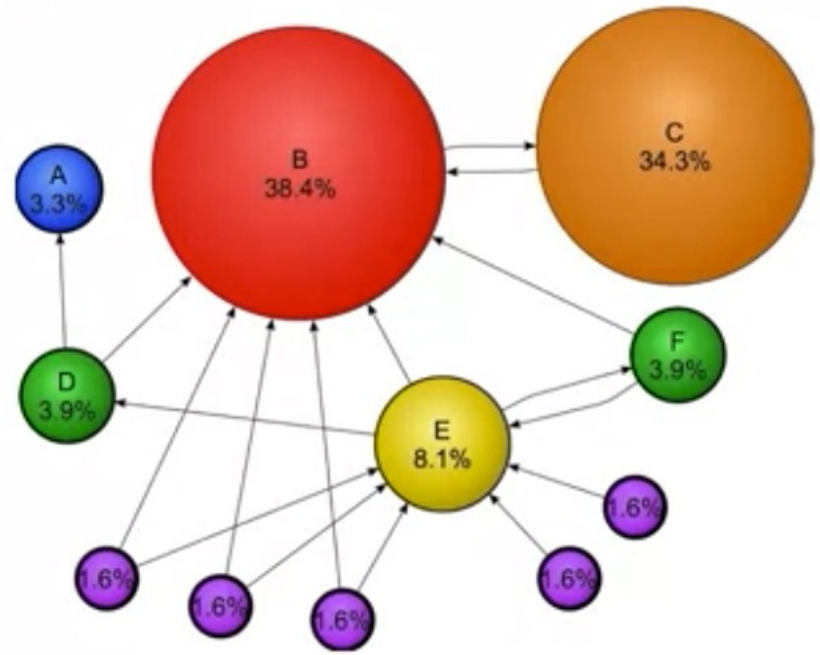
PageRank

Developed by Google founders to measure the importance of webpages from the hyperlink network structure.

PageRank assigns a score of importance to each node. Important nodes are those with many in-links from important pages.

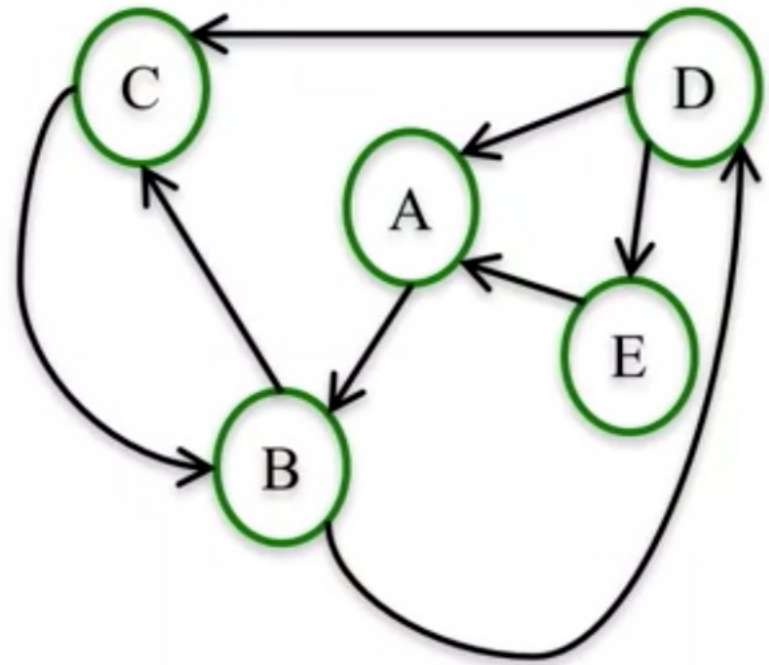
PageRank can be used for any type of network, but it is mainly useful for directed networks.



A node's PageRank depends on the PageRank of other nodes (Circular definition?).

PageRank

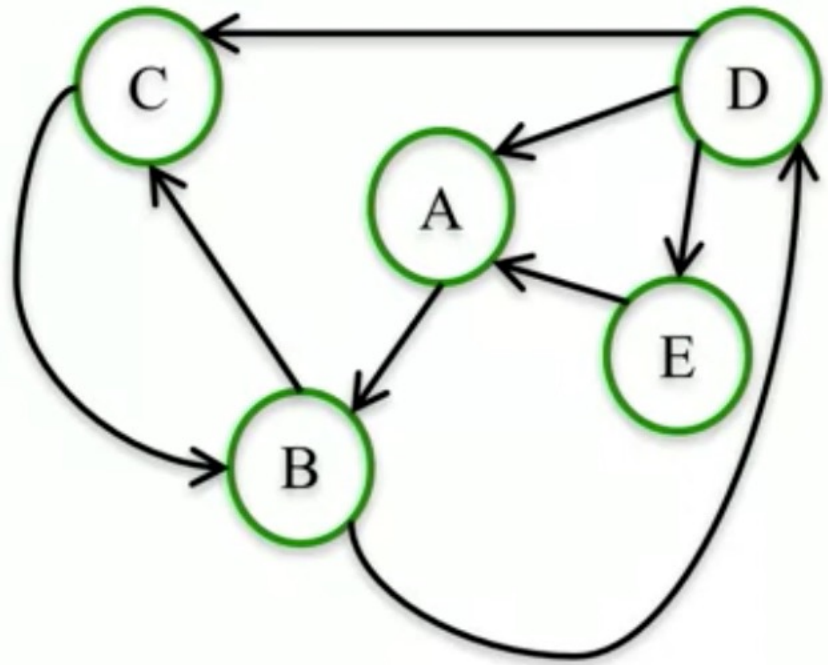
Who should be the most "important" node in this network?



PageRank – Step 1

Page Rank (k = 1)					
	A	B	C	D	E
Old	1/5	1/5	1/5	1/5	1/5
New	4/15	2/5	1/6	1/10	1/15

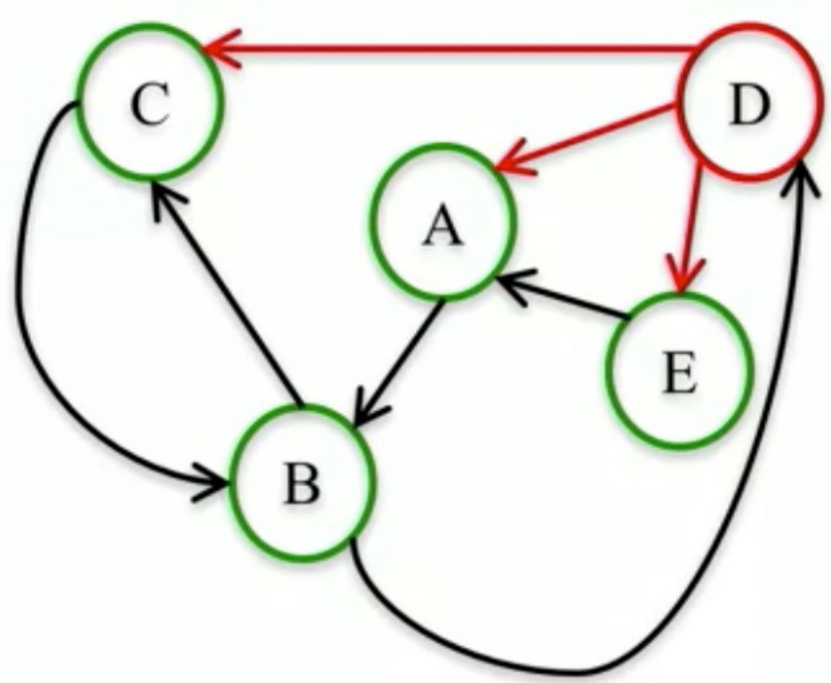
A: (1/3)*(1/5) + 1/5 = 4/15
B: 1/5 + 1/5 = 2/5
C: (1/3)*(1/5) + (1/2)*(1/5) = 5/30 = 1/6
D: (1/2)*(1/5) = 1/10
E: (1/3)*(1/5) = 1/15



PageRank – Step 2

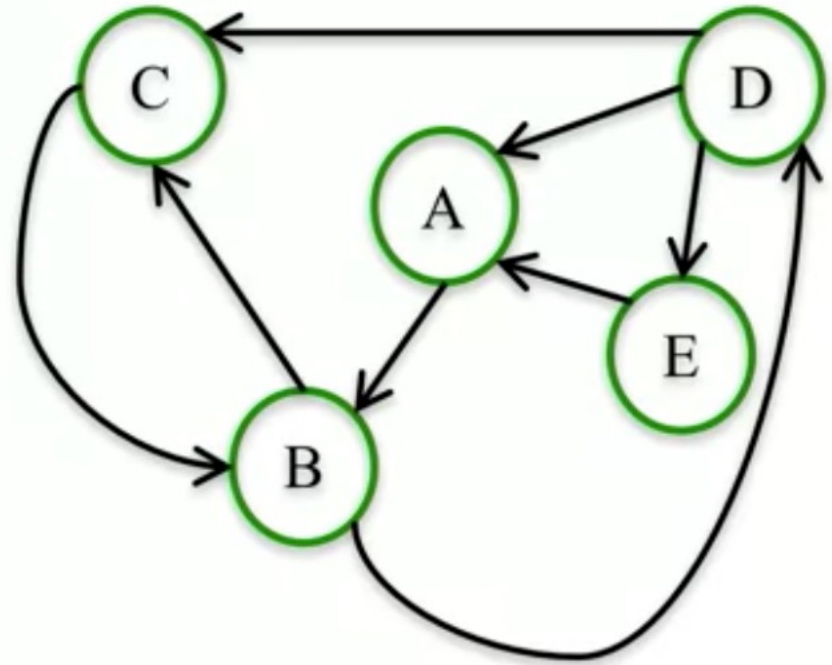
Page Rank (k = 2)					
	A	B	C	D	E
Old	4/15	2/5	1/6	1/10	1/15
New	1/10	13/30	7/30	2/10	

A: (1/3)*(1/10) + 1/15 = 1/10
B: 1/6 + 4/15 = 13/30
C: (1/3)*(1/10) + (1/2)*(2/5) = 7/30
D: (1/2)*(2/5) = 2/10
E: (1/3)*(1/10)



PageRank

	Page Rank				
	A	B	C	D	E
k=2	1/10	13/30	7/30	2/10	1/30
k=2	.1	.43	.23	.20	.03
k=3	.1	.33	.28	.22	.06
k=∞	.12	.38	.25	.19	.06



What if continue with k = 4,5,6,...?

Summary

Steps of Basic PageRank:

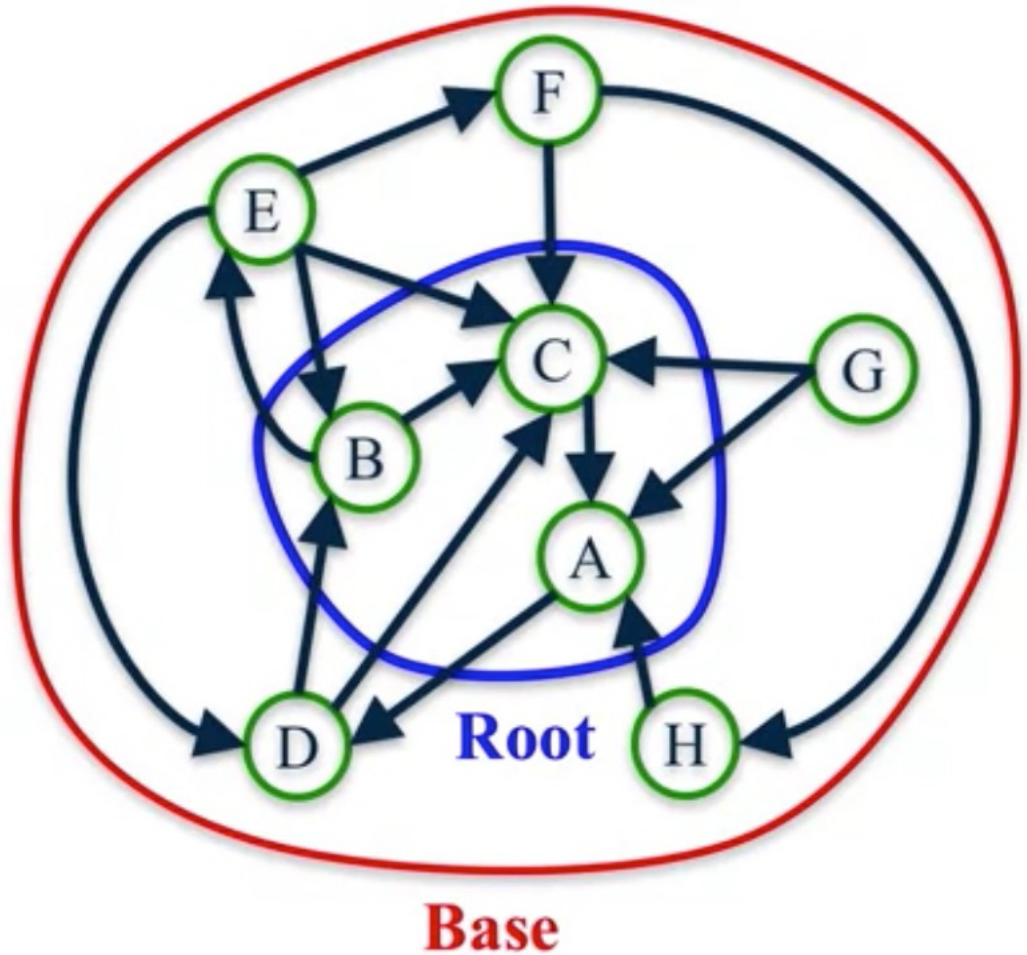
1. All nodes start with PageRank of $1/n$
2. Perform the *Basic PageRank Update Rule* k times:
   - **Basic PageRank Update Rule**: Each node gives an equal share of its current PageRank to all the nodes it links to.
   - The new PageRank of each node is the sum of all the PageRank it received from other nodes.

For most networks, PageRank values converge as k gets larger ($k \to \infty$)

Hubs and Authorities

Given a query to a search engine:

- **Root**: set of highly relevant web pages (e.g. pages that contain the query string) – potential *authorities*.
- Find all pages that link to a page in root – potential *hubs*.
- **Base**: root nodes and any node that links to a node in root.
- Consider all edges connecting nodes in the base set.



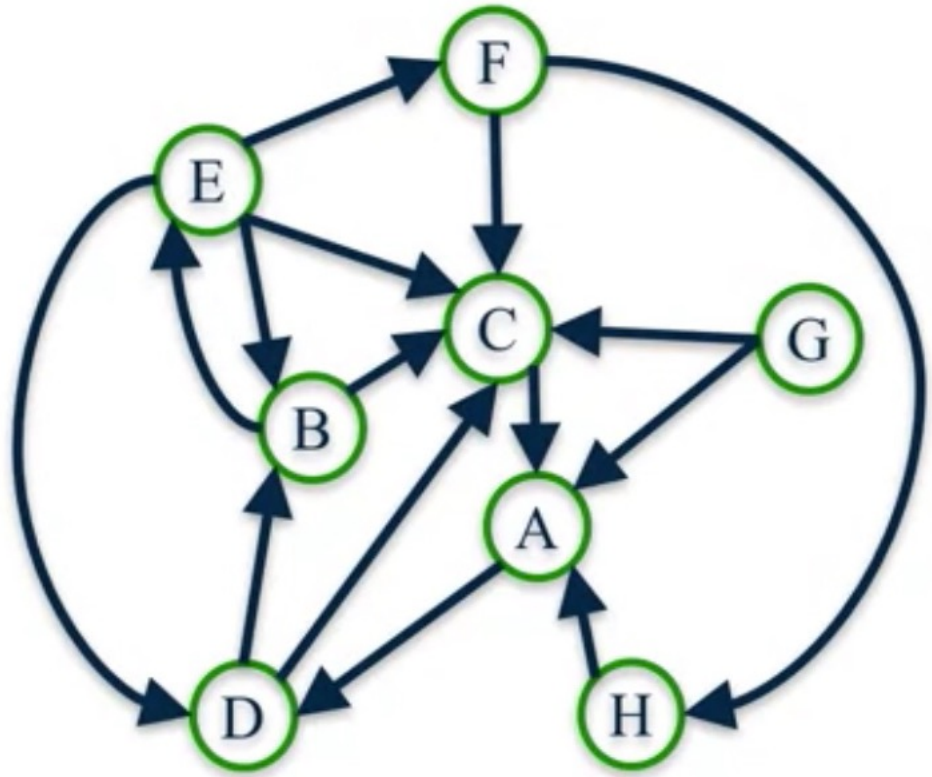
HITS Algorithm

Computing k iterations of the HITS algorithm to assign an *authority score* and *hub score* to each node.

1. Assign each node an authority and hub score of 1.
2. Apply the ***Authority Update Rule***: each node's *authority* score is the sum of *hub* scores of each node that *points to it*.
3. Apply the ***Hub Update Rule***: each node's hub score is the sum of authority scores of each node that *it points to*.
4. **Nomalize** Authority and Hub scores: $\text{auth}(j) = \frac{\text{auth}(j)}{\sum_{i \in N} \text{auth}(i)}$

HITS Algorithm Example

	Old Auth	Old Hub	New Auth	New Hub
A	1	1	3	1
B	1	1	2	2
C	1	1	5	1
D	1	1	2	2
E	1	1	1	4
F	1	1	1	2
G	1	1	0	2
H	1	1	1	1

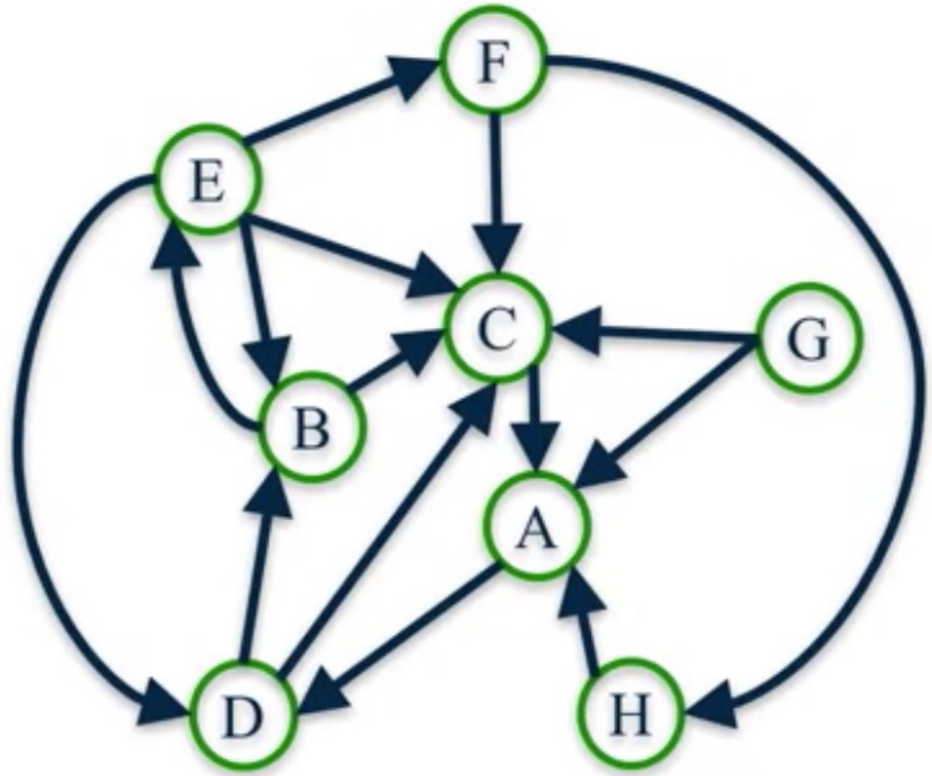


Normalize:

$\sum_{i \in N} \text{auth}(i) = 15 \quad \sum_{i \in N} \text{hub}(i) = 15$

HITS Algorithm Example

	Old Auth	Old Hub	New Auth	New Hub
A	1	1	3/15	1/15
B	1	1	2/15	2/15
C	1	1	5/15	1/15
D	1	1	2/15	2/15
E	1	1	1/15	4/15
F	1	1	1/15	2/15
G	1	1	0/15	2/15
H	1	1	1/15	1/15

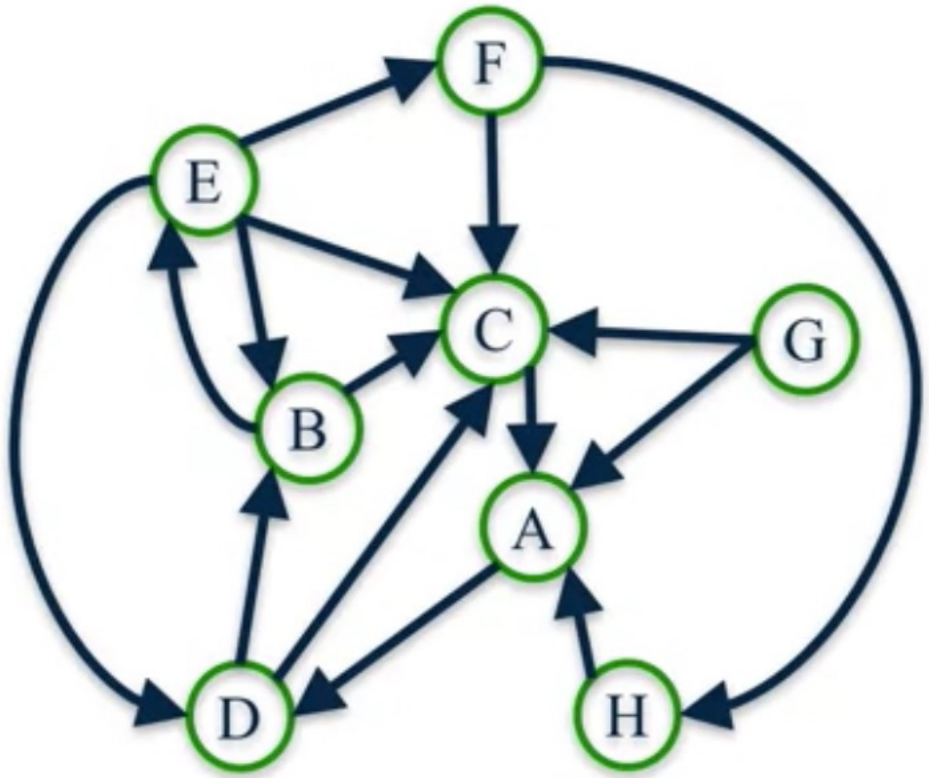


Normalize:

$\sum_{i \in N} \text{auth}(i) = 15 \quad \sum_{i \in N} \text{hub}(i) = 15$

HITS Algorithm Example

	Old Auth	Old Hub	New Auth	New Hub
A	1/5	1/15	4/15	2/15
B	2/15	2/15	6/15	2/5
C	1/3	1/15	12/15	1/5
D	2/15	2/15	1/3	7/15
E	1/15	4/15	2/15	2/3
F	1/15	2/15	4/15	2/5
G	0	2/15	0	8/15
H	1/15	1/15	2/15	1/5

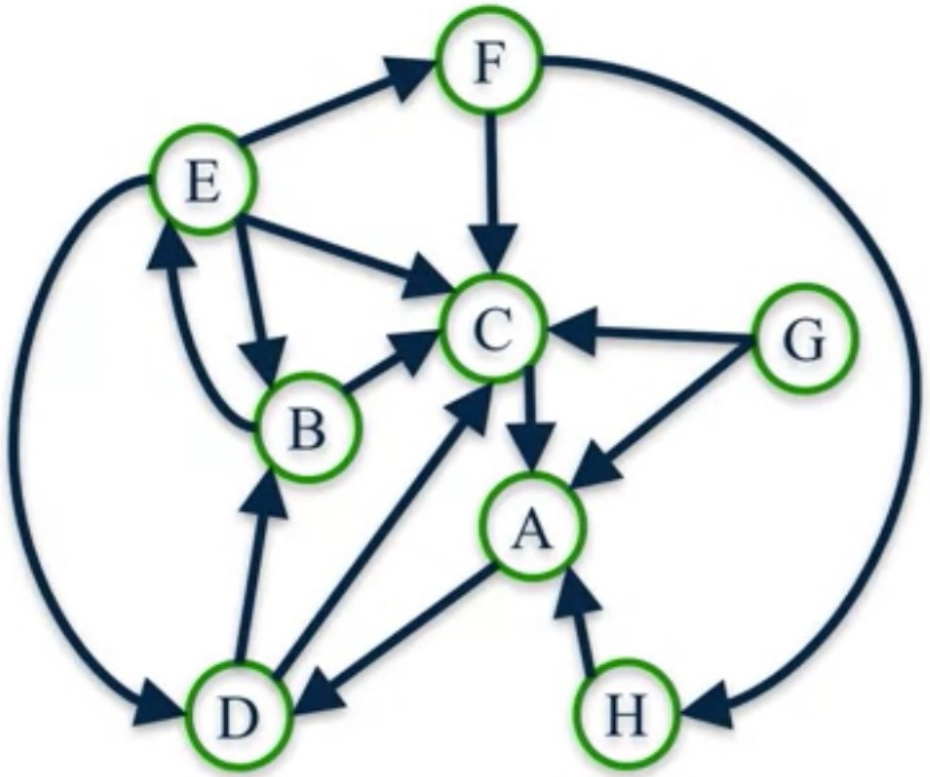


Normalize:

$$\sum_{i \in N} \text{auth}(i) = {}^{35}/_{15}$$

HITS Algorithm Example

	Old Auth	Old Hub	New Auth	New Hub
A	1/5	1/15	4/35	2/15
B	2/15	2/15	6/35	2/5
C	1/3	1/15	12/35	1/5
D	2/15	2/15	1/7	7/15
E	1/15	4/15	2/35	2/3
F	1/15	2/15	4/35	2/5
G	0	2/15	0	8/15
H	1/15	1/15	2/35	1/5

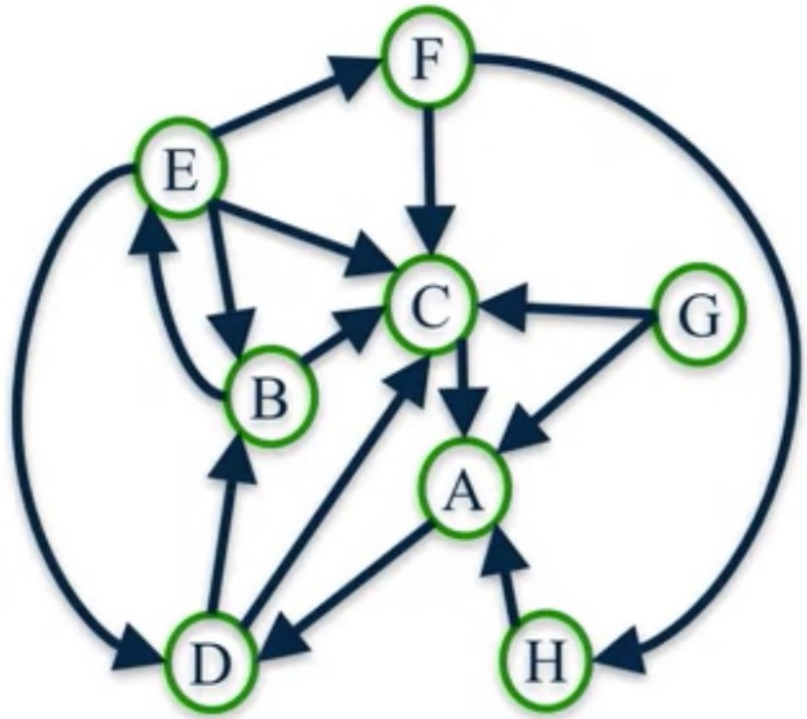


Normalize:

$$\sum_{i \in N} \text{hub}(i) = {}^{45}/_{15} = 3$$

HITS Algorithm Convergence

	k	A	B	C	D	E	F	G	H
Auth	**2**	.11	.17	.34	.14	.06	.11	0	.06
	4	.10	.18	.36	.13	.06	.11	0	.06
	6	.09	.19	.37	.13	.06	.11	0	.06
Hub	**2**	.04	.13	.07	.16	.22	.13	.18	.07
	4	.04	.14	.05	.18	.25	.14	.17	.04
	6	.04	.14	.04	.18	.26	.14	.16	.04

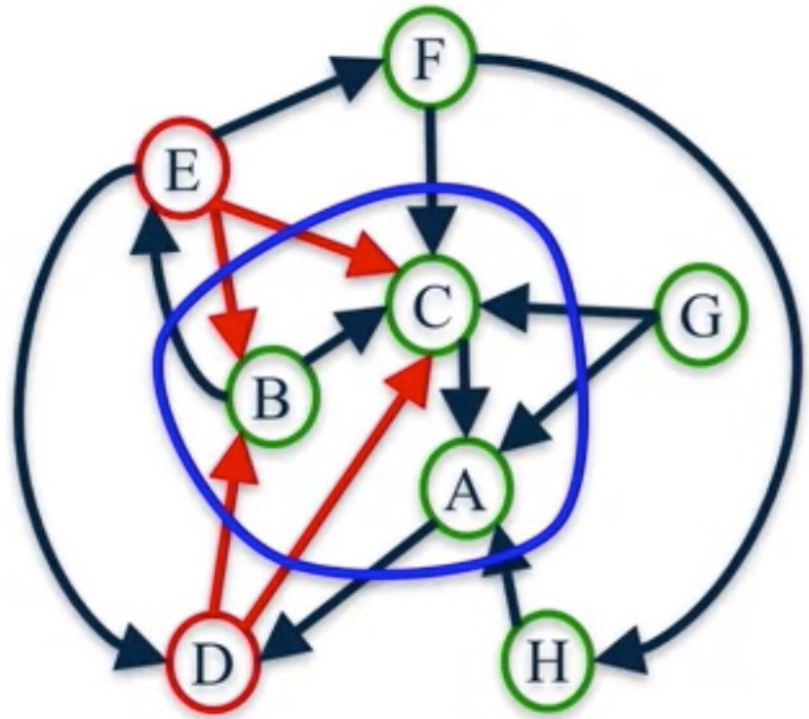


HITS Algorithm Convergence

For most networks, as k gets larger, authority and hub scores converge to a unique value.

As $k \longrightarrow \infty$ the hub and authority scores approach:

	A	B	C	D	E	F	G	H
Auth	.08	**.19**	**.40**	.13	.06	.11	0	.06
Hub	.04	.14	.03	**.19**	**.27**	.14	.15	.03



HITS Algorithm NetworkX

You can use NetworkX funtion hits(G) to compute the hub and authority scores of network G.

hits(G) outputs two dictionaries, keyed by node, with the hub and authority scores of the nodes.