

Лабораторная работа №5
по дисциплине «Типы и структуры данных»
Тема :Обработка разреженных матриц.

Условие задачи:

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

- вектор A содержит значения ненулевых элементов;
- вектор JA содержит номера столбцов для элементов вектора A;
- связный список IA, в элементе Nk которого находится номер компонент в A и JA, с которых начинается описание строки Nk матрицы A.

Задание:

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.

Исходные данные:

- Матрица (количество строк m и столбцов n вводится пользователем), размерностью $m \times n$, где m и n - целые числа в интервале [1;500], содержащая целые положительные числа от 0 до 9.
- Вектор из n элементов, где n не превышает 500, состоящий из целых положительных чисел от 0 до 9.

Пользователь вводит размерность матрицы и выбирает способ заполнения матрицы и вектора: вручную с клавиатуры либо случайными числами (при выборе заполнения матрицы и вектора случайными числами пользователь должен ввести процент заполнения матрицы нулями).

Выходные данные:

В результате работы программы на экран выводится:

- Вектор-столбец из m элементов, полученный в результате перемножения исходной матрицы и вектора столбца из n

элементов с применением стандартного алгоритма умножения матрицы на вектор.

- Вектор-столбец из m элементов в форме, состоящей из трех векторов (в соответствии с условием), полученный в результате перемножения матрицы, хранящейся в форме, состоящей из трех векторов, и вектора, хранящегося в той же форме.
- Количество времени, затраченное на умножение в матричной форме. Количество времени, затраченное на умножение в форме из трех векторов.
- Количество памяти, используемое при умножении двумя способами.

Представление матрицы в форме трех векторов:

Пример:

матрица 5×4

9	0	0	4
0	5	4	0
0	0	0	1
1	6	1	0
0	0	8	0

A содержит значения ненулевых элементов матрицы:

A[9] = {9,4,5,4,1,1,6,1,8}

JA содержит номера столбцов для элементов вектора **A**:

JA[9] = {0,3,1,2,3,0,1,2,2}

(нулевой элемент вектора **A** - 9 находится в матрице в нулевом столбце, первый элемент вектора **A** - 4 находится в матрице в 3-ем столбце и т.д.)

IA, в элементе N_k которого находится номер компонент в **A** и **JA**, с которых начинается описание строки N_k матрицы **A**:

IA[5] = {0,2,4,5,8} - состоит из m элементов (по количеству строк)

(в первой строке матрицы хранятся элементы 9 и 4, которые в **A** и **JA** располагаются с индексами 0 и 1. В **IA** заносится индекс первого элемента на этой строке - 0. Во второй строке матрицы хранятся элементы 5 и 4, которые в **A** и **JA** располагаются по индексам 2 и 3. В **IA** заносится индекс первого элемента на этой строке - 2 и т.д. Если в строке нет ненулевых элементов, то в **IA** заносится -1.)

Возможные ошибки пользователя:

Неверный ввод размерности матрицы. Пользователь вводит числа, не принадлежащие интервалу [1;500]. Реакция - сообщение "Размерность матрицы - 2 числа в интервале [1;500]: " и повтор ввода.

Неверный выбор пункта меню при выборе способа заполнения матрицы или вектора. Пользователь вводит число, не равное 1 или 2 на просьбу "Выберите способ ввода матрицы: %s(1)Ручной ввод / (2)Random:". Реакция - повтор ввода.

Тестовые данные:

Размерность	Матрица	Векторная форма представления матрицы	Вектор - столбец	Результат умножения (вектор - столбец)	Векторная форма представления вектора - столбца
5 5	0 7 0 4 0 0 0 0 0 0 6 0 0 0 0 0 0 0 0 0 0 8 4 0 8	A: 7 4 6 8 4 8 JA: 1 3 0 1 2 4 IA: 0 -1 2 -1 3	5 3 1 9 6	57 0 30 0 76	A: 57 30 76 JA: 0 0 0 IA: 0 -1 1 -1 2
5 1	8 0 7 2 0	A: 8 7 2 JA: 0 0 0 IA: 0 -1 1 2 -1	8	64 0 56 16 0	A: 64 56 16 JA: 0 0 0 IA: 0 -1 1 2 -1
1 5	7 1 0 0 0	A: 7 1 JA: 0 1 IA: 0	0 3 0 9 0	3	A: 3 JA: 0 IA: 0
-1 5	Размерность матрицы - 2 числа в интервале [1;500]				

Сравнение эффективности по времени двух методов умножения матриц

Размерность матрицы	Заполнение матрицы ненулевыми элементами	Стандартный алгоритм (время выполнения)	Векторный алгоритм (время выполнения)
3*3	80%	1564	1638
3*3	50%	1390	1602
3*3	20%	1203	1518
10*10	80%	5014	8625
10*10	50%	5919	6001
10*10	20%	4192	3464
30*30	80%	20 268	26 433
30*30	50%	24 771	23 149
30*30	20%	25 265	10 307
100*100	80%	185 564	232 202
100*100	50%	191 144	156 883
100*100	20%	215 532	66 301

Сравнение эффективности по памяти двух методов умножения матриц

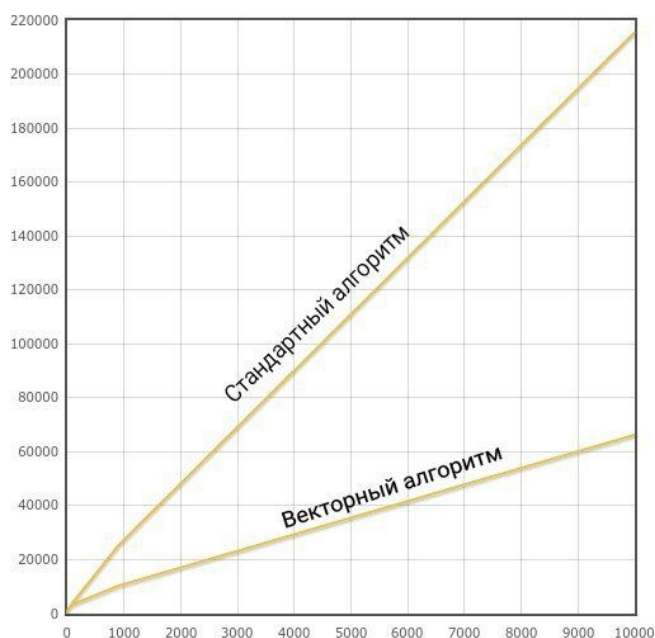
Размерность матрицы	Заполнение матрицы ненулевыми элементами	Стандартный алгоритм (память)	Векторный алгоритм (память)
3*3	80%	60	132
3*3	20%		100
10*10	80%	480	936
10*10	20%		328
30*30	80%	3840	6696
30*30	20%		1560
100*100	80%	40800	67968
100*100	20%		10264

Примечание: жирным шрифтом в таблицах помечаются строки, где процент заполнения матрицы ненулевыми элементами меньше $\frac{1}{4}$ т.е. матрица разреженная.

Вывод: выбор более эффективного алгоритма умножения матрицы на вектор-столбец напрямую зависит от размерности этой матрицы (т.е. количества элементов). При малом количестве элементов эффективнее использовать стандартный алгоритм работы с матрицами, так как в алгоритме, построенном на работе с тремя векторами, производится много проверок и циклов. Памяти также используется меньше в первом алгоритме, так как память требуется только на хранение матрицы и векторов (исходного и результата).

При возрастании количества элементов эффективность второго алгоритма становится более наглядной. Так как при большом количестве элементов (нулевых в том числе) первый алгоритм работает со всеми элементами матрицы и вектора, а второй только с ненулевыми. При большом количестве элементов второй алгоритм также эффективнее по памяти и эта эффективность возрастает с ростом количества элементов (при условии разреженности матрицы). Так как на хранение матрицы и вектора в матричной форме требуется всегда постоянное количество памяти, а в векторах хранится информация только о ненулевых элементах, что позволяет использовать в разы меньше памяти.

На графике: по оси X - количество элементов матрицы
по оси Y - время, потраченное на умножение



Контрольные вопросы:

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

Разреженная матрица содержит достаточно большое количество элементов, из которых лишь малая часть является ненулевыми (n^{1+g} для матрицы размерности n , $g < 1$).

Простейшая схема хранения такой матрицы - массив ненулевых элементов (AN) и два массива их координат (I и J)

Связная схема хранения матриц, предложенная Кнудом, 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?

Под хранение обычной матрицы требуется $m \cdot n \cdot \text{sizeof}(\text{type_of_element})$ байт.

Память под разреженную матрицу выделяется в зависимости от метода ее хранения.

3. Каков принцип обработки разреженной матрицы?

Принцип обработки разреженной матрицы предполагает работу только с ненулевыми элементами массива.

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

Эффективность использования стандартного алгоритма обработки матриц зависит от количества ненулевых элементов в матрице. Чем их больше, тем менее эффективно использовать стандартные алгоритмы.