения алгоритмов

b) Задачи, реализуемые программой

Умножение матрицы на вектор-столбец, сравнение разных алгоритмов обработки матриц.

c) Способ обращения к программе

Программа может быть вызвана через консоль (./app.exe) или запущена через любую среду разработки C.

Пользователь взаимодействует с программой через меню, выбирая нужный ему пункт и вводя необходимые данные.

d) Возможные аварийные ситуации и ошибки пользователя

Аварийные ситуации и ошибки:

- некорректные размеры матрицы или вектора-столбца (не целые или не положительные числа)
- некорректный процент заполненности (в т.ч. число, меньшее 0 или большее 100)
- некорректный элемент матрицы (в т.ч. слишком большой по модулю)
- некорректные индексы вводимого элемента
- несоответствие числа столбцов матрицы числу строк вектора-столбца при умножении
- отсутствие матрицы или вектора-столбца при умножении

- некорректный выбор пункта меню

В случае аварийной ситуации пользователю выдается сообщение об ошибке и происходит возвращение в меню.

3. Внутренние структуры данных

```
// стандартный вид матрицы
typedef struct
{
    int rows;     // количество строк
    int columns;  // количество столбцов
    int **data;   // матрица
} matrix_std_t;

// элемент связанного списка
typedef struct node
{
    int index;          // значение элемента
    struct node *next;  // указатель на следующий элемент
} node_t;

// матрица в форме 3-х объектов
typedef struct
{
    int rows;       // количество строк
    int columns;    // количество столбцов
    int nn_cnt;     // количество ненулевых элементов
    int *A;         // значения ненулевых элементов
    int *IA;        // номера строк ненулевых элементов
    node_t JA;      // первый элемент связанного списка,
                    // хранящего номера компонент в A и IA,
                    // с которых начинается описание столбца
} matrix_sparse_t;
```

4. Описание алгоритма

Алгоритм умножения матрицы на вектор-столбец (матрица и вектор-столбец хранятся в виде 3-х объектов):

Проходим по массиву ненулевых элементов матрицы (A), умножая каждый элемент этого массива на элемент вектора-столбца так, чтобы номер столбца текущего элемента матрицы (номер столбца мы получаем из связанного списка JA) был равен номеру элемента в векторе-столбце (IA). Полученное произведение мы прибавляем к соответствующему значению элемента результирующего вектора-столбца.

5. Основные функции, используемые в программе

Имя	`int choose_action(void);`
Функция	Выбор действия в меню
Имя	`int input_std_matr(matrix_std_t *matr, int is_vector);`
Функция	Ввод матрицы в стандартном виде с клавиатуры
Имя	`int generate_matrix(matrix_std_t *std, matrix_sparse_t *sprs, double percent, int n, int m);`
Функция	Создание матрицы из случайных элементов с определенным размером и процентом заполненности
Имя	`int sparse_to_std(matrix_sparse_t *from, matrix_std_t *to);` `int std_to_sparse(matrix_std_t *from, matrix_sparse_t *to);`
Функция	Перевод матрицы из разреженного формата в стандартный и обратно

Имя	`void print_std_matrix(matrix_std_t *matr, int vect);` `void print_sparse_matrix(matrix_sparse_t *matr, int vect);`
Функция	Вывод матрицы в стандартном и разреженном форматах
Имя	`int multi_std(matrix_std_t *matr, matrix_std_t *vector, matrix_std_t *res, double *time);` `int multi_sparse(matrix_sparse_t *matr, matrix_sparse_t *vector, matrix_sparse_t *res, double *time);`
Функция	Умножение матриц в стандартном и разреженном форматах
Имя	`int full_analysis(matrix_std_t *m_std, matrix_std_t *v_std, matrix_sparse_t *m_sprs, matrix_sparse_t *v_sprs);`
Функция	Сравнение времени и выделяемой памяти при выполнении различных алгоритмов

5. Тесты

Позитивные тесты

№ теста	Ввод	Действие	Вывод
1	Меню: 1 10 10 50	Создание матрицы 10x10 с 50 (50%) случайными ненулевыми элементами	Матрица успешно добавлена
2	Меню: 2 10 100	Создание вектора-столбца размером 10x1 с 10 (100%) случайными ненулевыми элементами	Вектор-столбец успешно добавлен
3	Меню: 3	Добавление матрицы,	Матрица успешно

	Вид матрицы: 1 2 2 1 0 0 2	введенной пользователем в стандартном виде	добавлена
4	Меню: 3 Вид матрицы: 2 2 2 1 9 0 0	Создание матрицы по введенным пользователем ненулевым элементам (Стандартный вид: 9 0 0 0)	Матрица успешно добавлена
5	Меню: 5 Вид матрицы: 1	Вывод матрицы в стандартном виде	Матрица в стандартном виде
6	Меню: 5 Вид матрицы: 2	Вывод матрицы в виде 3-х объектов	Матрица в виде 3-х объектов
7	Меню: 7 Исходная матрица: 0 0 2 4 0 1 5 0 3 Исходный вектор-столбец: 1 2 3	Умножение в стандартном виде	6 7 14
8	Меню: 8 Исходная матрица: A: 4 5 2 1 3 IA: 1 2 0 1 2 JA: 0 2 2 5 Исходный вектор-столбец: A: 1 2 3 IA: 0 1 2 JA: 0 3	Умножение в виде 3-х объектов	A: 6 7 14 IA: 0 1 2 JA: 0 3

10	Меню: 9 Исходная матрица: 0 0 2 4 0 1 5 0 3 Исходный вектор-столбец: 1 2 3	Сравнение времени выполнения алгоритмов и выделяемой памяти	Размерность матрицы: 3x3 Процент заполненности матрицы: 56% Процент заполненности вектора-столбца: 100% <Время, память для каждого алгоритма>
11	Меню: 7 Исходная матрица: 0 0 2 0 0 1 5 0 0 Исходный вектор-столбец: 0 2 0	Умножение в стандартном виде (результат - нулевой вектор-столбец)	0 0 0
12	Меню: 8 Исходная матрица: A: 5 2 1 IA: 2 0 1 JA: 0 1 1 3 Исходный вектор-столбец: A: 2 IA: 1 JA: 0 1	Умножение в виде 3-х объектов (результат - нулевой вектор-столбец)	A: IA: JA: 0 0

Негативные тесты

№ теста	Ввод	Действие	Вывод
1	Меню: 10	Выбор несуществующего пункта меню	Ошибка ввода: необходимо ввести номер одного из предложенных вариантов
2	Меню: 1 ten	Некорректный ввод размеров матрицы	Ошибка при вводе размеров
3	Меню: 3 Вид матрицы: 2 2 2 1 element 1 1	Некорректный ввод элемента матрицы	Ошибка при вводе элемента
4	Меню: 3 Вид матрицы: 2 2 2 5	Количество ненулевых элементов больше общего количества элементов матрицы	Ошибка при вводе количества ненулевых элементов
5	Меню: 3 Вид матрицы: 2 2 2 1 1 2 0	Индекс ненулевого элемента выходит за границы матрицы	Ошибка при вводе индекса ненулевого элемента
6	Меню: 3 Вид матрицы: 2 2 2 2 1 0 0 2 0 0	Попытка записать на одно место в матрице два элемента	Элемент с введенным индексом уже существует
7	Меню: 5	Попытка вывода	Матрица не существует.

	(матрица не создана)	несуществующей матрицы	Введите ее вручную или сгенерируйте, выбрав соответствующий пункт меню
8	Меню: 6 (вектор не создан)	Попытка вывода несуществующего вектора	Вектор-столбец не существует. Введите его вручную или сгенерируйте, выбрав соответствующий пункт меню
9	Меню: 7 (или 8) Размер матрицы: 2x2 Размер вектора-столбца: 3	Количество столбцов матрицы не равно количеству строк вектора-столбца (необходимое условие для умножения матриц)	Несоответствие количества столбцов матрицы количеству элементов вектора-столбца
10	Меню: 1 10 10 110	Процент заполненности больше 100	Ошибка при вводе процента заполненности

6. Анализ эффективности работы алгоритмов

Изменила измерение времени: теперь учитывается время выделения памяти. На 50% заполненности стандартный алгоритм все равно немного проигрывает (хотя и гораздо меньше, чем до этого). На 75% и 100% стандартный алгоритм стал выигрывать по времени намного больше, чем раньше.

Размер матрицы	Процент заполненности	Стандартный алгоритм		Алгоритм обработки разреженных матриц	
		Время, с	Память, байт	Время, с	Память, байт
10x10	5	0.000006	440	0.000004	328
	10	0.000007	440	0.000004	368
	50	0.000008	440	0.000009	688
	75	0.000008	440	0.000009	888
	100	0.000009	440	0.000011	1088
50x50	5	0.000052	10200	0.000014	2248
	10	0.000053	10200	0.000017	3248
	50	0.000054	10200	0.000048	11248
	75	0.000049	10200	0.000058	16248
	100	0.000053	10200	0.000074	21248
100x100	5	0.000114	40400	0.000030	6448
	10	0.000187	40400	0.000039	10448
	50	0.000178	40400	0.000140	42448
	75	0.000187	40400	0.000256	62448
	100	0.000177	40400	0.000263	82448

500x500	5	0.004341	1002000	0.000373	112048
	10	0.004411	1002000	0.000667	212048
	50	0.004387	1002000	0.004123	1012048
	75	0.004353	1002000	0.004705	1512048
	100	0.004327	1002000	0.006306	2012048

7. Выводы

Использование алгоритмов хранения и обработки разреженных матриц выгодно при небольшом количестве ненулевых элементов, чуть меньше, чем до 50% заполненности.

В таком случае, алгоритм выигрывает как в размерах занимаемой памяти, так и в скорости обработки. При заполненности 50% и более алгоритм обработки и хранения разреженных матриц начинает проигрывать в памяти и времени (особенно заметно это на матрицах больших размеров).

8. Контрольные вопросы

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

Разреженная матрица — это матрица, содержащая большое количество нулей. Методы хранения: линейный связный список, кольцевой связный список, двунаправленные стеки и очереди, диагональная схема хранения симметричных матриц.

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

Под обычную матрицу (N – количество строк, M – количество столбцов) выделяет N*M ячеек памяти.
Для разреженной матрицы количество ячеек памяти зависит от способа. В случае, когда сохраняются ненулевые элементы и их индексы, требуется

количество ячеек в размере К * 3 (К — количество ненулевых элементов). В случае, когда сохраняются ненулевые элементы, номера их строк и указатели на элементы, с которых начинается описание каждого столбца, требуется количество ячеек в размере K * 2 + M.

3. **Каков принцип обработки разреженной матрицы?**

Алгоритмы обработки разреженных матриц предусматривают действия только с ненулевыми элементами. Количество операций в этом случае будет пропорционально количеству ненулевых элементов.

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

Стандартные алгоритмы обработки матриц выгоднее применять при большом количестве ненулевых элементов, т.к. на большом количестве ненулевых элементов память, выделяемая под хранение матрицы в виде 3-х объектов, больше памяти, выделяемой под хранение матрицы в стандартном виде.