

Homework #10

Graphs III

Anastasiia Konoplina

1. Create an adjacency matrix representation G of this graph. Calculate $G \cdot G$. Draw the respective result.

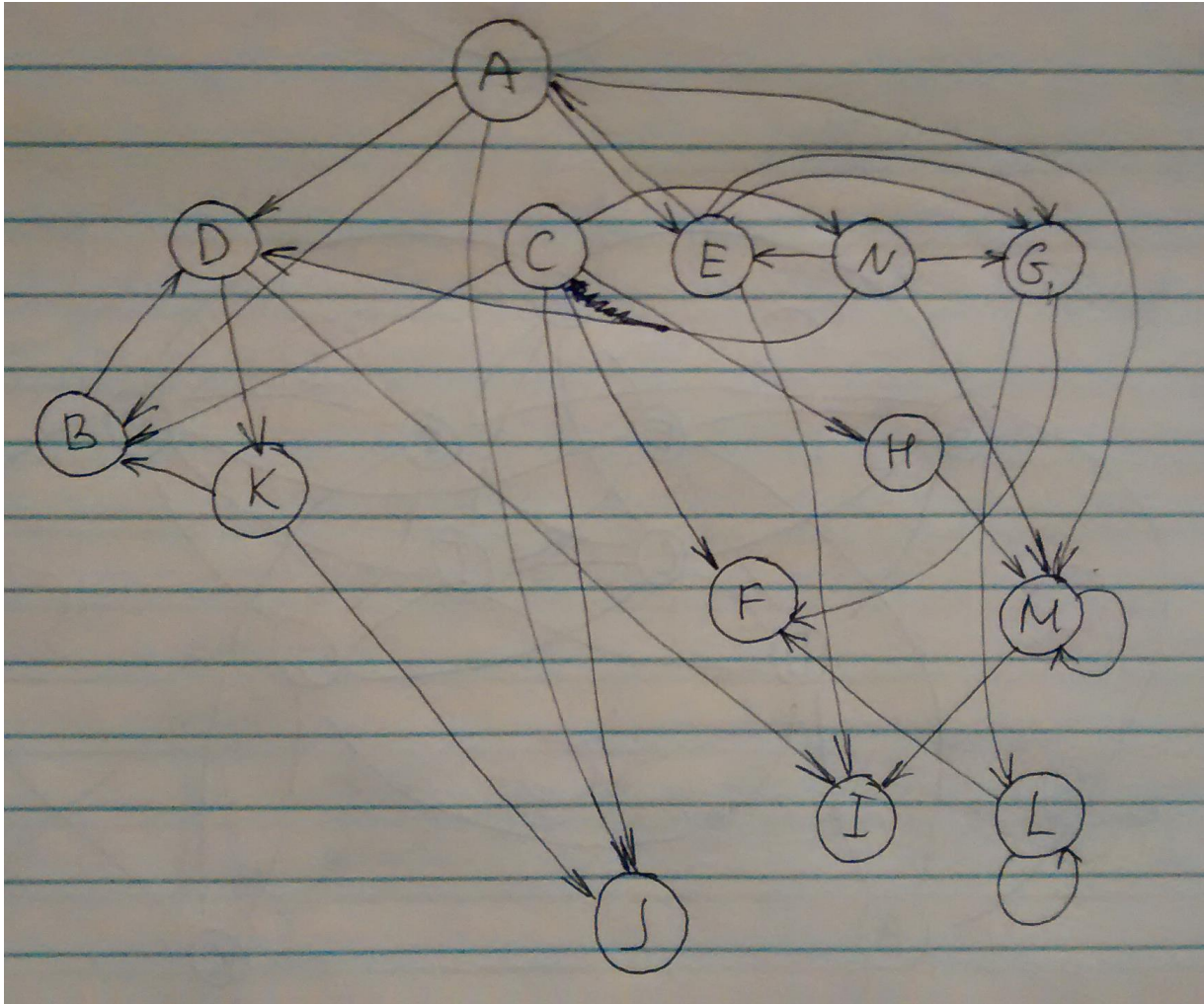
Adjacency matrix:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
A	0	0	1	1	0	0	1	0	0	0	0	0	0	0
B	0	0	0	0	0	0	0	0	0	0	1	0	0	0
C	0	0	0	1	1	0	0	0	0	0	0	0	0	0
D	0	1	0	0	0	0	0	0	0	1	0	0	0	0
E	0	0	0	0	0	1	0	1	0	0	0	0	0	1
F	0	0	0	0	0	0	0	0	1	0	0	0	0	0
G	0	0	0	0	0	0	0	0	0	0	0	0	1	0
H	0	0	0	0	0	0	1	0	0	0	0	0	0	0
I	0	0	0	0	0	0	0	0	0	0	0	0	0	0
J	0	0	0	0	0	0	0	0	1	0	0	0	0	0
K	0	0	0	1	0	0	0	0	0	0	0	0	0	0
L	0	0	0	0	0	0	0	0	0	0	0	0	1	0
M	0	0	0	0	0	1	0	0	0	0	0	1	0	0
N	1	0	0	0	0	0	1	0	0	0	0	0	0	0

$G \cdot G$ (I put zeros as diagonal elements):

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
A	0	1	0	1	1	0	0	0	0	1	0	0	1	0
B	0	0	0	1	0	0	0	0	0	0	0	0	0	0
C	0	1	0	0	0	1	0	1	0	1	0	0	0	1
D	0	0	0	0	0	0	0	0	1	0	1	0	0	0
E	1	0	0	0	0	0	2	0	1	0	0	0	0	0
F	0	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	0	0	0	0	1	0	0	0	0	0	1	0	0
H	0	0	0	0	0	0	0	0	0	0	0	0	1	0
I	0	0	0	0	0	0	0	0	0	0	0	0	0	0
J	0	0	0	0	0	0	0	0	0	0	0	0	0	0
K	0	1	0	0	0	0	0	0	0	1	0	0	0	0
L	0	0	0	0	0	1	0	0	0	0	0	1	0	0
M	0	0	0	0	0	0	0	0	1	0	0	0	1	0
N	0	0	1	1	0	0	1	0	0	0	0	0	1	0

Respective result:



Excel document with matrices are [here](#). I am not sure that 1 on diagonal should be considered as loops (most likely no, but I draw them) as well as other numbers than 1 in matrix (I had 2 for edge E-G I draw 2 edges, but it seems like in this task it doesn't matter what number, it is still only 1 edge between nodes). So in this case $G \cdot G$ shows paths of length 2 of G .

2. Calculate the transitive closure of the G using the methods a) $G \cdot G \cdot G \dots \cdot G$, b) $G_2 = G \cdot G$, $G_4 = G_2 \cdot G_2$, ... and b) the Warshall algorithms. Verify that the methods produce the same end result.

In this task I will put "1" in diagonal since we need not exactly n paths but up to n paths.

Excel file with initial matrix is [here](#).

a) Calculations for both a and b available [here](#) (MathCad). Result is [here](#) tab "a":

4096	21845	4032	24512	4096	34914	12222	4160	47942	21845	19179	30754	36800	4160	A	A	B	C	D	E	F	G	H	I	J	K	L	M	N
0	5461	0	5461	0	0	0	0	5448	5460	5462	0	0	0	B	1	1	1	1	1	1	1	1	1	1	1	1	1	1
4160	19050	4096	21846	4032	32705	12415	4096	43120	19050	16384	28609	34784	4096	C	0	1	0	1	0	0	0	0	1	1	1	0	0	0
0	5462	0	5461	0	0	0	0	5461	5462	5461	0	0	0	D	1	1	1	1	1	1	1	1	1	1	1	1	1	1
4096	16384	4160	19179	4096	38816	12352	4032	46294	16384	13589	34784	41024	4032	E	0	1	0	1	0	0	0	0	1	1	1	0	0	0
0	0	0	0	0	1	0	0	14	0	0	0	0	0	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	8191	1	0	8178	0	0	8191	8192	0	G	0	0	0	0	0	1	0	0	1	0	0	0	0	0
0	0	0	0	0	8178	14	1	8100	0	0	8178	8191	0	H	0	0	0	0	0	1	1	0	1	0	0	1	1	0
0	0	0	0	0	0	0	0	8178	1	0	0	0	0	I	0	0	0	0	0	1	1	1	1	0	0	1	1	0
0	0	0	0	0	0	0	0	14	1	0	0	0	0	J	0	0	0	0	0	0	0	0	1	1	0	0	0	0
0	5461	0	5462	0	0	0	0	5460	5461	5461	0	0	0	K	0	1	0	1	0	0	0	0	1	1	1	0	0	0
0	0	0	0	0	8191	0	0	8178	0	0	8192	8192	0	L	0	0	0	0	0	1	0	0	1	0	0	1	1	0
0	0	0	0	0	8192	0	0	8191	0	0	8192	8192	0	M	0	0	0	0	0	1	0	0	1	0	0	1	1	0
4032	19179	4096	21845	4160	36941	12146	4095	47086	19179	16384	32846	38945	4096	N	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Result of calculations

Put 1 if not 0

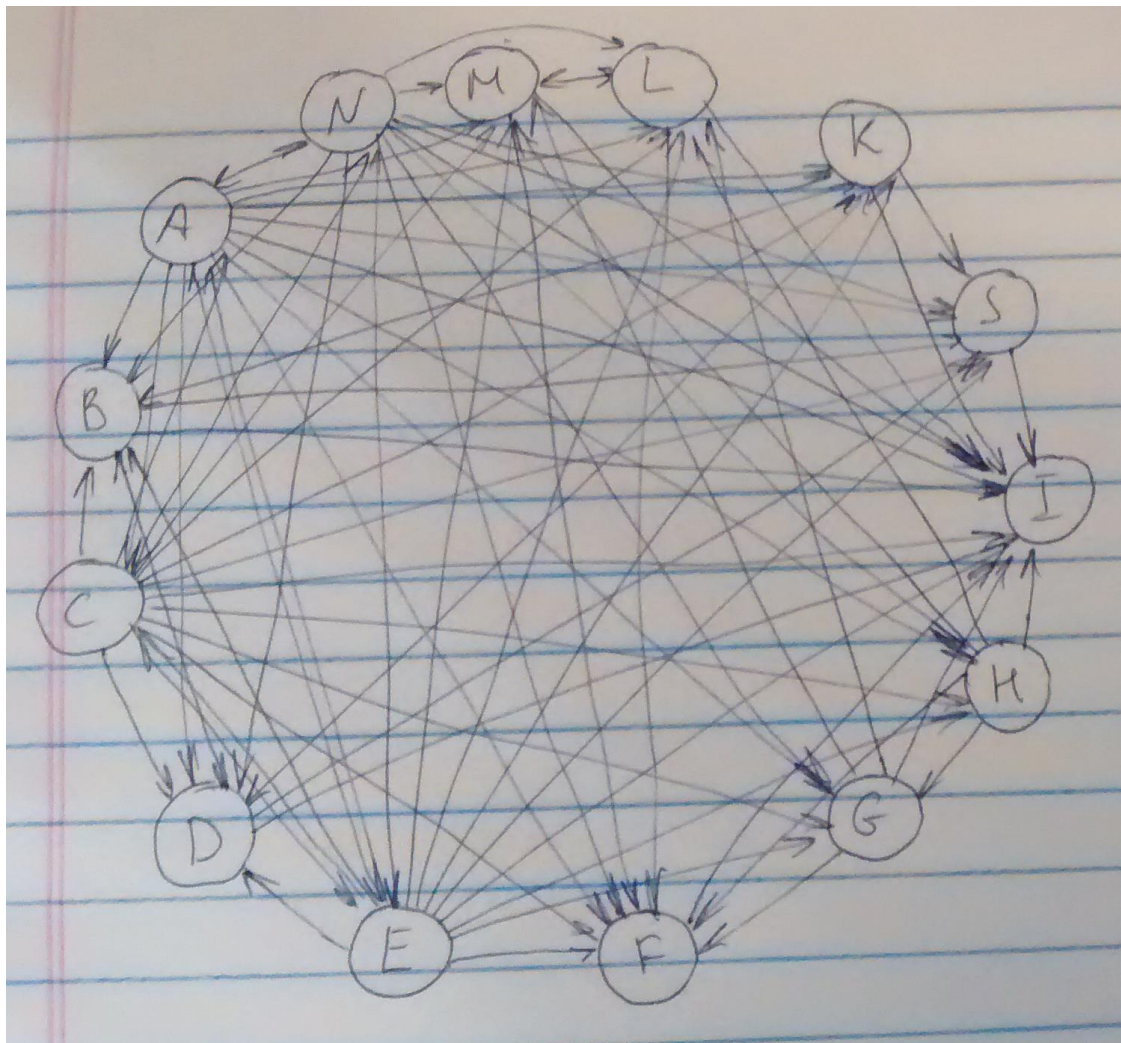
- b) Since we need to multiply previous result iteratively and there are 14 nodes in graph, basically I calculated G^{16} . Calculations for both a and b available [here](#) (MathCad). Result is here tab "b":

16512	98176	16384	109227	16256	163714	49406	16384	226868	98176	87381	147330	171968	16384	A	A	B	C	D	E	F	G	H	I	J	K	L	M	N
0	21845	0	21846	0	0	0	0	21829	21844	21845	0	0	0	B	1	1	1	1	1	1	1	1	1	1	1	1	1	1
16384	87382	16512	98432	16384	155457	49279	16256	207292	87382	76330	139201	163968	16256	C	0	1	0	1	0	0	0	0	1	1	1	0	0	0
0	21845	0	21845	0	0	0	0	21846	21845	21846	0	0	0	D	1	1	1	1	1	1	1	1	1	1	1	1	1	1
16256	76587	16384	87381	16512	180352	48896	16384	220993	76587	65536	163968	188480	16384	E	0	1	0	1	0	0	0	0	1	1	1	0	0	0
0	0	0	0	0	1	0	0	16	0	0	0	0	0	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	32767	1	0	32752	0	0	32767	32768	0	G	0	0	0	0	0	1	0	0	1	0	0	0	0	0
0	0	0	0	0	32752	16	1	32647	0	0	32752	32767	0	H	0	0	0	0	0	1	1	1	1	0	0	1	1	0
0	0	0	0	0	0	0	0	1	0	0	0	0	0	I	0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	16	1	0	0	0	0	J	0	0	0	0	0	0	0	0	1	1	0	0	0	0
0	21846	0	21845	0	0	0	0	21844	21846	21845	0	0	0	K	0	1	0	1	0	0	0	0	1	1	1	0	0	0
0	0	0	0	0	32767	0	0	32752	0	0	32768	32768	0	L	0	0	0	0	0	1	0	0	1	0	0	1	1	0
0	0	0	0	0	32768	0	0	32767	0	0	32768	32768	0	M	0	0	0	0	0	1	0	0	1	0	0	1	1	0
16384	87381	16256	98176	16384	172239	49008	16511	224276	87381	76587	155728	180097	16512	N	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Result of calculations

Put 1 if not 0

- c) Code is [here](#). Results are the same for all 3 algorithms. Transitive closure is below:



3. Simulate the random walk (Markov Chain) procedure starting from A. When you have 2 edges, use 50:50 probability, when 3 then 1/3 probability, and so on. Count the nr of times you visit each node. You may observe getting stuck. Add a link from "dead end" I to A. Now repeat.

The code is [here](#).

Since probability is equal for all nodes, I didn't use any additional conditions, just selected node randomly.

Number of times every node was visited:

{'A': 11, 'M': 10, 'D': 9, 'I': 9, 'L': 6, 'C': 5, 'B': 5, 'F': 5, 'K': 5, 'G': 4, 'J': 4, 'E': 3, 'H': 1, 'N':

1}

4. Replicate the result of 3 by adjacency matrix representation and using matrix multiplications (with and without $I \rightarrow A$). Compare speed, how fast can you outperform random walk by matrix operations?

5. Study the page rank procedure without the $I \rightarrow A$ link (you can use a simplification using random walk) - allow in 20% of cases re-appear in any of the nodes of the graph (with equal probability). List the nodes with highest "page rank" (most often visited nodes, what proportion of time would be spent in each node).

I used "Dynamical system point of view" approach from [here](#). Highest "page rank" has nodes:

- M (17%)
- D (14%)
- F and L (10% each)
- B and J (9%)

Calculations are done in MathCad and available [here](#).

6. Bonus (1p) Add 5 new nodes into the graph - two ones pointing only to A, one that A points into, and 2 without any edges in or out. What would be their respective page rank probabilities?

2 without any edges and 2 which point only to A has zeros – last in a rank. And one that A points to – is higher in the rank than A, C, E and is equal to $1.142 \cdot 10^{-4}$. MathCad file with calculations is [here](#).