

**Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ ИМ. Р. Е. АЛЕКСЕЕВА»
(НГТУ)**

Институт радиоэлектроники и информационных технологий

Кафедра информатики и систем управления

**Методические указания
по выполнению лабораторных работ
по дисциплине «ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ»
Тема: «Разреженные матрицы»**

Направление подготовки
09.03.02 «Информационные системы и технологии»
код и наименование направления подготовки

Профиль подготовки
«Безопасность информационных систем»
наименование профиля подготовки

Уровень высшего образования
Бакалавр

Форма обучения
Очная

Нижний Новгород

2015

Разработчик(и)/составитель(и) методических указаний по выполнению лабораторных работ по дисциплине «Технологии программирования»:

к.т.н., Капранов С.Н.

ученое звание, степень, фамилия, инициалы

Кафедра «Информатика и системы управления»

наименование кафедры

Дата, подпись  23.04.15

Методические указания по выполнению лабораторных работ по дисциплине «Технологии программирования» рассмотрены на заседании кафедры «Информатика и системы управления»

наименование кафедры

Протокол № 6 от « 24 » 04 20 15 г.

Заведующий кафедрой ИСУ _____ д.т.н., профессор
Э.С.Соколова

ученое звание, степень фамилия, имя, отчество

Дата, подпись *Сосау*

Методические указания по выполнению лабораторных работ по дисциплине «Технологии программирования» утверждены учебно-методическим советом института радиоэлектроники и информационных технологий

Протокол № 3 от «24» 04 2015 г.

Методические указания зарегистрированы в учебном отделе под учетным номером 1162 .

Ведущий инженер

Чуева Чуева Н.А. 17.03.2016

Разреженные матрицы

Традиционно считается, что разреженные данные — это данные, которые в основном содержат нулевые, или пустые, значения. В некоторых случаях разреженность данных может быть очевидной. Рассмотрим, например, нанесение на карту всех видимых на небе звезд. Очевидно, что на ночном небе будет гораздо больше пустого пространства, чем звезд, видимых невооруженным глазом (в противном случае ночное небо было бы очень ярким). Несмотря на то, что количество видимых звезд очень велико, тем не менее, они распределены по огромной площади. Таким образом, количество видимых звезд по сравнению с количеством мест, которые могли бы быть заняты видимыми звездами, может служить примером разреженных данных.

Разреженность данных определяется не количеством данных, а отношением занятых позиций данных к количеству всех возможных позиций. Такое отношение определяет разреженность данных. Для обоснования использования разреженной матрицы для данных той или иной степени разреженности необходимо оценить скорость работы и сложность кода, а также объем сэкономленной при этом памяти.

Разреженными называются матрицы, в которых ненулевых элементов много меньше общего числа элементов.

Хранение разреженной матрицы в памяти должно обеспечивать:

1) экономию памяти

– $N \times M$ – размерность матрицы

– NZ – количество ненулевых элементов,

– $Z \ll N \times M$.

2) быстрый доступ к нулевым и ненулевым элементам по их индексу.

Разреженные матрицы используются для хранения сравнительно небольшого объема данных, которые располагаются в большой области данных. Реализация разреженных матриц связана со значительными издержками, и это делает их все более непрактичными по мере заполнения области данных значимыми величинами и в тоже время при большей разреженности эти структуры становятся более эффективными. Таким образом, существует хорошо определенный набор задач, для решения которых необходимо использовать структуру данных типа *разреженной матрицы*. К таким задачам можно отнести:

1. численное решение больших разреженных систем уравнений
2. представление графов в виде разреженных матриц смежности для решения задач различного типа

Хранение разреженной матрицы общего вида.

Координатный формат хранения

Самым простым способом хранения произвольной разреженной матрицы является координатный формат: хранятся только ненулевые элементы матрицы, и их координаты (номера строк и столбцов).

Структура хранения

Матрица храниться в виде трех одномерных массивов.

Все ненулевые элементы a_{ij} построчно записываются в одномерный массив A .

Первый индекс (номер строки) каждого ненулевого элемента записывается в одномерный массив LI.

Второй индекс (номер столбца) каждого ненулевого элемента записывается в одномерный массив LJ.

Пример, матрица B

0	0	2	0	0	0
0	0	0	0	0	0
0	5	0	0	0	0
0	0	0	0	9	1
0	0	0	0	0	3
8	0	0	4	0	0

```

N      1  2  3  4  5  6  7
      a13;a32;a45;a46;a56;a61;a64;
A      :  2; 5; 9; 1; 3; 8; 4;
LJ:     3; 2; 5; 6; 6; 1; 4;
LI:     1; 3; 4; 4; 5; 6; 6;

```

Координатный формат может быть:

- Упорядоченный (по строкам/столбцам)
 - Быстрый доступ к строкам/столбцам
 - Необходимость перепакровки при вставке/удалении элементов
- Неупорядоченный
 - «Переборный» доступ к элементам
 - Быстрая вставка/удаление элементов

При данном способе упаковки для матрицы $n \times m$ элементов с NZ ненулевыми элементами затраты памяти составят

- NZ ячеек под элементы массива
- NZ ячеек под массив LJ
- NZ ячеек под массив LI

Хотя многие математические библиотеки поддерживают матрично-векторные операции в координатном формате, данный формат обеспечивает медленный доступ к элементам матрицы, и является затратным по используемой памяти. В рассмотренном выше примере избыточность по памяти образом проявляется в массиве LI, в котором строчные координаты хранятся неоптимальным образом.

Перейдем далее к рассмотрению более экономных форматов хранения. Разреженный строчный формат — это одна из наиболее широко используемых схем хранения разреженных матриц. Эта схема предъявляет минимальные требования к памяти и в то же время оказывается очень удобной для нескольких важных операций над разреженными матрицами: сложения, умножения, перестановок строк и столбцов, транспонирования,

решения линейных систем с разреженными матрицами коэффициентов как прямыми, так и итерационными методами и т. д.

Разреженный строчный формат

(Compressed Row Storage **CRS** или Compressed Sparse Rows **CSR**)

Структура хранения

Матрица храниться в виде трех одномерных массивов.

Все ненулевые элементы a_{ij} построчно, с первой до последней строки, записываются в одномерный массив **A**.

Второй индекс каждого ненулевого элемента (номера столбцов) записывается в одномерный массив **LJ**.

Местоположение первого ненулевого элемента в каждой строке записывается в одномерный массив **LI**. Элементы i -ой строки хранятся в позициях матрицы **A** с номера (**LI**[i]) по (**LI**[$i+1$]-1). Если в строке i встречаются только нулевые элементы (строка является пустой), то значение **LI**[i] = **LI**[$i+1$]. Если матрица **A** состоит из n строк, то длина массива **LI** будет $(n+1)$.

Пример:

Рассмотрим матрицу **A**.

a_{11}	a_{12}	0	0	0
a_{21}	a_{22}	0	a_{24}	0
0	0	a_{33}	a_{34}	0
0	a_{42}	a_{43}	a_{44}	a_{45}
0	0	0	a_{54}	a_{55}

N 1 2 3 4 5 6 7 8 9 10 11 12 13
A: $a_{11}; a_{12}; a_{21}; a_{22}; a_{24}; a_{33}; a_{34}; a_{42}; a_{43}; a_{44}; a_{45}; a_{54}; a_{55};$
LJ: 1; 2; 1; 2; 4; 3; 4; 2; 3; 4; 5; 4; 5;
LI: 1; 3; 6; 8; 12; **14;**

Последний элемент (14) в матрице **LI** определяет количество ненулевых элементов+1. Это нужно для того что бы не выйти за границы массива **A** при просмотре последней строки исходной матрицы.

Пример с числами, матрица **B**

0	2	0	1	0	0
0	0	0	0	0	0
0	5	0	0	0	1
0	0	0	0	9	1
0	0	0	0	0	0
0	8	0	4	0	3

N 1 2 3 4 5 6 7 8 9
 a₁₂ ; a₁₄ ; a₃₂ ; a₃₆ ; a₄₅ ; a₄₆ ; a₆₂ ; a₆₄ ; a₆₆ ;
A: 2; 1; 5; 1; 9; 1; 8; 4; 3;
LJ: 2; 4; 2; 6; 5; 6; 2; 4; 6;
LI: 1; 3; 3; 5; 7; 7; **10;**

При данном способе упаковки для матрицы $n \times m$ элементов с NZ ненулевыми элементами затраты памяти составят

- NZ ячеек под элементы массива
- NZ ячеек под массив LJ
- $n+1$ ячейка под массив LI

Алгоритм доступа к элементу с индексами i, j :

```

int procedure(i, j)
{
    AA=0; // значение искомого элемента
    N1=LI[i];
    N2=LI[i+1];
    for(k=N1; k<N2; k++)
    {
        if (LJ[k]==j)
        {
            AA=A[k];
            <break>;
        }
    }
    return AA;
}

```

Данный способ представления также является полным. и упорядоченным. поскольку элементы каждой строки хранятся в соответствии с возрастанием столбцовых индексов.

Разреженный строчный формат обеспечивает эффективный доступ к строкам матрицы, но затруднен доступ к столбцам. Поэтому предпочтительно использовать этот способ хранения в тех алгоритмах, в которых преобладают строчные операции.

Модификации:

Иногда бывает удобно использовано полный неупорядоченный способ хранения, при котором внутри каждой строки элементы могут храниться в произвольном порядке.

Результаты многих матричных операций получаются неупорядоченными, и упорядочивание может быть весьма затратным. В то же время, многие алгоритмы для разреженных матриц не требуют, чтобы представление было упорядоченным.

Разреженный столбцовый формат

(Compressed Column Storage CCS или Compressed Sparse Columns CSC)

Если в решаемой задаче необходимо осуществлять доступ к элементам по столбцам, то схему хранения можно изменить. Хранить элементы можно не по строкам, а по столбцам.

После рассмотрения строчного формата хранения очевидным является и разреженный столбцовый формат.

Структура хранения

Матрица хранится в виде трех одномерных массивов.

Все ненулевые элементы a_{ij} построчно записываются в одномерный массив A.

Первый индекс каждого ненулевого элемента записывается в одномерный массив LI.

Местоположение первого ненулевого элемента в каждом столбце записывается в одномерный массив LJ.

Если в столбце i встречаются только нулевые элементы, то значение $LJ[i] = LJ[i+1]$;

Столбцовые представления могут рассматриваться как строчные представления транспонированных матриц. Разреженный столбцовый формат обеспечивает эффективный доступ к столбцам матрицы, но в тоже время доступ к строкам затруднен. Поэтому предпочтительно использовать этот способ хранения в тех алгоритмах, в которых преобладают столбцовые операции.

Пример с числами, матрица B

Исходная матрица

0	2	0	1	0	0
0	0	0	0	0	0
0	5	0	0	0	1
0	0	0	0	9	1
0	0	0	0	0	0
0	8	0	4	0	3

Схема хранения

N:	1	2	3	4	5	6	7	8	9
A:	2	5	8	1	4	9	1	1	3
LI:	1	3	6	1	6	4	3	4	6
LJ:	1	1	4	4	6	7	10		

```

N   1  2  3  4  5  6  7  8  9
    a21;a14;a32;a36;a45;a46;a62;a64;a66;
A:  2; 5;  8; 1;  4; 9;  1; 1; 3;
LI: 1; 3;  6; 1;  6; 4;  3; 4; 6;

```

LJ: 1; 1; 4; 4; 6; 7; 10;

Алгоритм доступа такой же, как и для **CRS**, только i и j меняются местами, а также LI и LJ.

6.1. Упаковка структурно симметричных матриц

Разреженная матрица называется структурно симметричной, если для неё выполняется следующее:

Если $a_{ij} \neq 0$, то и $a_{ji} \neq 0$

Если $a_{ij} = 0$, то и $a_{ji} = 0$

Такая матрица хранится следующим образом:

Все диагональные элементы записываются в массив AD. Ненулевые элементы стоящие над диагональю построчно записываются в одномерный массив AU, при этом их вторые индексы в том же порядке записываются в одномерный массив LJ. Значения элементов стоящих под диагональю по столбцам записываются в одномерный массив AL. Заметим, что их первые индексы уже находятся в LJ. Местоположение первого элемента в каждой строке (каждого столбца) записывается в массив LI.

- $LI[i]$ указывает на начало i -ой строки в массиве AU
- Элементы строки i в массиве AU находятся по индексам от $LI[i]$ до $LI[i + 1] - 1$ включительно
- Обрабатываются пустые строки ($LI[p] = LI[p + 1]$)
- Единообразно обрабатывается последняя строка ($LI[N]=NZ+1$).
- $LI[j]$ указывает на начало j -ого столбца в массиве AL
- Элементы столбца j в массиве AL находятся по индексам от $LI[j]$ до $LI[j + 1] - 1$ включительно
- Обрабатываются пустые столбцы ($LI[p] = LI[p + 1]$)
- Единообразно обрабатывается последний столбец ($LI[N]=NZ+1$).

Для рассмотренной ранее матрицы A

a_{11}	a_{12}	0	0	0
a_{21}	a_{22}	0	a_{24}	0
0	0	a_{33}	a_{34}	0
0	a_{42}	a_{43}	a_{44}	a_{45}
0	0	0	a_{54}	a_{55}

массивы заполняются следующим образом:

$AD: a_{11}; a_{22}; a_{33}; a_{44}; a_{55};$
 $AU: a_{12}; a_{24}; a_{34}; a_{45};$
 $AL: a_{21}; a_{42}; a_{43}; a_{54};$
 $LJ: 2 \quad 4 \quad 4 \quad 5$
 $LI: 1 \quad 2 \quad 3 \quad 4$

Исходная матрица

0	2	0	1	0	0
1	0	0	0	0	0
0	0	5	0	0	0
5	0	0	7	9	1
0	0	0	4	0	0
0	0	0	4	0	3

Схема хранения

AD:	0	0	5	7	0	3
AU:	2	1	9	1		
AL:	1	5	4	4		
LJ:	2	4	5	6		
LI:	1	3	3	3	5	5

□ Объем памяти для хранения массива $N \times N$ с $2 \cdot NZ$ ненулевыми элементами (исключая диагональ):

- Массив значений **AD** - N ячеек
- Массив значений **AU** - NZ ячеек
- Массив значений **AL** - NZ ячеек
- Массив **LJ** - NZ ячеек
- Массив **LI** - N ячеек

Алгоритм доступа:

Доступ к диагональному элементу тривиален. Если искомый элемент лежит в верхнем треугольнике, т.е. $i < j$, то применяем алгоритм из CRS, в противном случае применяем тот же алгоритм для массива AL, заменив в алгоритме i на j .

6.2. Использование связанных списков.

Вышерассмотренные способы упаковки не позволяют легко включать новые элементы и исключать обнулившиеся. Чтобы обеспечить возможность этих действий для рассмотренных ранее способов упаковки вместо линейных структур применяются связанные списки.

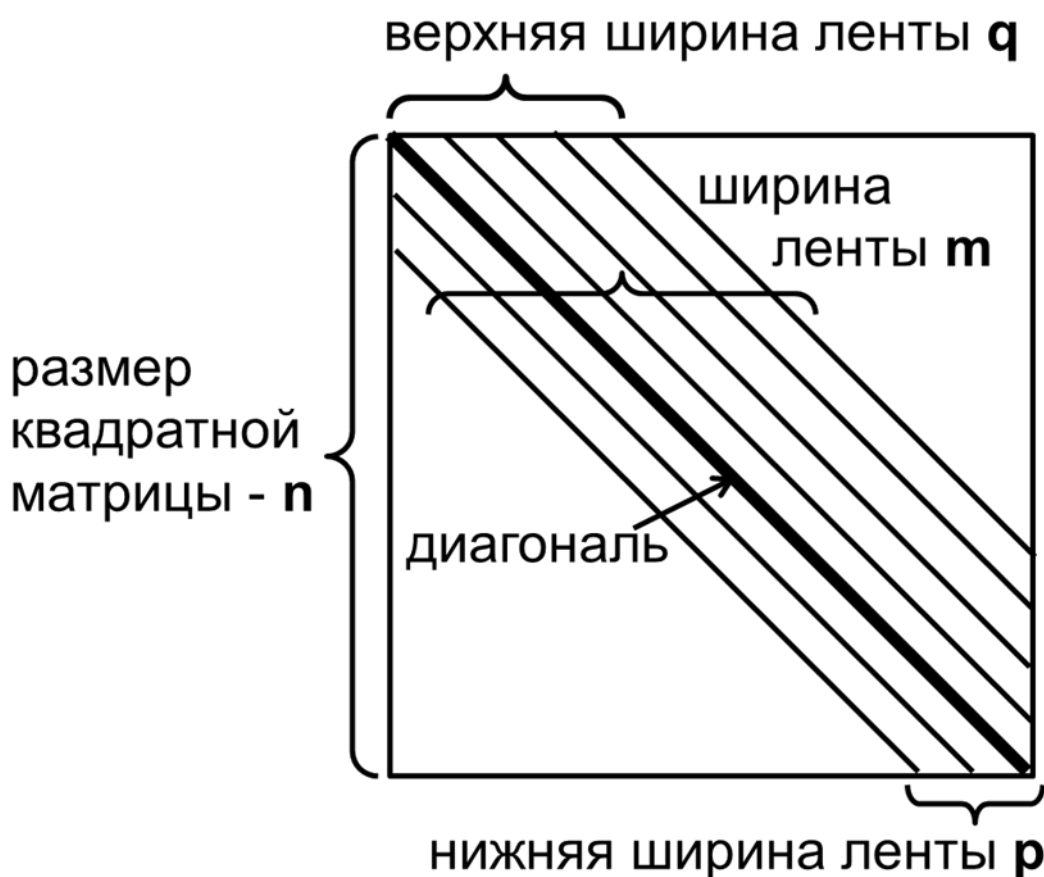
6.3. Ленточные матрицы

Матрица A называется ленточной, если все ее ненулевые элементы заключены внутри ленты, образованной между диагоналями, параллельными главной.

Ленточный формат используется, когда можно выделить плотную ленту, состоящую из ненулевых элементов и имеющую определенную ширину. Существует несколько модификаций этого формата.

Если для матрицы A справедливо: $a_{ij} = 0$ при $i > j + p$ и $a_{ij} = 0$ при $j > i + q$, то p называется нижней шириной ленты, q - верхней шириной ленты.

Величина $m = p + q + 1$ называется шириной ленты матрицы A .

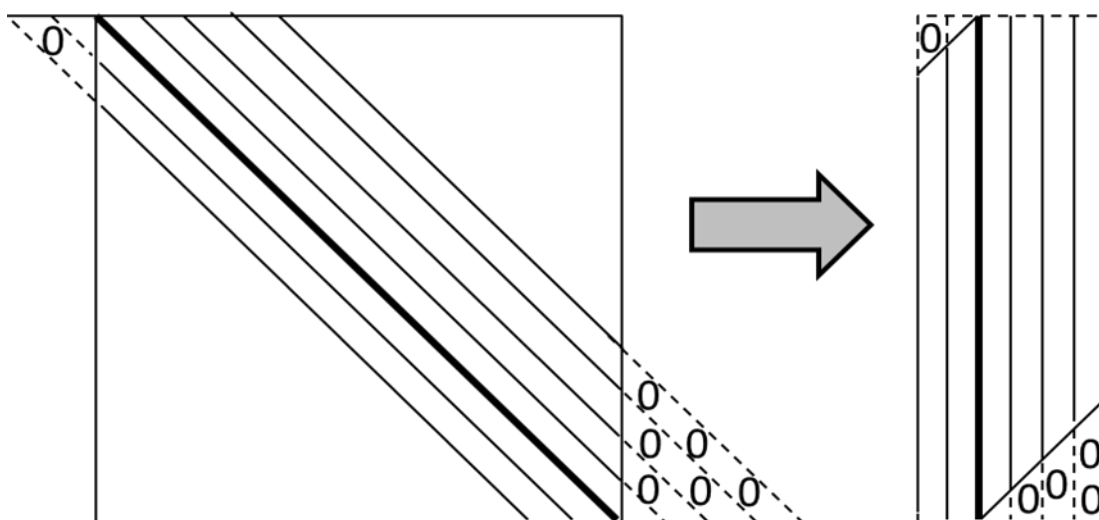


6.3.1. Ленточный строчный формат

Строчный ленточный формат для хранения исходной матрицы A использует массив размера $n \times m$, в котором *построчно* хранятся ненулевые элементы матрицы A .

Побочные диагонали доопределяются нулями до размерности n : в начале диагоналей для нижнего треугольника и в конце диагоналей для верхнего треугольника.

Рисунок ниже, иллюстрирует хранение матрицы по столбцам.



Пример

Исходная матрица A

$n=6; p=1; q=2; m=4;$

5	8	2	0	0	0
7	9	0	8	0	0
0	4	0	2	5	0
0	0	1	3	9	1
0	0	0	0	4	3
0	0	0	0	6	0

Упакованная матрица P

0	5	8	2
7	9	0	8
4	0	2	5
1	3	9	1
0	4	3	0
6	0	0	0

Элемент исходной матрицы a_{ij} расположен в упакованном массиве $P[i, j - i + p + 1]$,

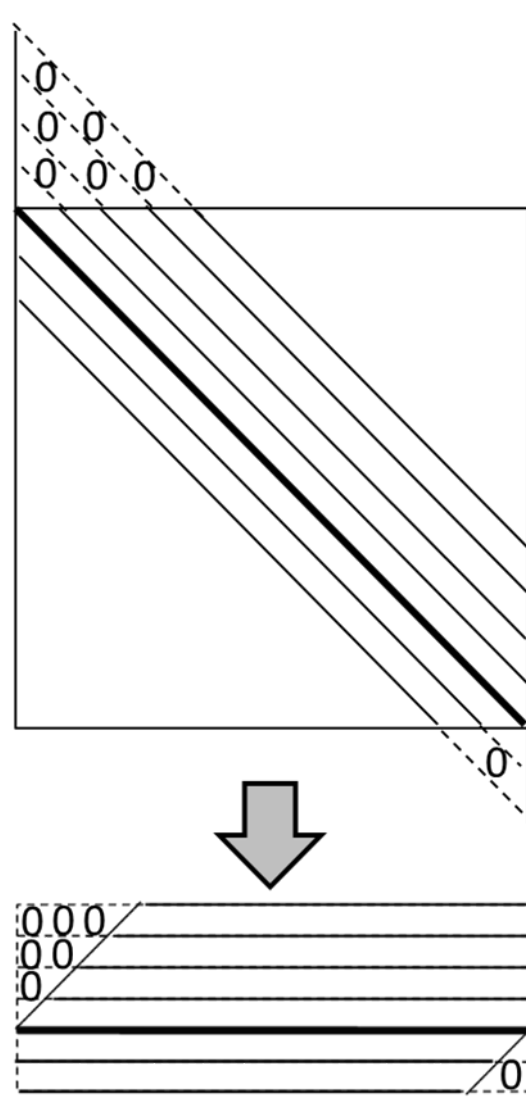
если a_{ij} находятся в пределах упакованного массива, т.е.

если $i < j$ и $(j-i) \leq q$

если $i > j$ и $(i-j) \leq p$

Аналогичным образом можно рассмотреть структуру хранения ленточной матрицы по строкам.

6.3.2. Ленточный столбцовый формат

Общий вид	Пример																																																												
	<p>Исходная матрица</p> <p>$n=6; p=1; q=2; m=4;$</p> <table data-bbox="880 452 1425 990"><tr><td>5</td><td>8</td><td>2</td><td>0</td><td>0</td><td>0</td></tr><tr><td>7</td><td>9</td><td>0</td><td>8</td><td>0</td><td>0</td></tr><tr><td>0</td><td>4</td><td>0</td><td>2</td><td>5</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>3</td><td>9</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>4</td><td>3</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>6</td><td>0</td></tr></table> <p style="text-align: center;">↓</p> <table data-bbox="880 1113 1425 1471"><tr><td>0</td><td>0</td><td>2</td><td>8</td><td>5</td><td>1</td></tr><tr><td>0</td><td>3</td><td>9</td><td>2</td><td>0</td><td>8</td></tr><tr><td>5</td><td>9</td><td>0</td><td>3</td><td>4</td><td>0</td></tr><tr><td>7</td><td>4</td><td>1</td><td>0</td><td>6</td><td>0</td></tr></table> <p>Упакованная матрица</p>	5	8	2	0	0	0	7	9	0	8	0	0	0	4	0	2	5	0	0	0	1	3	9	1	0	0	0	0	4	3	0	0	0	0	6	0	0	0	2	8	5	1	0	3	9	2	0	8	5	9	0	3	4	0	7	4	1	0	6	0
5	8	2	0	0	0																																																								
7	9	0	8	0	0																																																								
0	4	0	2	5	0																																																								
0	0	1	3	9	1																																																								
0	0	0	0	4	3																																																								
0	0	0	0	6	0																																																								
0	0	2	8	5	1																																																								
0	3	9	2	0	8																																																								
5	9	0	3	4	0																																																								
7	4	1	0	6	0																																																								

Элемент исходной матрицы a_{ij} расположен в упакованном массиве $P[i - j + q + 1, j]$,
 если a_{ij} находятся в пределах упакованного массива, т.е.
 если $i < j$ и $(j - i) \leq q$
 если $i > j$ и $(i - j) \leq p$

6.3.3. Ленточный формат матрицы с одинаковыми полуширинами ленты

Рассмотрим частным случаем ленточной матрицы, у которой $\beta = p = q$ – полуширина ленты. В этом случае $a_{ij} = 0$, если $|i - j| > \beta$.

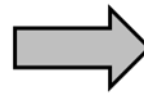
Такая матрица может упаковываться в двумерный массив, имеющий столько же строк, сколько исходный и $(2\beta + 1)$ столбцов. Т.е. это частный случай строчного формата.

Средний столбец ($\beta + 1$) соответствует главной диагонали. Остальные элементы располагаются на том же удалении от диагонального элемента, что и в неупакованной матрице.

Исходная матрица

$n=8$; $\beta = p = q = 2$; $m = 2\beta + 1 = 5$;

5	8	2	0	0	0	0	0
7	9	0	8	0	0	0	0
1	4	0	2	5	0	0	0
0	9	1	3	9	1	0	0
0	0	7	0	4	3	0	0
0	0	0	0	6	0	0	0
0	0	0	0	6	0	4	3
0	0	0	0	0	3	6	0



Упакованная матрица

0	0	5	8	2
0	7	9	0	8
1	4	0	2	5
9	1	3	9	1
7	0	4	3	0
0	6	0	0	0
6	0	4	3	0
3	6	0	0	0

Алгоритм доступа к элементу a_{ij} .

Если $|i - j| > \beta$ то этот элемент равен нулю.

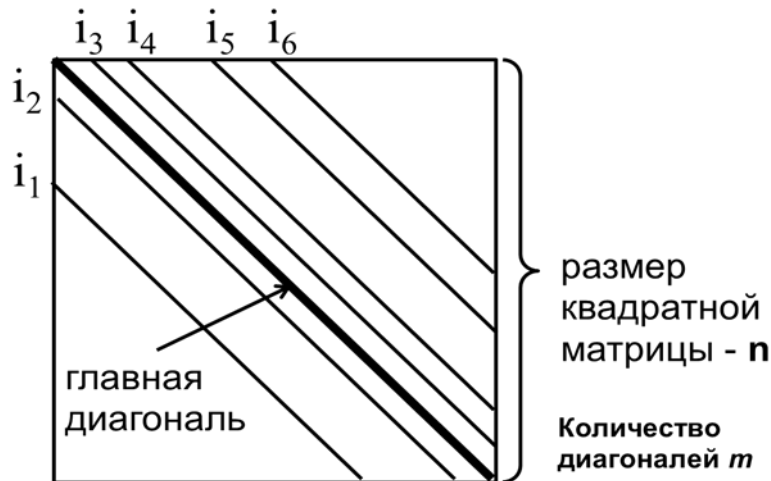
В противном случае берём элемент

$$A[i, j - i + \beta + 1].$$

Дома рассмотреть симметричный столбцовый формат хранения ленточной матрицы.

6.3.4. Диагональный формат хранения (Матрица с плотной диагональю)

Диагональный формат хранения матриц используется, когда все ненулевые элементы матрицы расположены на различных, не плотно расположенных- диагоналях. Хранение таких матриц аналогично хранению ленточных матриц, то есть используется массив размера $n \times m$, где n - размерность исходной матрицы, m - количество ненулевых диагоналей.



Побочные диагонали доопределяются до общей размерности нулями аналогично ленточному формату. При этом дополнительно хранится массив целых чисел **Index** размера m , в котором указывается сдвиг каждой диагонали от главной - положительные индексы для верхнего треугольника, отрицательные для нижнего треугольника. Так, для матрицы, показанной на Рис. выше, массив **Index** будет содержать элементы $(-i_1; -i_2; 0; i_3; i_4; i_5; i_6)$

Пример хранения матрицы в диагональном строчном формате показан на Рис.

Исходная матрица

$n=6; m=4;$

5	8	0	1	0	0
0	9	0	0	2	0
1	0	0	2	0	3
0	9	0	3	9	0
0	0	7	0	4	3
0	0	0	0	0	0



Упакованный вид

0	5	8	1
0	9	0	2
1	0	2	3
9	3	9	0
7	4	3	0
0	0	0	0

Index:

-2	0	1	3
----	---	---	---

Доступ к элементу a_{ij} – если элемент $(j-i)$ – есть в матрице **Index** и он расположен в k -ой позиции, то берем элемент $A[i, k]$ и 0 – в другом случае.

Задания к лабораторной работе «Разреженные матрицы»

Задача 1. Даны две разреженные матрицы общего вида. Перемножить их и результат занести в разреженную матрицу CSR.

Задача 2. Даны две разреженные структурно симметричные матрицы. Перемножить их и результат занести в разреженную матрицу CSS.

Задача 3. Даны две разреженные ленточные матрицы. Перемножить их и результат занести в разреженную матрицу CSR.

Задача 4. Даны две разреженные матрицы общего вида. Сложить их и результат занести в разреженную матрицу CSS.

Задача 5. Даны две разреженные структурно симметричные матрицы. Сложить их и результат занести в разреженную матрицу CSR.

Задача 6. Даны две разреженные ленточные матрицы. Сложить их и результат занести в разреженную матрицу CSS.

Задача 7. Даны две разреженные матрицы общего вида. Из одной матрицы вычесть другую и результат занести в разреженную матрицу CSR.

Задача 8. Даны две разреженные структурно симметричные матрицы. Из одной матрицы вычесть другую и результат занести в разреженную матрицу CSS.

Задача 9. Даны две разреженные ленточные матрицы. Из одной матрицы вычесть другую и результат занести в разреженную матрицу CSR.

Задача 10. Дана разреженная структурно симметричная матрица. Найти её определитель.

Задача 11. Дана разреженная ленточная матрица. Найти её определитель.

Задача 12. Дана разреженная матрица CSR. Найти её определитель.

Задача 13. Дана разреженная матрица общего вида. Найти матрицу, обратную к ней.

Задача 14. Дана разреженная ленточная матрица. Найти матрицу, обратную к ней.

Задача 15. Дана разреженная структурно симметричная матрица. Найти матрицу, обратную к ней.

Задача 16. Дана разреженная матрица CSS. Найти сумму её элементов.

Задача 17. Дана разреженная ленточная матрица. Найти сумму её элементов.

Задача 18. Дана разреженная структурно симметричная матрица. Найти сумму её элементов.

Задача 19. Дана разреженная матрица CSR. Найти количество её различных элементов и вывести их на экран.

Задача 20. Дана разреженная ленточная матрица. Найти количество её различных элементов и вывести их на экран.

Задача 21. Дана разреженная структурно симметричная матрица. Найти количество её различных элементов и вывести их на экран.

Задача 22. Дана разреженная матрица общего вида (CSS или CSR) и число b . Матрица просматривается слева на право, и сверху вниз. На места ненулевых элементов матрицы вначале поместить все её ненулевые элементы большие b , а затем ненулевые элементы меньшие b . Элементы не сортировать.

Задача 23. Дана разреженная ленточная матрица и число b . Матрица просматривается слева на право, и сверху вниз. На места ненулевых элементов матрицы вначале поместить все её ненулевые элементы большие b , а затем ненулевые элементы меньшие b . Элементы не сортировать.

Задача 24. Дана разреженная структурно симметричная матрица и число b . Матрица просматривается слева на право, и сверху вниз. На места ненулевых элементов матрицы вначале поместить все её ненулевые элементы большие b , а затем ненулевые элементы меньшие b . Элементы не сортировать.

Задача 25. Дана разреженная матрица общего вида (CSS или CSR). Найти сумму её элементов a_{ij} , у которых сумма $(i+j)$ является чётной.

Задача 26. Дана разреженная ленточная матрица. Найти сумму её элементов a_{ij} , у которых сумма $(i+j)$ является чётной.

Задача 27. Дана разреженная структурно симметричная матрица. Найти сумму её элементов a_{ij} , у которых сумма $(i+j)$ является чётной.

Задача 28. Дана разреженная матрицы общего вида (CSS или CSR). Переставить столбцы в матрице по возрастанию сумм элементов в этих столбцах.

Задача 29. Дана разреженная матрицы общего вида (CSS или CSR). Переставить строки в матрице по возрастанию сумм элементов в этих строках.

Задача 30. Дана разреженная матрицы общего вида (CSS или CSR). Отобразить элементы относительно диагонали, проходящей с левого верхнего угла к правому нижнему углу.

Задача 31. Дана разреженная структурно симметричная матрица. Зеркальное отображение относительно диагонали, проходящей с левого верхнего угла к правому нижнему углу.

Задача 32. Дана разреженная матрицы общего вида(CSS или CSR). Зеркальное отображение относительно диагонали, проходящей с левого нижнего угла к правому верхнему углу.

Задача 33. Дана разреженная структурно симметричная матрица. Зеркальное отображение относительно диагонали, проходящей с левого нижнего угла к правому верхнему углу.

Задача 34. Дана разреженная матрицы общего вида(CSS или CSR). Осуществить циклический сдвиг в матрице каждого столбца на n разрядов.

Задача 35. Дана разреженная матрицы общего вида(CSS или CSR). Осуществить циклический сдвиг в матрице каждой строки на n разрядов.

Задача 36. Дана разреженная матрицы общего вида(CSS или CSR). Осуществить циклический сдвиг в матрице. Сдвинуть всю матрицу. С первой строки элементы переносятся на вторую. И т.д. С последней на первую.

Задача 37. Дана разреженная матрицы общего вида(CSS или CSR). Осуществить циклический сдвиг в матрице. Сдвинуть всю матрицу. С первого столбца элементы переносятся на второй. И т.д. С последнего на первый.

Задача 38. Дана разреженная матрицы общего вида(CSS или CSR). Циклически сдвинуть все диагонали a, b, c , и т.д..

$$\begin{pmatrix} a & b & c & d \\ d & a & b & c \\ c & d & a & b \\ b & c & d & a \\ a & b & c & d \\ d & a & b & c \end{pmatrix}$$

Задача 39. Дана разреженная ленточная матрица. Циклически сдвинуть все диагонали a, b, c , и т.д..

$$\begin{pmatrix} a & b & c & d \\ d & a & b & c \\ c & d & a & b \\ b & c & d & a \end{pmatrix}$$

Задача 40. Дана разреженная матрицы общего вида(CSS или CSR). Повернуть матрицу на 90 градусов.

Задача 41. Дана разреженная матрицы общего вида(CSS или CSR). Повернуть матрицу на 180 градусов.

Задача 42. Дана разреженная матрицы общего вида(CSS или CSR). Повернуть матрицу на 270 градусов.

Список литературы

1. Джордж А., Лю Дж. Численное решение больших разреженных систем уравнений. Пер. с англ. - М:Мир, 1984, 336 с.
2. О. Эстербрю, З. Златаев. Прямые методы для разреженных матриц. Пер. с англ.- М.: Мир, 1987. 120 с
3. Серджио Писсанецки. Технология разреженных матриц. Пер. с англ. — М.: Мир, 1988. —410 с.
4. Баталов Б. В., Егоров Ю. Б., Русаков С. Г. Основы математического моделирования больших интегральных схем на ЭВМ. - М.: Радио и связь, 1982. - 168 с.
5. Р. Тьюарсон. Разреженные матрицы. М.: Мир, 1977. – 192 с.

СОДЕРЖАНИЕ

Разреженные матрицы	3
Хранение разреженной матрицы общего вида.	3
<i>Координатный формат хранения</i>	<i>3</i>
<i>Разреженный строчный формат</i>	<i>5</i>
<i>Разреженный столбцовый формат</i>	<i>7</i>
6.1. Упаковка структурно симметричных матриц.....	8
6.2. Использование связных списков.	9
6.3. Ленточные матрицы	10
6.3.1. <i>Ленточный строчный формат</i>	<i>10</i>
6.3.2. <i>Ленточный столбцовый формат</i>	<i>12</i>
6.3.3. <i>Ленточный формат матрицы с одинаковыми полуширинами ленты</i>	<i>13</i>
6.3.4. <i>Диагональный формат хранения</i>	<i>14</i>
Задания к лабораторной работе «Разреженные матрицы»	15
Список литературы	18