Task 2: From Sparse Matrix to Normal Matrix (Haskell - 8 Points)

May 26, 2022

Introduction

This task is dual to Task 1, where you convert a sparse matrix into a list of triplets consisting of non-zero elements and their positions. In this task, you need to do an oppositive conversion. More precisely, implement a Haskell program that first reads from the standard input a size of the matrix and a list of triplets `(row,col,val)` and then converts this representation into the list of rows `type Matrix = [[Int]]`. Finally, it prints the resulting matrix to the screen, e.g., using `print`.

Implementation

Represent the triplets `(row,col,val)` as values of the type `type Triplet = (Int, Int, Int)`. Implement a function `main :: IO ()` that works as follows:

1. It displays a message `Enter number of rows:`,
2. then let the user enter a positive integer m,
3. next it displays a message `Enter number of columns:`,
4. then let the user enter a positive integer n,
5. finally it displays a message `Enter triplets separated by spaces:`,
6. then let the user enter a list of space-separated values of type `Triplet`,
7. then it converts this list into a $m \times n$-matrix of type `Matrix`,
8. and displays the resulting matrix by `print`.

You may assume only valid inputs.

Example 1

```
> main
Enter number of rows:
1
Enter number of columns:
5
Enter triplets separated by spaces:
(0,0,7) (0,3,9)
[[7,0,0,9,0]]
```

Example 2

```
> main
Enter number of rows:
3
Enter number of columns:
3
Enter triplets separated by spaces:
(0,0,1) (1,1,2) (2,2,3)
[[1,0,0],[0,2,0],[0,0,3]]
```

Hint

The triplets can be parsed directly by the function `read`, e.g.

```
> read "(1,2,3)" :: (Int,Int,Int)
(1,2,3)
```

To split the triplets separated by spaces, use the function `words`, e.g.

```
> words "(1,2,3) (4,5,6)"
["(1,2,3)","(4,5,6)"]
```