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 ПО ДИСЦИПЛИНЕ:
ТИПЫ И СТРУКТУРЫ ДАННЫХ

ОБРАБОТКА РАЗРЕЖЕННЫХ МАТРИЦ

Студент Жаринов М. А.

Вариант 2

Группа **ИУ7-32Б**

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

Студент _____ **Жаринов М. А.**

Преподаватель _____ **Барышникова М. Ю.**

2024

Описание условия задачи

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

- 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 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.

Техническое задание

Исходные данные:

Номер пункта меню – Целое число от 1 до 12.

Текстовый файл с матрицами: Файл, содержащий 2 целочисленные матрицы в стандартном виде и из размерности, разделенные пробельными символами.

Матрицы в стандартном виде: 2 матрицы, формат совпадает с файлом.

Матрицы в координатном виде: 2 матрицы, сначала задается размерности матриц, потом количество элементов в матрицах, и нужное количество троек строка-столбец-значение.

Выходные данные:

Сумма матриц в стандартном или разреженном виде, результаты сравнения алгоритмов.

Реализуемые задачи:

Пункт меню 1 – Выход из программы

Пункт меню 2 – Чтение матриц из файла

Пункт меню 3 – Ввод матриц в стандартном виде

Пункт меню 4 – Ввод матриц координатным способом

Пункт меню 5 – Заполнение матриц заданным количеством ненулевых элементов

Пункт меню 6 – Печать матриц в стандартном виде

Пункт меню 7 – Печать матриц в разреженном виде

Пункт меню 8 – Найти сумму стандартных матриц и вывести в стандартном виде

Пункт меню 9 – Найти сумму стандартных матриц и вывести в разреженном виде

Пункт меню 10 – Найти сумму разреженных матриц и вывести в стандартном виде

Пункт меню 11 – Найти сумму разреженных матриц и вывести в разреженном виде

Пункт меню 12 – Проведение сравнительного анализа различных способов умножения матриц

Способ обращения к программе:

Строка запуска программы ./app.exe

После запуска с помощью меню нужно заполнить матрицы или запустить анализ методов умножения.

Аварийные ситуации:

1. EOF в вводе. Код ошибки 1

Все остальные ошибочные ситуации не являются аварийными и вызывают повторное приглашение к вводу или возвращение в меню.

Внутренние структуры данных

Матрица в стандартном виде хранится в структуре standart_matrix_t.

```
typedef struct
{
int matrix[MATRIX_ROWS][MATRIX_COLS];
int rows;
int cols;
} standard_matrix_t;
```

Поле `matrix` – Двумерный массив значений матрицы

Поле `rows` – количество строк матрицы

Поле `cols` – количество столбцов матрицы

Матрица в разреженном виде хранится в структуре standart_matrix_t.

```
typedef struct
{
int a[MATRIX_ROWS * MATRIX_COLS];
int ia[MATRIX_ROWS * MATRIX_COLS];
int ja[MATRIX_COLS];
int rows;
int cols;
int elements_num;
} sparse_matrix_t;
```

Поле `a` –массив ненулевых значений матрицы

Поле i`a` –массив индексов строк соответствующего элемента массива a

Поле j`a` –массив индексов элементов в массиве a, с которых начинается описание i-ого столбца или -1 если такой отсутствует.

Поле `rows` – количество строк матрицы

Поле `cols` – количество столбцов матрицы

Поле `elements_num` –количество ненулевых значений матрицы

Для удобства работы указатели на одну и ту же матрицу в разном виде хранятся в структуре both_matrix_t.

```
typedef struct
{
standard_matrix_t *standard_matrix;
sparse_matrix_t *sparse_matrix;
```

```
    } both_matrix_t;
```

Поле `standard_matrix_t` – указатель на матрицу в стандартном представлении

Поле `sparse_matrix_t` – указатель на матрицу в разреженном представлении

Описание алгоритма

1. Программа запрашивает номер элемента в меню, считывает его и выполняет соответствующее действие:

Пункт меню 1 – Выход из программ

Пункт меню 2 – Чтение матриц из файла

1. Программа запрашивает имя файла и считывает его.
2. Программа производит чтение указанного файла в матрицы в стандартном виде.
3. Программа приводит в соответствие матрицы в разреженном виде.

Пункт меню 3 – Ввод матриц в стандартном виде:

1. Программа запрашивает размерности матриц.
2. Программа запрашивает значения элементов матрицы.
3. Программа приводит в соответствие матрицы в разреженном виде.

Пункт меню 4 – Ввод матриц координатным способом

1. Программа запрашивает размерности матриц.
2. Программа запрашивает количество элементов к вводу.
3. Программа запрашивает значения элементов матрицы в виде тройки строка – столбец-значение.

Пункт меню 5 – Заполнение матриц заданным количеством ненулевых элементов

1. Программа запрашивает размерности матриц.
2. Программа запрашивает количество ненулевых элементов.
3. Программа заполняет матрицы заданным количеством ненулевых элементов и перемешивает их.

Пункт меню 8 – Найти сумму стандартных матриц и вывести в стандартном виде

1. Программа выполняет поэлементное сложение матриц.
2. Программа печатает получившуюся матрицу.

Пункт меню 9 – Найти сумму стандартных матриц и вывести в разреженном виде

1. Программа выполняет поэлементное сложение матриц.
2. Программа переводит получившуюся матрицу в разреженный вид.
3. Программа печатает получившуюся матрицу в разреженном виде.

Пункт меню 10 – Найти сумму разреженных матриц и вывести в стандартном виде

1. Программа обходит матрицу по столбцам, и если очередные элементы в двух матрицах находятся в одинаковых строчках, то складывает их, иначе записывает в выходную матрицу только элемент с меньшим индексом строки и переходит на следующий элемент в столбце.
2. Программа переводит получившуюся матрицу в стандартный вид.
3. Программа печатает получившуюся матрицу в стандартном виде.

Пункт меню 11 – Найти сумму разреженных матриц и вывести в разреженном виде

1. Программа обходит матрицу по столбцам, и если очередные элементы в двух матрицах находятся в одинаковых строчках, то складывает их, иначе записывает в выходную матрицу только элемент с меньшим индексом строки и переходит на следующий элемент в столбце.
2. Программа печатает получившуюся матрицу в разреженном виде.

Пункт меню 12 – Проведение сравнительного анализа различных способов умножения матриц

1. Программа в цикле задает нужные размерности матрицы
2. Программа заполняет матрицу случайным образом с заполненностью от 0 до 100 процентов.
3. Программа находит среднее время сложение матриц двумя алгоритмами, выигрыш при использовании разреженного алгоритма и печатает результаты.

Основные функции:

Функция для обработки выбранного пункта меню. Принимает 2 входные матрицы, выходную матрицу и номер выбранного пункта меню. Возвращает флаг необходимости завершения программы

```
bool process_menu(menu_item_t menu_item, both_matrix_t in_matrix_1, both_matrix_t in_matrix_2, both_matrix_t out_matrix);
```

Функция чтения матриц из файла. Принимает 2 входные матрицы

```
void read_file(both_matrix_t matrix_1, both_matrix_t matrix_2);
```

Функция ввода матриц пользователем в стандартном виде. Принимает 2 входные матрицы

```
void read_user_standard(both_matrix_t matrix_1, both_matrix_t matrix_2);
```

Функция ввода матриц пользователем в координатном виде. Принимает 2 входные матрицы

```
void read_user_coord(both_matrix_t matrix_1, both_matrix_t matrix_2);
```

Функция заполнения матриц пользователем с заданным количеством ненулевых элементов. Принимает 2 входные матрицы

```
void fill_with_sparsity(both_matrix_t matrix_1, both_matrix_t matrix_2);
```

Функция сложения матриц в стандартном виде. Принимает 2 указателя на входные матрицы в стандартном виде и указатель на выходную матрицу

```
void sum_standard(const standard_matrix_t *in_matrix_1, const standard_matrix_t *in_matrix_2, standard_matrix_t *out_matrix);
```

Функция сложения матриц в разреженном виде. Принимает 2 указателя на входные матрицы в разреженном виде и указатель на выходную матрицу

```
void sum_sparse(sparse_matrix_t *in_matrix_1, sparse_matrix_t *in_matrix_2, sparse_matrix_t *out_matrix);
```

Функция перевода матрицы из стандартного вида в разреженный. Принимает матрицу

```
void sparse_to_standard(both_matrix_t matrix);
```

Функция перевода матрицы из разреженного вида в стандартный.

Принимает матрицу

```
void standard_to_sparse(both_matrix_t matrix);
```

Функция анализа разных способов сложения матриц

```
void run_analysis(void);
```

Набор тестов

Обработка пунктов меню

Описание теста	Вход	Результат
Неположительное число	0	Для выбора пункта меню введите целое число от 1 до 12
Число больше 12	13	Для выбора пункта меню введите целое число от 1 до 12
Максимальный пункт меню	12	[сведения о скорости сложения]
Минимальный пункт меню	1	[выход из программы]
Символ вместо числа	a	Для выбора пункта меню введите целое число от 1 до 12
Выбор пунктов меню, которые требуют данные в матрице при пустой матрице	6	Нельзя производить данное действие над пустыми матрицами! Прочтите матрицы из файла или заполните вручную!

Чтение файла

Описание теста	Ввод	Содержимое файла	Результат
Не существующий файл	2 not_exist.txt	-	Введите корректное имя файла!
Слишком длинное имя файла	2 hsdgfiwegbiesgbchsbchsbchsbchsdbcihdsbchsdbcdshcbsdicbskcbsdcb.txt	-	Ошибка ввода! Введите имя файла не длиннее 30 символов!

Пустой файл	2 empty.txt	-	В файле должно быть хотя бы 2 числа - кол-во строк и столбцов!
Файл с матрицами 1 на 1	2 t.txt	1 1 1 0	Файл прочитан успешно!
Файл с посторонним символом	2 t.txt	2 2 1 2 0 2 3 3 5 a 5	В файле не только целые числа или слишком мало чисел!
Файл с недостаточным количеством значений	2 t.txt	2 2 1 2 0 2 3 3 5	В файле не только целые числа или слишком мало чисел!
Файл с слишком большим количеством значений	2 t.txt	1 2 2 3 3 4 5	В файле больше данных чем требуется!

Ввод матриц

Описание теста	Ввод	Вывод
Неположительная размерность	3 -3 3	Количество строк должно быть от 1 до 1000, а столбцов - от 1 до 1000!
Отрицательное кол-	4/5	Введено количество не от 0 до

во элементов к вводу	2 2 -1	4!
Слишком большое кол-во элементов к вводу	4/5 2 2 13	Введено количество не от 0 до 4!
Слишком большая размерность	3/4/5 10000 1000	Количество строк должно быть от 1 до 1000, а столбцов - от 1 до 1000!
Отрицательный индекс при вводе координатным методом	4 2 2 2 -1 1 3	Индексы не могут отрицательными или быть больше соответствующих размерностей!

Сложение матриц (разреженным методом)

Описание теста	Матрица 1	Матрица 2	Вывод
Матрицы из 0	0 0 0 0 0 0	0 0 0 0 0 0	0 0 0 0 0 0
Матрицы без 0	1 2 2 1 1 1	2 1 1 2 2 2	3 3 3 3 3 3
Матрицы с нулевыми столбцами	3 0 0 1 0 0 3 0 0	3 0 0 3 0 4 2 0 4	6 0 0 4 0 4 5 0 4

Результаты сравнения

Для анализа я измерил средние скорости сложения при размерах 10 на 10, 100 на 100,10 на 1000, 1000 на 10 и 1000 на 1000 элементов и получил следующие результаты (в наносекундах), таблицы сокращены:

Размер 10 на 10

Процент заполнения	Время сложения стандартных матриц	Время сложения разреженных матриц	Выигрыш времени при сложении разреженных матриц
0	240	82	65.83%
1	237	104	56.12%
2	238	108	54.62%
3	239	106	55.65%
4	239	121	49.37%
5	239	130	45.61%
16	239	208	12.97%
17	240	213	11.25%
18	239	214	10.46%
19	239	232	2.93%
20	239	233	2.51%
21	239	233	2.51%
22	240	248	-3.33%
23	240	257	-7.08%
24	240	261	-8.75%
25	240	270	-12.50%
26	240	278	-15.83%
99	235	576	-145.11%
100	234	580	-147.86%

Размер 100 на 100

Процент заполнения	Время сложения стандартных матриц	Время сложения разреженных матриц	Выигрыш времени при сложении разреженных матриц
0	21960	599	97.27%
1	21764	1552	92.87%
2	22018	2322	89.45%
3	21795	3102	85.77%
4	21799	3938	81.93%
5	21909	4665	78.71%
6	21217	5479	74.18%
7	21217	6394	69.86%
8	22102	7166	67.58%
9	21346	8029	62.39%
10	21340	9167	57.04%
11	21347	10137	52.51%
12	21343	10940	48.74%
13	21349	11909	44.22%
14	21352	13532	36.62%
15	21371	14188	33.61%
16	21381	15816	26.03%
17	21389	16131	24.58%
18	21936	18059	17.67%
19	22590	19059	15.63%
20	21357	19868	6.97%
21	22761	23840	-4.74%
22	22407	23123	-3.20%
23	23694	25650	-8.26%
24	21497	26300	-22.34%
25	21372	28476	-33.24%
26	21665	30078	-38.83%
27	21242	32748	-54.17%
28	21349	32805	-53.66%
29	21532	33960	-57.72%
30	21230	36575	-72.28%
59	21347	71227	-233.66%
60	21947	71405	-225.35%
61	21329	73545	-244.81%
62	21638	72125	-233.33%
63	21343	70688	-231.20%
64	21327	69809	-227.33%
65	21463	71630	-233.74%
66	21332	70516	-230.56%
98	22283	51393	-130.64%
99	21363	52619	-146.31%
100	22575	49478	-119.17%

Размер 10 на 1000

Процент заполнения	Время сложения стандартных матриц	Время сложения разреженных матриц	Выигрыш времени при сложении разреженных матриц
0	20461	14105	31.06%
1	20466	13845	32.35%
2	20461	13890	32.11%
3	20468	13872	32.23%
4	20460	13881	32.16%
5	20570	13911	32.37%
6	20462	14173	30.74%
7	21036	13887	33.98%
8	20709	13932	32.72%
9	20464	13960	31.78%
10	20587	13946	32.26%
11	20466	13909	32.04%
12	20460	13919	31.97%
13	20452	13822	32.42%
14	20461	13966	31.74%
15	20456	13962	31.75%
16	20460	13954	31.80%
17	29124	14235	51.12%
18	20456	15765	22.93%
19	28327	15615	44.88%
20	20863	13996	32.91%
21	20457	14027	31.43%
22	20457	14016	31.49%
80	20461	14284	30.19%
81	20952	14340	31.56%
82	21702	14511	33.14%
83	21061	15130	28.16%
84	20337	14247	29.95%
85	20459	14376	29.73%
86	20339	14216	30.10%
87	20333	14218	30.07%
88	20497	14356	29.96%
89	20333	14296	29.69%
90	20460	14209	30.55%
91	20463	14349	29.88%
92	20474	14265	30.33%
93	20330	14294	29.69%
94	20340	14249	29.95%
95	20343	14238	30.01%
96	20342	14281	29.80%
97	20335	14288	29.74%
98	20338	14273	29.82%
99	20340	14271	29.84%
100	20945	15191	27.47%

Размер 1000 на 10

Процент заполнения	Время сложения стандартных матриц	Время сложения разреженных матриц	Выигрыш времени при сложении разреженных матриц
0	22131	46959	-112.19%
1	22440	47033	-109.59%
2	22302	49517	-122.03%
3	21893	46864	-114.06%
4	21969	46900	-113.48%
5	21930	47050	-114.55%
6	22128	46792	-111.46%
7	21880	47034	-114.96%
8	21888	46911	-114.32%
9	21883	46796	-113.85%
10	21888	46867	-114.12%
11	22022	47111	-113.93%
12	21890	47083	-115.09%
13	21884	48322	-120.81%
14	22023	47769	-116.91%
15	22013	47575	-116.12%
16	23443	47768	-103.76%
17	22355	47236	-111.30%
18	21878	47069	-115.14%
19	21886	47515	-117.10%
20	21883	47245	-115.90%
79	22051	47689	-116.27%
80	21895	47565	-117.24%
81	21894	47191	-115.54%
82	22150	47160	-112.91%
83	22042	47707	-116.44%
84	21907	47341	-116.10%
85	21870	47433	-116.89%
86	21902	47205	-115.53%
87	22037	47599	-116.00%
88	22828	47661	-108.78%
89	22134	48404	-118.69%
90	23545	48618	-106.49%
91	22040	47756	-116.68%
92	21895	47181	-115.49%
93	21886	47183	-115.59%
94	22041	47187	-114.09%
95	21888	47325	-116.21%
96	21880	47664	-117.84%
97	21889	47184	-115.56%
98	21887	47220	-115.74%
99	22084	47190	-113.68%
100	21899	47273	-115.87%

Размер 1000 на 1000

Процент заполнения	Время сложения стандартных матриц	Время сложения разреженных матриц	Выигрыш времени при сложении разреженных матриц
0	2193402	5754	99.74%
1	2066872	183636	91.12%
2	2160072	356494	83.50%
3	2065852	529982	74.35%
4	2073967	704790	66.02%
5	2215494	882290	60.18%
6	2081722	1056362	49.26%
7	2077036	1232275	40.67%
8	2080422	1401700	32.62%
9	2079821	1575211	24.26%
10	2086201	1754082	15.92%
11	2190029	1921840	12.25%
12	2082026	2102838	-1.00%
13	2070061	2278822	-10.08%
14	2154045	2445423	-13.53%
15	2075546	2625040	-26.47%
16	2220496	2799747	-26.09%
17	2179611	2967052	-36.13%
18	2075363	3159942	-52.26%
19	2230477	3300243	-47.96%
20	2239631	3469411	-54.91%
21	2084318	3655434	-75.38%
22	2209680	3791231	-71.57%
23	2079116	3963891	-90.65%
24	2084603	4121885	-97.73%
68	2071761	7897164	-281.18%
69	2195626	7849594	-257.51%
70	2099729	7820877	-272.47%
71	2069227	7786428	-276.30%
72	2212587	7743772	-249.99%
73	2070991	7715086	-272.53%
74	2070274	7670488	-270.51%
75	2168670	7615419	-251.16%
76	2073716	7541761	-263.68%
77	2175073	7625629	-250.59%
78	2292907	7442296	-224.58%
79	2143121	7362919	-243.56%
80	2082197	7467859	-258.65%
96	2074433	5441201	-162.30%
97	2072599	5354775	-158.36%
98	2148441	5488474	-155.46%
99	2128207	5275928	-147.90%
100	2092563	5187785	-147.92%

Ответы на контрольные вопросы

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

Разреженная матрица – это такая матрица, в которой количество нулевых элементов в ней позволяет извлечь выгоду за счет их игнорирования.

Существуют разные схемы хранения разреженных матриц –

1. Хранение матрицы в виде набора троек строка – столбец – значение.

2.Связная схема Кнута – избыточная, но позволяет легко осуществлять любые операции с элементами матрицы.

3.Разреженный строчный формат Чанга и Густавсона.

В одном массиве хранятся ненулевые значения матрицы в порядке обхода по строкам, во втором соответствующие индексы столбцов, в третьем – индекс элемента в первом массиве, с которого начинается описание i-ой строки. Также можно поменять местами строки и столбцы(CSC)

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

В моей реализации под все 3 матрицы, хранимые 2 способами, выделяется статическая память. Стандартная матрица занимает 4МБ, разреженная – 8МБ.

3. Каков принцип обработки разреженной матрицы?

Принцип обработки разреженной матрицы заключается в том, чтобы обрабатывать только ненулевые элементы, что позволяет ускорить обработку относительно стандартных алгоритмов при малом проценте ненулевых элементов в матрице.

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

Стандартные алгоритмы обработки матриц эффективнее применять, когда процент ненулевых элементов больше 10-20, а также, если алгоритм обработки зависит, в том числе, и от нулевых элементов матрицы.

Вывод

При выполнении над разреженными матрицами таких операций, как сложение, стоит использовать специальные алгоритмы для разреженных матриц. Это позволяет уменьшить время обработки на 99% при очень низкой заполненности и остается эффективным до 20% заполненности. Данный эффект достигается за счет того, что при сложении обрабатываются только ненулевые элементы матрицы.

Также, если матрица не близка к квадратной, стоит задуматься над конкретным видом разреженной матрицы – по столбцам или по строкам. Так, при использовании алгоритма хранения CSC сложение матриц из малого количества строк по сравнению со столбцами дает стабильный выигрыш в 30% при любом уровне заполненности, но сложение матриц из большого кол-ва строк по сравнению со столбцами (1000 на 10) стабильно дольше в 2 раза при любом уровне заполненности.