How to efficiently select only certain columns of a triangular matrix

I have the following problem:

I need certain columns of a huge triangular 1-0 matrix.

E.g.

```
Matrix =

1 0 0 0
1 1 0 0
1 1 1 0
1 1 1 1

Index =

[1 4]

Result =

1 0
1 0
1 0
1 1
```

I figured the easiest way would be:

```
index = [10 20 300] %arbitrary index
buf = tril(ones(60000,60000))
matr = buf(:,index)
```

However, this does not work as the buffer matrix is too large and leads to MATLAB throwing an error. Thus, this approach is blocked.

How can I solve that problem efficiently? (E.g. it would be trivial by just looping over the index array and concatenating self-made rows, however this would be slow and I was hoping for a faster approach)

The index array will not be larger than 1/10th of the available columns.

matlab matrix

edited 10 hours ago

Adriaan
8,666 5 20 51

asked 10 hours ago

Sim
2,073 1 20 50

So the problem is in creating `buf` or in selecting columns from it? – Luis Mendo 10 hours ago

@LuisMendo creating `buf` is the breaking point. However, not necessary the problem as I am hoping that my approach can be achieved with easier means. How the problem is solved is not really an issue as long as it is effective (e.g. looping over the index array and creating the matrix by concatenation is only a last resort) – Sim 10 hours ago

Then I don't get what you want. Can you explain it better? Possible X-Y problem? – Luis Mendo 10 hours ago

@LuisMendo I extended on the explanation of my problem. I do not really care how it is solved as long as it is fairly fast. The only relevant thing for me is having the needed matrix at the end. – Sim 10 hours ago

1 The obvious answer is to only generate the columns that you need. What isn't obvious is how to do that because you've told us nothing about the matrix itself. – beaker 10 hours ago

1 Answer

If the matrix contains ones on the main diagonal and below, and zeros otherwise, you can do it as follows without actually generating the matrix:

```
N = 10; % number of rows of (implicit) matrix
Index = [1 4]; % column indices
Result = bsxfun(@ge, (1:N).', Index);
```

answered 9 hours ago

Luis Mendo
76k 8 39 89

shouldn't it be `number of columns?` in the comment – Sim 9 hours ago

No, it's rows. The number of columns is implicitly given by `Index` ... if I'm interpreting your question correctly – Luis Mendo 9 hours ago

ah yes, completly my error. I mistook rows for columns - exactly what I wanted and needed. – Sim 9 hours ago