# Работа с разреженными матрицами

Алексей Сальников

2021

## 1. Введение

Разреженные матрицы — матрицы, где большая часть элементов имеет нулевые значения. Несмотря на то, что матричные операции универсальны, для работы с разреженными матрицами требуются форматы хранения данных отличные от форматов для плотных матриц, и как следствие требуются другие алгоритмы для реализации матричных операций. В большей степени это скорее работа со списками, деревьями, хэш-функциями в противовес работе с двумерным массивом.

В рамках задания требуется написать написать несколько программ:

1. **Генератор матриц** –*matrix_ generator*. Матрицы и векторы записываются в текстовые файлы в определённом формате: описание формата представлено далее в тексте. Генератор должн уметь генерировать матрицы удовлетворяющие нужным свойствам: например степени разреженности матрицы.

2. **Конвертеры** – *sparse2dense*, *dense2sparse*. Преобразуют файлы с матрицами из разреженного представления в плотное, и наоборот. При преобразовании из плотного, в разреженное указывается погрешность относительноно 0, $\varepsilon$. Все элементы матрицы $|a_{i,j}| < \varepsilon$ считаются нулевыми и в разреженную матрицу как значения не попадают.

3. **Умножитель** – *multiplyer*. Программа осуществляющая операцию умножения над несколькими матрицами.

4. **Построитель индекса** – *indexer*. Индекс – это внешняя по отношению к самой матрице стуктура данных, которая предназначена для ускорения доступа к элементам матрицы. На вход построителю даётся файл с матрицей, на выходе получаем файл с представлением индекса.

5. **Отображатель индекса** – *index_ draw*. Программа, которая распечатывает индекс в понятном человеку виде[1]

## 2. Организация программного кода

Желательно функции для работы с матрицами вынести в отдельный PASCAL модуль, а функции по работе с индексами (деревьями) в свой модуль. Базовую информацию по работе с модулями можно мосмотреть здесь [1] и здесь [2].

Различные программы, присутствующие в задании могут пользоваться функциями из модулей, это сократит общий размер программного кода и облегчит отладку.

В своём внутреннем представлении матрица должна храниться как сбалансированное бинарное дерево поиска. При этом ключём (заменой числа, с которым происходят сравнения) в дереве

[1]Поскольку индексами являются деревья, то по сути в этом месте предполагается печать дерева.

должна служить пара координат $(i, j)$. Соответственно для любых $i_1, i_2$ и $j_1, j_2$ необходимо определить операцию меньше либо равно, которая даст однозначный ответ $(i_1, j_1) \leq (i_2, j_2)$. Можно в этом месте определить приориетет строки над столбцом (аналог лексикографического порядка в строке). Далее приведён пример элемента дерева:

```
type tree_node_t = record
    internal_node_number :longint;
    row, column :longint;
    element      :double;
    parent       :^tree_node;
    left, right :^tree_node;
end;
```

На время операции перемножения, результирующая матрица не обязательно сбалансированное дерево. Но по завершению перемножения необходимо, чтобы дерево было сбалансированным.

Должны присутствовать функции, которые читают матрицу из файла и записывают в файл. Должны быть функции печатающие матрицу в поток вывода в 2-х форматах: как разреженная матрица, в том виде, как она хранится в файле, и как плотная матрица (в этом представлении все нули печатается). Тоже самое должно работать и для деревьев, образующих индекс.

## 2.1. Генератор – matrix_generator

На вход генератору, в аргументах программы, передаются следующие параметры.

1. Имя файла, где будет сохранена создаваемая матрица.
2. Размерность матрицы: число строк в матрице: *num_rows*, число столбцов в матрице: *num_columns*.
3. Режим генерации матрицы *mode*.
4. Степень разреженности матрицы *matrix_density*, число с плавающей точкой не превосходящее единицы.
5. необязательный параметр print – означает, что созданную матрицу необходимо распечатать в стандартный поток вывода в формате плотной матрицы.

Степень разреженности работает для строки матрицы – это число в интервале $(0, 1]$, означает долю не нулевых элементов по отношению к нулевым. Элементы должны быть равномерно распределены в строке матрицы. Значение параметра программы *matrix_density=1* — означает генерацию плотной матрицы.

Должны быть предусмотрены следующие режимы генерации матрицы:

1. **all_one** — заполненную единицами,
2. **random_low** — со случайными значениями по модулю меньшими 1,
3. **random_high** — со случайными значениями в отрезке $[-1000, 0) \cup (0, 1000]$ и с добавленной дробной частью от параметра *random_low*.
4. **random_integers** – со случайными значениями в отрезке $[-1000, 0) \cup (0, 1000]$ без добавления дробной части.
5. **one** — единичную.

Файл создаваемой генератором матрицы должен иметь расширение **.mtr**

## 2.2. Умножитель – multiplyer

Умножитель умножает матрицы и записывает результат умножения в файл в формате разреженной матрицы.

В аргументах программы умножителю в качестве первого параметра передаётся имя матрицы, где будет сохранён результат умножения. Далее значение $\varepsilon$ которое задаёт, какие элементы будут считаться нулевыми и не попадут в результирующую матрицу. Далее идут подряд имена матриц, которые необходимо перемножать. Например для таких аргументов:

```
./multiplyer Res 0.25 A B C D
```

Должна быть реализована следующая операция: $Res = A*B*C*D$, при этом в результат попадут значения по модулю большие $1/4$.

Здесь имена матриц, это часть имени файла без расширения **.mtr**. Тоесть для $A$ в параметрах программы должно соответствовать 2 файла в файловой системе: $A.mtr$ – файл с матрицей и $A.dot$ – файл с индексом. Если файл с индексом отсутствует, то необходимо создать его в памяти программы.

Для результирующей матрицы необходимо всегда создавать файл с индексом.

Разумеется необходимо сообщать обо всех ошибочных ситуациях в процессе перемножения матриц и файлы с результатом создавать/перезаписывать необходимо только если вся цепочка умножений была произведена успешно.

## 2.3. Построитель индекса – indexer

Данная программа в своих аргументах получает имя матрицы, и тип индекса. затем перестраивает индекс, или строит его по новый, если индекс отсутствует.

Допустимы следующие типы индекса:

1. **red_black** – индекс строится как красно-чёрное дерево.
2. **avl** – индекс строится как АВЛ-дерево.
3. **fobonachi** – индекс строится как дерево Фибоначи.

## 2.4. Отображатель индекса

На вход принимает файл с индексом, и режим печати. На выходе печатает дерево, которое задаёт индекс. Режимы:

1. **root-left-right** – дерево печатается так: Корень, затем левое поддерево, затем правое поддерево.
2. **left-root-right** – по аналоги с предыдущем левое поддерево, корнень, правое
3. **right-root-left** – правое, конень, левое.
4. **levels** – дерево печатается по уровням: от корня к листьям. уровни разделяются пустыми строками.
5. **height** – печатается только высота дерева.

Для режмов root-left-right и levels – каждый элемент дерева распечатывается в следующем формате на своей строчке:

```
{150: (2,3) 33.5 152 NULL}
```

Здесь 150 – номер вершины, (2,3) – координаты в матрице, 152 номер вершины левого поддерева. Если у дерева есть поддерево, должен печататься номер вершины поддерева, либо *NULL* если поддерево отсутствует как в данном примере.

# 3. Конвертеры

Конвертеры как параметры программы принимают два имени матрицы и в качестве треьего возможного парметра значение $\varepsilon$. расширение к имени файла дописывается автоматически.

# 4. Форматы файлов

## 4.1. Разреженные матрицы

Формат файла для хранения разреженных матриц – текстовый. В файле могут встречаться комментарии. Комментарий начинается символом *'#'*.

Файл для хранения матрицы начинается со слова *sparse_matrix* далее пробельные символы, далее число строк в матрице, пробельные символы и число столбцов в матрице.

Затем следуют координаты и числа. Координаты и числа отделяются друг от друга пробельными символами. Каждое число с координатами на своей строке. Сперва в строке идёт координата номер строки в матрице, потом номер столбца, далее, само значение в матрице. В файле могут быть пустые строки и строки состоящие целиком из комментариев. Координаты нумеруются с единицы.

Пример файла с матрицей:

```
#
# This file describes
# sparse matrix
#
sparse_matrix 50000 5000

1       1    100.0
6000    2    0.85
7       1    -3.4
22      2    -12345678.000000000005789 # very long
                                       # number
```

## 4.2. Плотные матрицы

Формат файла для хранения разреженных матриц – текстовый. В файле могут встречаться комментарии. Комментарий начинается символом *'#'*.

Файл для хранения матрицы начинается со слова *dense_matrix* далее пробельные символы, далее число строк в матрице, пробельные символы и число столбцов в матрице.

Далее на каждой строчке через пробельные символы, в том числе переводы строки сами значения элементов матрицы.

```
#
# This file describes
# dense matrix
#
dense_matrix 3 3
0.12   1.25     50.2345678
0      0        250
123    12.44    55.0
```

## 4.3. Индекс

Файл с индексами задаётся в формате ориентированного графа graphviz [3]. В программе может встретиться только некоторое подмножество конструкций допустимых в graphviz.

Файл начинается со слова `digraph`. Далее в фигурных скобках идёт описание самого дерева. Открывающая и закрывающая скобки, должны стоять, каждая на своей строке.

Внутри сперва идёт описание вершин. Вершина сперва задаётся своим номером, далее, после пробельных символов следует открывающая квадратная скобка, затем `label="` далее через символы пробел идёт подряд координата i, координата j в матрице, затем через
n значение находящееся в матрице по этим координатам. Конец задаётся последовательностью символов `"];`. С новой строки в файле следующий узел дерева.

Далее, перед указанием связей между узлами дерева, следует обязательный коментарий. `//edges`. Связи задаются следующим образом.

`номер_вершины -> номер_левого [label="L"]; номер_вершины -> номер_правого [label="R"];`

Файлу в данном формате можно дать расшинение **.dot** и подать на вход утилите **dot**, входящей в пакет graphviz. В результате получим отрисованное дерево.

Пример команды:

```
dot -Tpdf -o result.pdf tree.dot
```

### 4.4. Пример файла с индексом

```
digraph
{
    1 [label="12  44\n0.19"];
    2 [label="1    1\n25.4"];
    3 [label="300  2\n444.6"];
    4 [label="34   12\n55.0"];

    //edges


     4 ->  1 [label="L"]; 4 -> 2 [label="R"];
                          2 -> 3 [label="R"];


}
```

## Список литературы


[1] `https://life-prog.ru/view_algoritmleng.php?id=117`

[2] `http://www.pascal.helpov.net/index/pascal_modules_programming`

[3] `https://ru.wikipedia.org/wiki/DOT_(%D1%8F%D0%B7%D1%8B%D0%BA)`