|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **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 Егорова Полина Александровна

Группа ИУ7 – 34Б

Преподаватель Барышникова Марина Юрьевна

2021 г.

**Цель работы**

Цель работы - реализовать алгоритмы обработки разреженных матриц, сравнить эффективность использования этих алгоритмов (по времени выполнения и по требуемой памяти) со стандартными алгоритмами обработки матриц при различном процентном заполнении матриц ненулевыми значениями и при различных размерах матриц.

**Задание**

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 вектора-строки и матрицы, 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.

**Входные данные**

Для генерации матрицы и вектора необходимо ввести количество строк и столбцов матрицы и процент заполненности ее и вектора ненулевыми элементами. При вводе матрицы и вектора вручную необходимо ввести количество строк и столбцов матрицы и количество ненулевых элементов матрицы и вектора.

**Выходные данные**

При выборе соответствующего пункта меню, будут выведены матрица или вектор либо в стандартной форме (п. 6 и 7), либо в разреженной (п. 2 и 3). При выборе пунктов 4 или 5 будет посчитано произведение вектора-строки на матрицу классическим или разреженным способом соответственно.

**Способ обращения к программе**

Программа может быть вызвана через консоль.

./app.exe

**Аварийные ситуации**

В случае аварийной ситуации выводится сообщение о той или иной ошибке.

Могут быть выведены такие ошибки, как:

* Некорректный ввод параметров матрицы или вектора
* Количество элементов, запрашиваемых на ввод, превышает возможное (исходя из размерностей) или меньше либо равно нулю
* Работа с пустой матрицей или вектором
* Некорректный ввод элемента, номера столбца или строки (буква)
* Попытка вставить элемент на несуществующею позицию матрицы (несоответствие размерностям)

**Структуры данных**

Для хранения матрицы в стандартном виде использовался динамическая матрица (способ реализации: как одномерный, для доступа к элементу используется адресная арифметика):

int \*matr = malloc(r \* c \* sizeof(int)); // r, c – количество строк и столбцов

Для хранения вектора в стандартном виде использовался динамический массив:

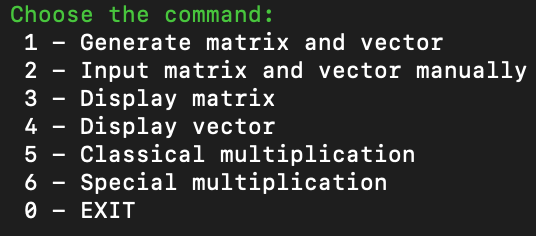
int \*vector = malloc(c \* sizeof(int));

Для хранения каждого из трех объектов (когда матрица представлена в разреженном виде) – JA, IA, A - используется односвязный список, представленный в виде структуры:

typedef struct list  
{  
 int data;  
 struct list \*next;  
} list;

**Описание алгоритма**

Данная программа предназначена для работы с разреженными и стандартными матрицами и представляет собой консольное приложение со следующими возможными операциями, представленными в меню:



1 – генерация матрицы и вектора

2 – ввод матрицы и вектора вручную

3 – вывод матрицы (возможность выбора типа: стандартная или разреженная)

4 - вывод вектора (возможность выбора типа: стандартная или разреженная)

5 – классическое умножение

6 – специальное умножение

0 - выход

**Реализация умножения**

Умножение вектора-строки и матрицы реализовано двумя способами: для объектов, представленных стандартным и разреженным способом. Умножение разреженных матриц осуществляется при помощи односвязного списка.

Для умножения должны быть введены матрица и вектор. Иначе будет выведено сообщение о том, что данные математические объекты пусты.

**Анализ эффективности разрежённого способа хранения**

Процент разреженности – процент нулевых элементов в матрице.

Память (стандартная матрица) – объем памяти в байтах, занимаемый вектором-ответом, представленным в стандартном виде.

Память (разреженная матрица) – объем памяти в байтах, занимаемый вектором-ответов в разреженной форме.

Время – время затраченное на выполнение умножения вектора-строки на матрицу (в тактах процессора).

**Сравнение памяти**

1. 0% разреженности

|  |  |  |
| --- | --- | --- |
| Количество элементов | Стандартная матрица (Б) | Разреженная матрица (Б) |
| 50\*50 | 200 | 200 |
| 100\*100 | 400 | 400 |
| 1000\*1000 | 4000 | 4000 |

1. 25% разреженности

|  |  |  |
| --- | --- | --- |
| Количество элементов | Стандартная матрица (Б) | Разреженная матрица (Б) |
| 50\*50 | 200 | 200 |
| 100\*100 | 400 | 400 |
| 1000\*1000 | 4000 | 4000 |

1. 50% разреженности

|  |  |  |
| --- | --- | --- |
| Количество элементов | Стандартная матрица (Б) | Разреженная матрица (Б) |
| 50\*50 | 200 | 200 |
| 100\*100 | 400 | 400 |
| 1000\*1000 | 4000 | 4000 |

1. 75% разреженности

|  |  |  |
| --- | --- | --- |
| Количество элементов | Стандартная матрица (Б) | Разреженная матрица (Б) |
| 50\*50 | 200 | 188 |
| 100\*100 | 400 | 400 |
| 1000\*1000 | 4000 | 4000 |

1. 98% разреженности

|  |  |  |
| --- | --- | --- |
| Количество элементов | Стандартная матрица (Б) | Разреженная матрица (Б) |
| 50\*50 | 200 | 8 |
| 100\*100 | 400 | 16 |
| 1000\*1000 | 4000 | 1388 |

**Сравнение времени**

1. 0% разреженности

|  |  |  |
| --- | --- | --- |
| Количество элементов | Стандартная матрица (T) | Разреженная матрица (T) |
| 50\*50 | 12 | 15 |
| 100\*100 | 54 | 70 |
| 1000\*1000 | 4462 | 4034 |

1. 25% разреженности

|  |  |  |
| --- | --- | --- |
| Количество элементов | Стандартная матрица (T) | Разреженная матрица (T) |
| 50\*50 | 12 | 12 |
| 100\*100 | 44 | 60 |
| 1000\*1000 | 4560 | 4031 |

1. 50% разреженности

|  |  |  |
| --- | --- | --- |
| Количество элементов | Стандартная матрица (T) | Разреженная матрица (T) |
| 50\*50 | 10 | 8 |
| 100\*100 | 48 | 29 |
| 1000\*1000 | 4899 | 2121 |

1. 75% разреженности

|  |  |  |
| --- | --- | --- |
| Количество элементов | Стандартная матрица (T) | Разреженная матрица (T) |
| 50\*50 | 12 | 6 |
| 100\*100 | 48 | 20 |
| 1000\*1000 | 4657 | 1027 |

1. 98% разреженности

|  |  |  |
| --- | --- | --- |
| Количество элементов | Стандартная матрица (T) | Разреженная матрица (T) |
| 50\*50 | 12 | 4 |
| 100\*100 | 38 | 7 |
| 1000\*1000 | 6360 | 110 |

Затраты по памяти уменьшаются при использовании алгоритма умножения разреженных матриц с процентом разреженности не менее 75. Менее 75% количество используемых байт уравнивается.

Использование разреженной матрицы для выигрыша по времени оправдано при работы с матрицами с процентом разреженности не менее 50%.

**Контрольные вопросы**

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

Разреженная матрица – матрица с преимущественно нулевыми элементами. Число ненулевых элементов в матрице порядка n может выражаться как n^(1+g), где g < 1. Значения g лежат в интервале 0.2 ... 0.5, т.е. матрица разрежена.

Существуют различные методы хранения элементов матрицы в памяти.

Например, линейный связный список, т.е. последовательность ячеек, связанных в определенном порядке. Каждая ячейка списка содержит элемент списка и указатель на положение следующей ячейки.

Можно хранить матрицу, используя кольцевой связный список, двунаправленные стеки и очереди.

Существует диагональная схема хранения симметричных матриц, а также связные схемы разреженного хранения.

Связная схема хранения матриц, предложенная Кнутом, предлагает хранить в массиве (например, в AN) в произвольном порядке сами элементы, индексы строк и столбцов соответствующих элементов (например, в массивах I и J), номер (из массива AN) следующего ненулевого элемента, расположенного в матрице по строке (NR) и по столбцу (NC), а также номера элементов, с которых начинается строка (указатели для входа в строку – JR) и номера элементов, с которых начинается столбец (указатели для входа в столбец - JC).

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

Для хранения обычной матрицы: N \* M \* sizeof(elem). Память под разреженную матрицу выделяется в зависимости от схемы хранения. Кроме того, память зависит от количества ненулевых элементов.

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

Обработка разреженной матрицы предполагает работу только с ненулевыми элементами (таким образом, количество операций пропорционально количеству ненулевых элементов).

***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**

Использование разреженной матрицы оправдано при большом количестве нулевых элементов (разреженность > 30%). При таких размерностям с увеличением количества элементов возрастает выигрыш в большей мере по времени работы, нежели по памяти.

Время выполнения стандартного алгоритма зависит от размерности матрицы. Этот алгоритм эффективен при высоком заполнении матрицы. Однако при заполнении матрицы менее 75% разреженный алгоритм позволяет добиться более высокой скорости работы при использовании меньшего количества памяти.