Министерство образования и науки 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 (ИУ)

КАФЕДРА: Программное обеспечение ЭВМ и информационные технологии (ИУ7)

ОТЧЕТ ПО ЛАБРАТОРНОЙ РАБОТЕ №3

Название предмета: Типы и структуры данных

Студент: Кашима Ахмед

Группа: ИУ7-33Б

лаборант : рыбкин Ю.А.

20222 г.

Описание условия 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 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

Входные данные:

1. **Целое число, представляющее собой номер команды:** целое число в диапазоне от **0** до 7.
2. **Командно-зависимые данные:**
   - количество строк/столбцов матрицы, количество элементов матрицы, элементы матрицы в формате **“индекс строки - индекс столбца – значение элемента”**;
   - количество строк/столбцов матрицы, процент ненулевых элементов матрицы.

Выходные данные:

1. Исходные и результирующая матрицы в стандартном виде или разреженном столбцовом виде.
2. Количественная характеристика сравнения вариантов сложения матриц.

Обращение к программе:

Исполняемый файл app.exe создается путем автоматической сборки проекта с помощью файла makefile. Для выполнения работы следует запустить данный исполняемый файл без каких-либо аргументов.

Аварийные ситуации:

1. Некорректный ввод номера команды.
   На входе: число, большее чем 6 или меньшее, чем 0.
   На выходе: сообщение «Введена недопустимая команда! Повторите попытку.»
2. Некорректный ввод количества строк или столбцов матрицы.
   На входе: неположительное целое число или буква.
   На выходе: сообщение «Введено недопустимое значение! Повторите поппытку.»
3. Некорректный ввод индекса строки или столбца матрицы.
   На входе: число , выходящее за границы луча [0;количество_столбцов(строк)), или буква.
   На выходе: сообщение «Введено недопустимое значение! Повторите поппытку.»
4. Некорректный ввод элемента матрицы.
   На входе: число, выходящее за границы условия, или буква.
   На выходе: сообщение «Введено недопустимое значение! Повторите поппытку.»

Структуры данных

За стандартное хранение матрицы отвечает именованная структура **matrix_t**, описанная как:

```
typedef struct
{
   type_t **matrix;
   int rows;
   int cols;
} matrix_t;
```

Поля структуры:

- ***type_t* **matrix –** массив указателей на строки матрицы (**type_t - int**);
- ***int* rows –** количество строк матрицы;
- ***int* columns –** количество столбцов матрицы.

За хранение матрицы в разреженном столбцовом виде отвечает именованная структура **sparse_t**, описанная как:

```
typedef struct
{
   type_t *elems;
   int *row_entry;
   int *col_entry;
   int elems;
```

```
    int cols;
} matrix_t;
```

Поля структуры:

- ***type_t* *elems –** массив элементов матрицы, заполняемый проходом по столбцам;
- ***int* *row_entry –** массив, каждый элемент которого равен номеру строки соответствующего элемента из **elems**;
- ***int* *col_entry –** массив, каждый элемент которого указывает на индекс элемента из **elems**, с которого начинается описание столбца;
- ***int* elems –** количество элементов матрицы;
- ***int* cols –** количество столбцов матрицы.

Алгоритм

1. Пользователь вводит номер команды из меню.
2. Пока пользователь не введет 0 (Выход из программы), ему будет предложено выполнять действия с матрицами.
3. При вводе (или генерации) матрицы, матрица сразу хранится двумя способами хранения (стандартном и разреженном столбцовом).
4. В случае выбора стандартного сложения, матрицы складываются в стандартном виде “элемент к элементу”.
5. В случае выбора разреженного столбцового сложения, действия производятся непосредственно над разреженными столбцовыми матрицами. Сравнивается каждый столбец, при этом каждый из них сначала рассматривается как массив построчных вхождений. Если в массивах нет одинаковых вхождений, то в итоговую матрицу элементы записываются по порядку, в порядке возрастания по двум массивам. Если же одинаковые вхождения есть, то соответствующие элементы складываются, после все элементы вновь записываются в порядке возрастания индексов вхождения.

Тесты

	Тест	Пользовательский ввод	Результат
1	Некорректный ввод комманды	8	Incorrect Input! Try again.
2	Некорректный ввод количества строк	0	Incorrect Input! Try again.
3	Некорректный ввод количества строк	A	Incorrect Input! Try again.
4	Некорректный воод количества столбцов	-2	Incorrect Input! Try again.
5	Некорректный ввод количества столбцов	B	Incorrect Input! Try again.
6	Некорректный ввод индекса строки	При матрице 4x4: 4	Incorrect Input! Try again.
7	Некорректный ввод индекса строки	A	Incorrect Input! Try again.
8	Некорректный ввод индекса столбца	При матрице 3x3: 5	Incorrect Input! Try again.
9	Некорректный ввод индекса столбца	C	Incorrect Input! Try again.
10	Некорректный ввод элемента матрицы	D	Incorrect Input! Try again.
11	Некорректный ввод элемента матрицы	2.1	Incorrect Input! Try again.

Оценка эффективности

Измерения эффективности способов сложения будут производиться в единицах измерения – тактах процессорв. Для измерения была специально написана ассемблерная функция, поэтому погрешность измерений минимальна. При записи результатов использовалось среднее количество тактов, полученное по результатам 10 измерений.

Время сложения:

5% заполнения

Размеры	Разреженная матрица	Обычная матрица
10x10	1060	11900
100x100	33800	1549820
200x200	119080	6947080

10% заполнения

Размеры	Разреженная матрица	Обычная матрица
10x10	1160	8860
100x100	66920	579780
200x200	227680	1590860

20% заполнения

Размеры	Разреженная матрица	Обычная матрица
10x10	1540	7620
100x100	310940	388600
200x200	437380	3946040

30% заполнения

Размеры	Разреженная матрица	Обычная матрица
10x10	7200	20120
100x100	432400	1293560
200x200	648280	1544520

40% заполнения

Размеры	Разреженная матрица	Обычная матрица
10x10	11300	5820
100x100	11847640	388600
200x200	46828840	1608580

50% заполнения

Размеры	Разреженная матрица	Обычная матрица
10x10	25680	8020
100x100	6765340	1816860
200x200	52933140	1568680

Объём занимаемой памяти (в байтах):

5% заполнения

Размеры	Разреженная матрица	Обычная матрица
10x10	112	496
100x100	4432	40816
200x200	16832	161616

10% заполнения

Размеры	Разреженная матрица	Обычная матрица
10x10	152	496
100x100	8832	40816
200x200	32832	161616

20% заполнения

Размеры	Разреженная матрица	Обычная матрица
10x10	232	496
100x100	16432	40816
200x200	66456	161616

30% заполнения

Размеры	Разреженная матрица	Обычная матрица
10x10	312	496
100x100	24432	40816
200x200	98336	161616

40% заполнения

Размеры	Разреженная матрица	Обычная матрица
10x10	392	496

100x100	32432	40816
200x200	130664	161616

45% заполнения

Размеры	Разреженная матрица	Обычная матрица
10x10	432	496
100x100	37360	40816
200x200	146304	161616

50% заполнения

Размеры	Разреженная матрица	Обычная матрица
10x10	472	496
100x100	43328	40816
200x200	162584	161616

Вывод

При работе с большими не нулевые элементы разреженный матриц проиграет тест на время выполнении алгоритмов и также на объем данных.

Dimension	per(%)	Standard: time	size	Sparse: time	size
10	5	3472	496B	2226	112B
10	10	3258	496B	1930	152B
10	20	3287	496B	6014	232B
10	30	3308	496B	6064	312B
10	40	3335	496B	11380	392B
10	50	3315	496B	15774	472B
50	5	15985	10416B	5418	1232B
50	10	16460	10416B	15420	2232B
50	20	16066	10416B	52458	4232B
50	30	16058	10416B	116826	6232B
50	40	16122	10416B	210658	8232B
50	50	16013	10416B	332894	10232B
100	5	65738	40816B	27354	4432B
100	10	65704	40816B	91790	8432B
100	20	67402	40816B	414928	16432B
100	30	65619	40816B	937186	24432B
100	40	67161	40816B	1712728	32432B
100	50	67662	40816B	2629234	40840B
200	5	267258	161616B	184060	16832B
200	10	260282	161616B	1022164	34648B
200	20	266674	161616B	3493208	66872B
200	30	266611	161616B	7455678	98896B
200	40	260711	161616B	12422502	131040B
200	50	260793	161616B	19170804	164984B
400	5	1033243	643216B	1692904	67000B

так как у стандартного матрица стандартная время выполнении действие и измерение у нас в тактах процессора поэтому бывает Погрешности время выполнении на 1-5%.

Контрольные вопросы

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

Разреженная матрица – это матрица, содержащая большое количество нулей. Способы хранения: связная схема хранения, строчный формат, столбцовый формат, линейный связный список, кольцевой связный список, двунаправленные стеки и очереди.

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

Под обычную матрицу выделяет N * M ячеек памяти, где N – строки, а M – столбцы. Для разреженной матрицы – зависит от способа. В случае разреженного формата, требуется 3 * K ячеек памяти, где K – количество ненулевых элементов.

3. Каков принцип обработки разреженной матрицы?

Алгоритмы обработки разреженных матриц предусматривают действие 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?

Стандартные алгоритмы обработки матриц эффективнее применять при большом количестве ненулевых элементов (от 40%). Стоит отметить, что если не так важна память, занимаемая матрицами, но важно время, то в случае сложения лучше так же воспользоваться стандартными алгоритмами сложения матриц.

Вывод

Использование алгоритмов хранения и обработки разреженных матриц в случае сложения выгодно при малом количестве ненулевых элементов, примерно до 40% заполненности матриц. Хранение матриц в разреженном столбцовом формате проигрывает по памяти практически во всех случаях, так как структура для хранения разреженных матриц довольно нагружена целочисленными полями для хранения различных компонент матриц. Помимо занимаемой памяти, разреженные матрицы гораздо дольше обрабатываются при заполненности матриц от 40%, так как требуется много времени для составления портрета матрицы.