Additional remarks

IN3200/IN4200 Partial Exam, Spring 2019

This short note contains some additional remarks about the text of the partial exam:

`https://www.uio.no/studier/emner/matnat/ifi/IN3200/v19/teaching-material/in3200_in4200_partial_exam_v19.pdf`

About the file format of a web graph

- It can be assumed that each web link is uniquely listed, that is, there will NOT be multiple text lines describing the same web link.
- You may NOT assume that the links are sorted with respect to `FromNodeId`.
- For each `FromNodeId`, you may NOT assume that the links are sorted with respect to `ToNodeId`.

About the *CRS (Compressed Row Storage)* scheme

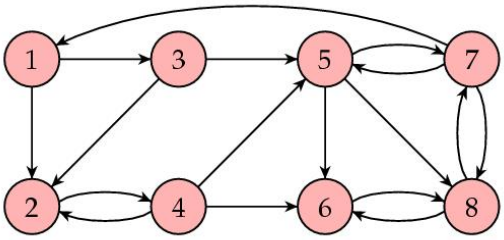

Figure 1: A web graph containing eight webpages that are linked together.

One important remark: The brief description of the CRS scheme given in Section 3.6.1 of the textbook has a"flaw". Specifically, the array `row_ptr` should, for convenience, be of length one plus the total number of webpages. For example, the hyperlink matrix corresponding to the web graph shown in Figure 1 should be represented by the following three arrays in the CRS format (remember that all indices start from 0 in the C convention):

```
row_ptr=[0,1,4,5,6,9,12,14,17]
col_idx=[6,0,2,3,0,1,2,3,6,3,4,7,4,7,4,5,6]
val=[0.333333,0.500000,0.500000,0.333333,0.500000,
1.000000,0.500000,0.333333,0.333333,0.333333,0.333333,
0.500000,0.333333,0.500000,0.333333,1.000000,0.333333]
```

Hint: The most efficient strategy (at least with respect to temporary memory usage) for building up a CRS-formatted hyperlink matrix is to read the web graph file *twice*. The first pass is only used to "count" the various information, such as the info needed to set up the array `row_ptr` (in particular, how many inbound links each webpage has) and the number of links excluding all self-links (if any). It is in the second pass that the values in arrays `col_idx` and `val` are filled.

Example result of the PageRank iterations

For the 8-webpage example shown in Figure 1, running the PageRank algorithm using $d = 1$ (no damping) will produce the following results:

```
---Initial guess:
scores=[0.125000,0.125000,0.125000,0.125000,0.125000,0.125000,0.125000,0.125000]
---After iteration 1:
scores=[0.041667,0.166667,0.062500,0.125000,0.145833,0.145833,0.104167,0.208333]
---After iteration 2:
scores=[0.034722,0.093750,0.020833,0.166667,0.107639,0.194444,0.152778,0.229167]
---After iteration 3:
scores=[0.050926,0.083333,0.017361,0.093750,0.116898,0.206019,0.150463,0.281250]
---After iteration 4:
scores=[0.050154,0.065394,0.025463,0.083333,0.090085,0.210841,0.179591,0.295139]
---After iteration 5:
scores=[0.059864,0.065586,0.025077,0.065394,0.100373,0.205376,0.177598,0.300733]
....
```

When converged, the PageRank scores will be as follows:

```
scores=[0.060000,0.067500,0.030000,0.067500,0.097500,0.202500,0.180000,0.295000]
```

Note: For real-world web graphs having many webpages (when N is very large), it is not recommended to display the intermediate or converged score values by `printf`!