Big Data Computing

Master's Degree in Computer Science

2019-2020

Gabriele Tolomei

Department of Computer Science

Sapienza Università di Roma

tolomei@di.uniroma1.it

SAPIENZA
UNIVERSITÀ DI ROMA

Recap from Previous Lecture

- We want to find an effective way to measure the **trustworthiness** of a page within the Web graph

Recap from Previous Lecture

- We want to find an effective way to measure the **trustworthiness** of a page within the Web graph
- More generally, we want to assign a score which indicates the **importance** of a node in a graph

Recap from Previous Lecture

- We want to find an effective way to measure the **trustworthiness** of a page within the Web graph
- More generally, we want to assign a score which indicates the **importance** of a node in a graph
- Derive such a score from the structural properties of the graph only (i.e., via **link analysis**)

Recap from Previous Lecture

- We want to find an effective way to measure the **trustworthiness** of a page within the Web graph
- More generally, we want to assign a score which indicates the **importance** of a node in a graph
- Derive such a score from the structural properties of the graph only (i.e., via **link analysis**)
- Exploit the fact that the Web is an example of a **scale-free network**

Computing Node Importance

Several **link analysis** approaches to compute **web page importance**

PageRank

Hubs and Authorities (HITS)

Personalized PageRank

Web Spam Detection

PageRank

One Slide PageRank

- A link analysis approach to the definition of web page importance

One Slide PageRank

- A link analysis approach to the definition of web page importance
- Introduced in 1998 by Sergey Brin and Larry Page*

*The Anatomy of a Large-Scale Hypertextual Web Search Engine. In Computer Networks, vol. 30, n. 1-7, pp. 107-117, 1998.

One Slide PageRank

- A link analysis approach to the definition of web page importance
- Introduced in 1998 by Sergey Brin and Larry Page*
- The core of Google search engine

*[*The Anatomy of a Large-Scale Hypertextual Web Search Engine*]. In Computer Networks, vol. 30, n. 1-7, pp. 107-117, 1998.

One Slide PageRank

- A link analysis approach to the definition of web page importance
- Introduced in 1998 by Sergey Brin and Larry Page*
- The core of Google search engine
- Assigns a numerical score to each web page with the purpose of indicating its relative importance within the whole collection

*[The Anatomy of a Large-Scale Hypertextual Web Search Engine](). In Computer Networks, vol. 30, n. 1-7, pp. 107-117, 1998.

PageRank's Intuition: Links as Votes

Based on **2** intuitions

PageRank's Intuition: Links as Votes

Based on **2** intuitions

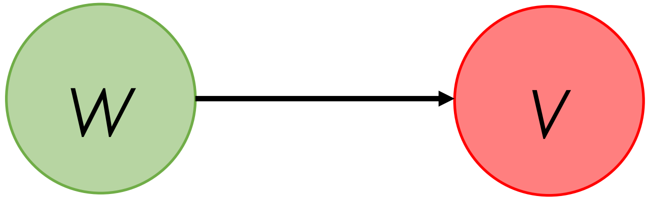
The more incoming links a web page has
the more important it is

PageRank's Intuition: Links as Votes

Based on **2** intuitions

The more incoming links a web page has
the more important it is

Each link from a web page *w* to a web page *v*
is interpreted as a **vote** by *w* to *v*



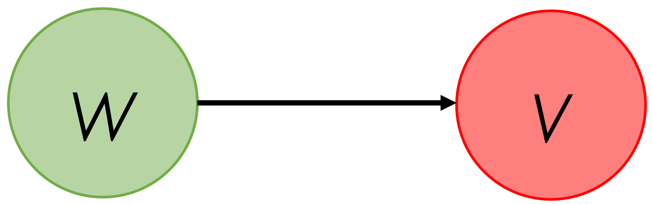
PageRank's Intuition: Links as Votes

Based on **2** intuitions

The more incoming links a web page has the more important it is

Links (i.e., votes) from important web pages should count more!

Each link from a web page *w* to a web page *v* is interpreted as a **vote** by *w* to *v*

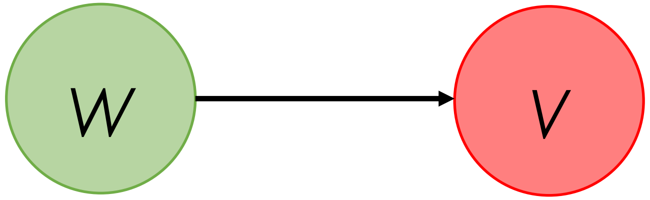


PageRank's Intuition: Links as Votes

Based on **2** intuitions

The more incoming links a web page has the more important it is

Each link from a web page *w* to a web page *v* is interpreted as a **vote** by *w* to *v*



Links (i.e., votes) from important web pages should count more!

Different web pages have different in-degree (scale-free network)

www.stanford.edu has more than 23K in-links

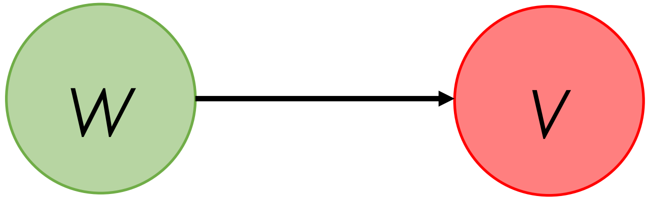
www.uniroma1.it/~tolomei has one or two in-links!

PageRank's Intuition: Links as Votes

Based on **2** intuitions

The more incoming links a web page has the more important it is

Each link from a web page *w* to a web page *v* is interpreted as a **vote** by *w* to *v*



Links (i.e., votes) from important web pages should count more!

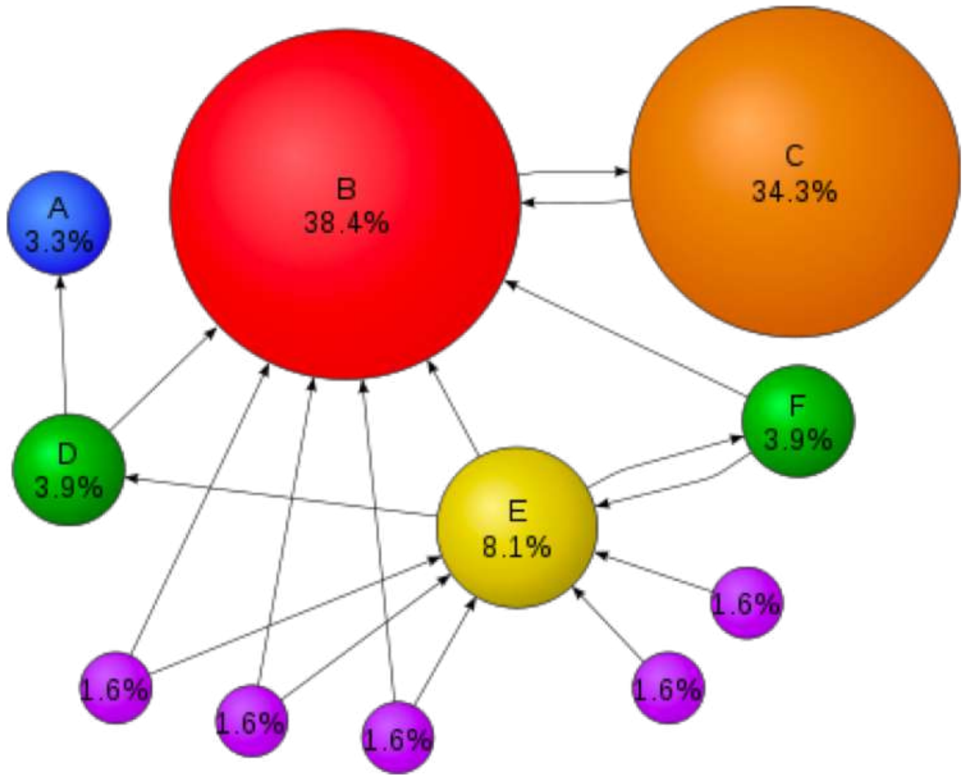
Different web pages have different in-degree (scale-free network)

www.stanford.edu has more than 23K in-links

www.uniroma1.it/~tolomei has one or two in-links!

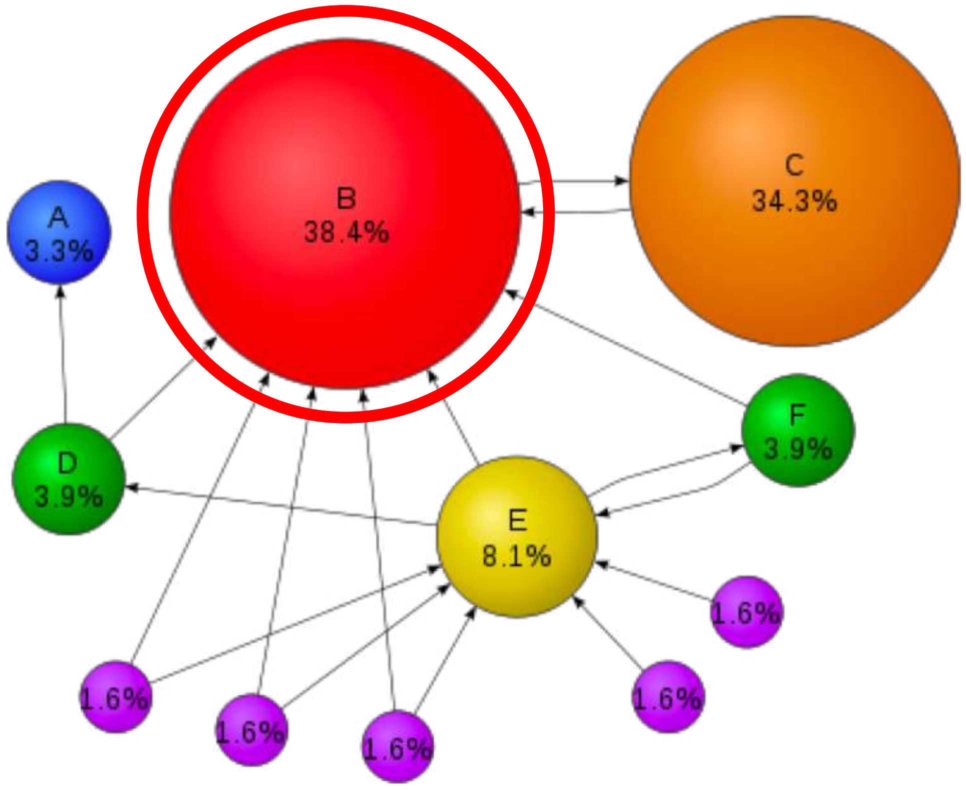
Recursive definition

PageRank Scores: Example



Circle size proportional to the node importance

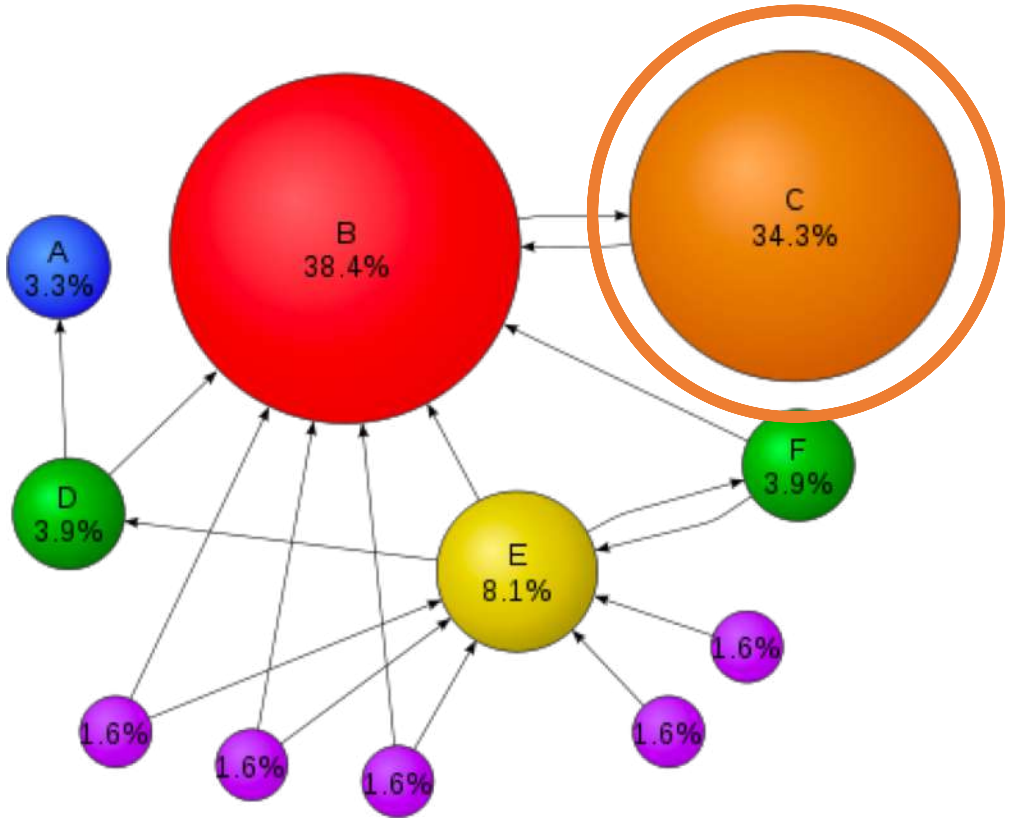
PageRank Scores: Example



Circle size proportional to the node importance

B has a high score since many nodes point to it

PageRank Scores: Example

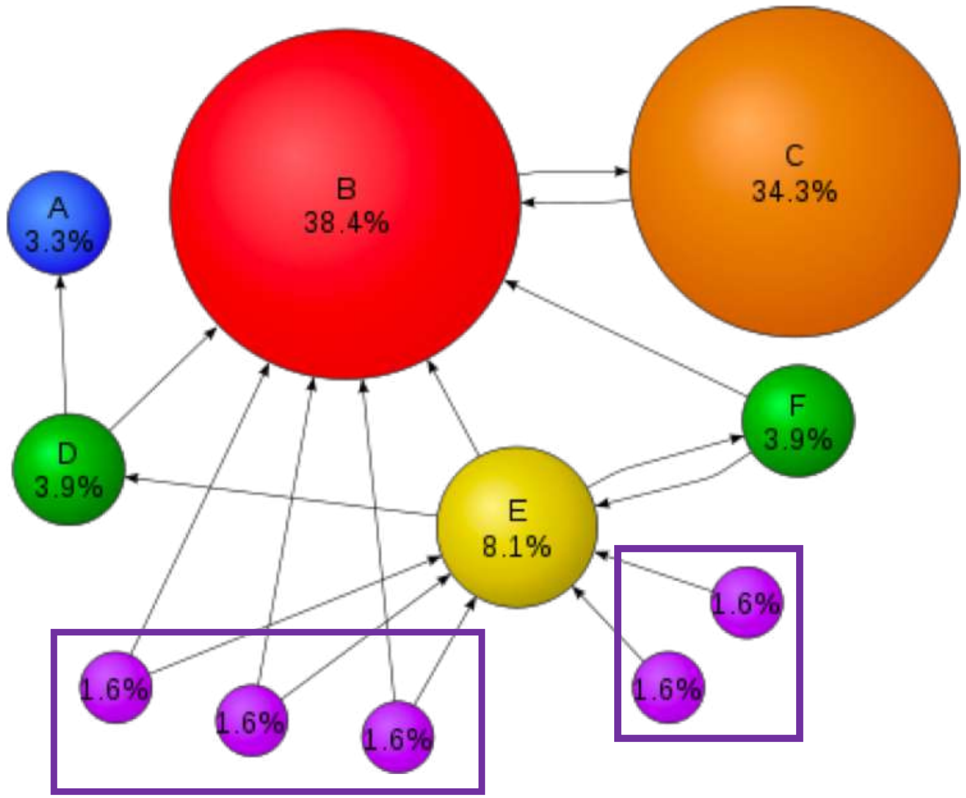


Circle size proportional to the node importance

B has a high score since many nodes point to it

C also has a high score even though it has only one incoming link but from an important node **B**

PageRank Scores: Example



Circle size proportional to the node importance

B has a high score since many nodes point to it

C also has a high score even though it has only one incoming link but from an important node **B**

Many other less important nodes

PageRank: Prelminaries

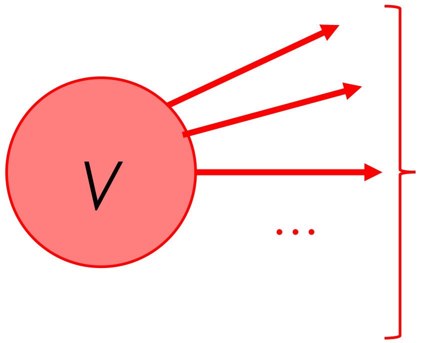
$G = (V, E)$ The Web Graph $|V| = N$ Number of Nodes (pages)

PageRank: Prelminaries

$G = (V, E)$ The Web Graph $|V| = N$ Number of Nodes (pages)

$O_v = \{w \in V : (v, w) \in E\}$ Set of pages linked by *v*

$|O_v| = o_v$ Out-degree of node *v*



PageRank: Prelminaries

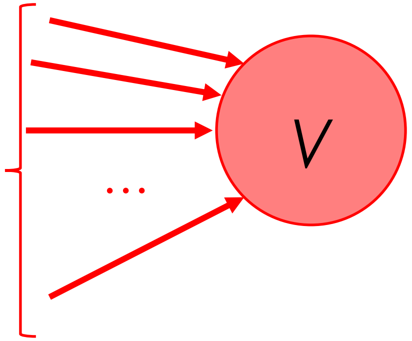
$G = (V, E)$ The Web Graph $|V| = N$ Number of Nodes (pages)

$O_v = \{w \in V : (v, w) \in E\}$ Set of pages linked by *v*

$|O_v| = o_v$ Out-degree of node *v*

$I_v = \{w \in V : (w, v) \in E\}$ Set of pages linked to *v*

$|I_v| = i_v$ In-degree of node *v*



PageRank: First Simple Recursive Formulation

Each link's vote to a page *v* is proportional to the importance of the source page *w*, which the link comes from

PageRank: First Simple Recursive Formulation

Each link's vote to a page v is proportional to the importance of the source page w, which the link comes from

If a page w has importance r_w and out-degree o_w, each out-link will get an **equal proportion** of the importance, i.e., r_w/o_w

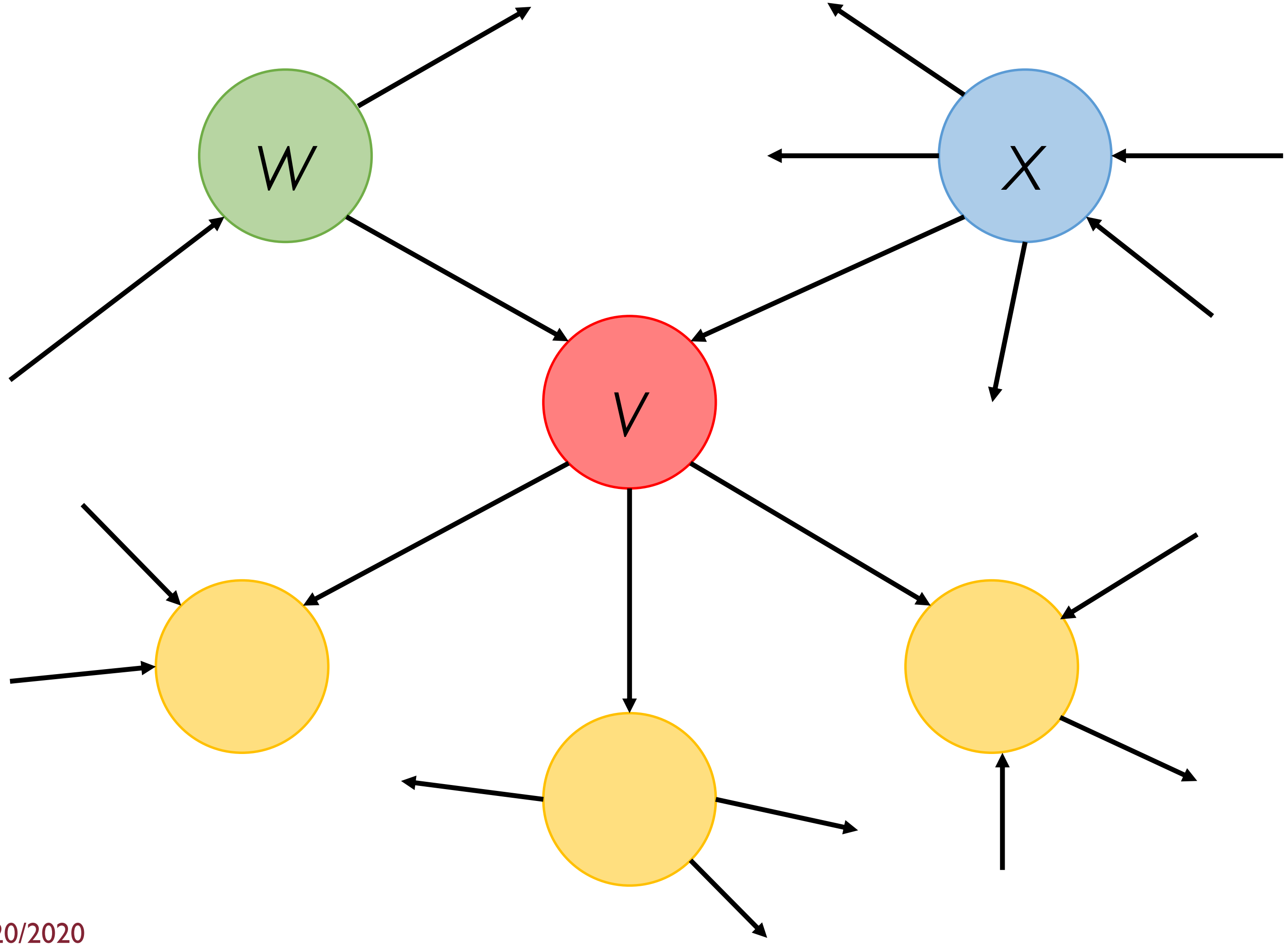
PageRank: First Simple Recursive Formulation

Each link's vote to a page v is proportional to the importance of the source page w, which the link comes from

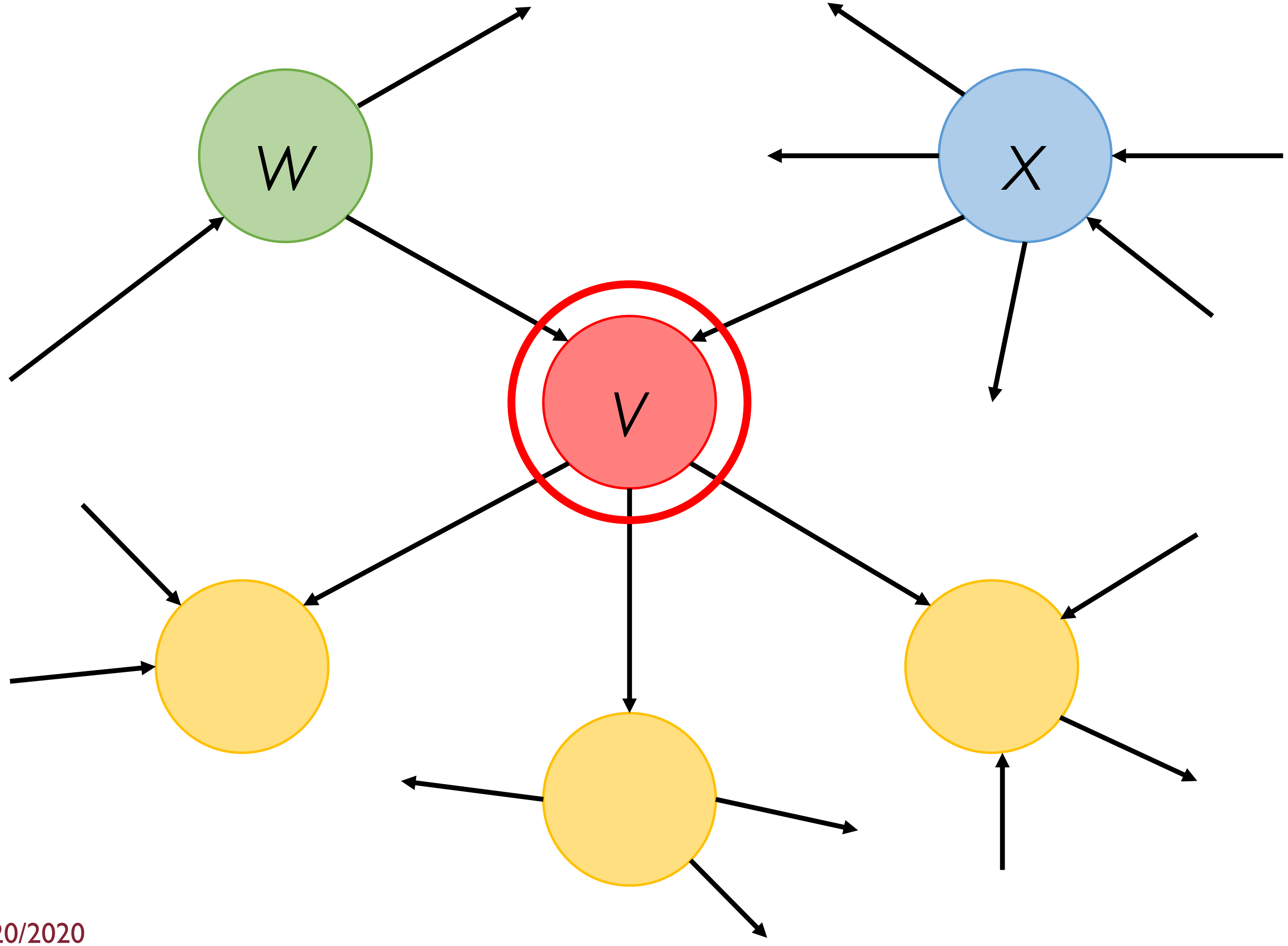
If a page w has importance r_w and out-degree o_w, each out-link will get an **equal proportion** of the importance, i.e., r_w/o_w

Each page v's importance can be computed just as the **sum of votes** of all its **incoming links** (i.e., in-degree)

PageRank: First Simple Recursive Formulation

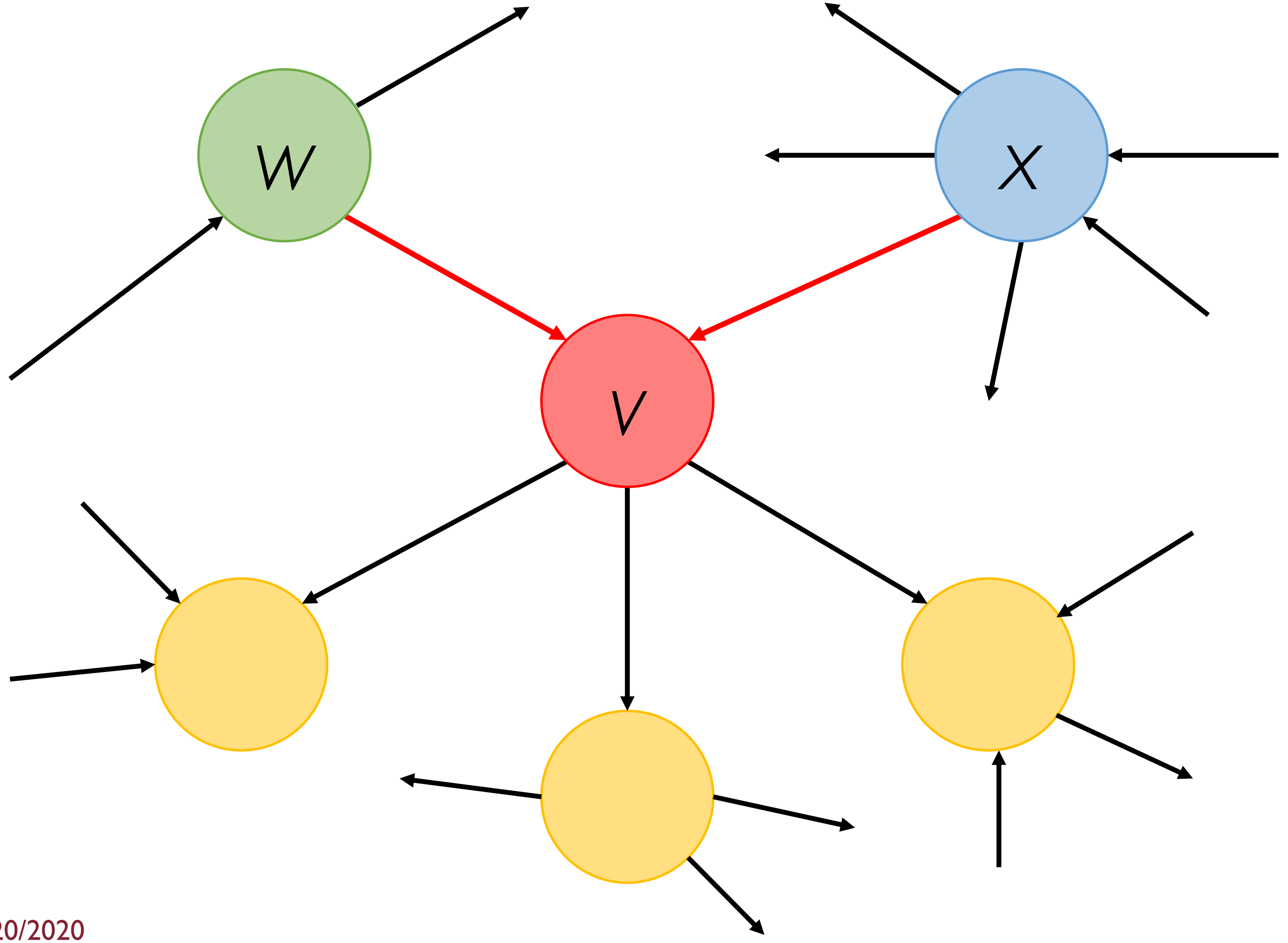


PageRank: First Simple Recursive Formulation



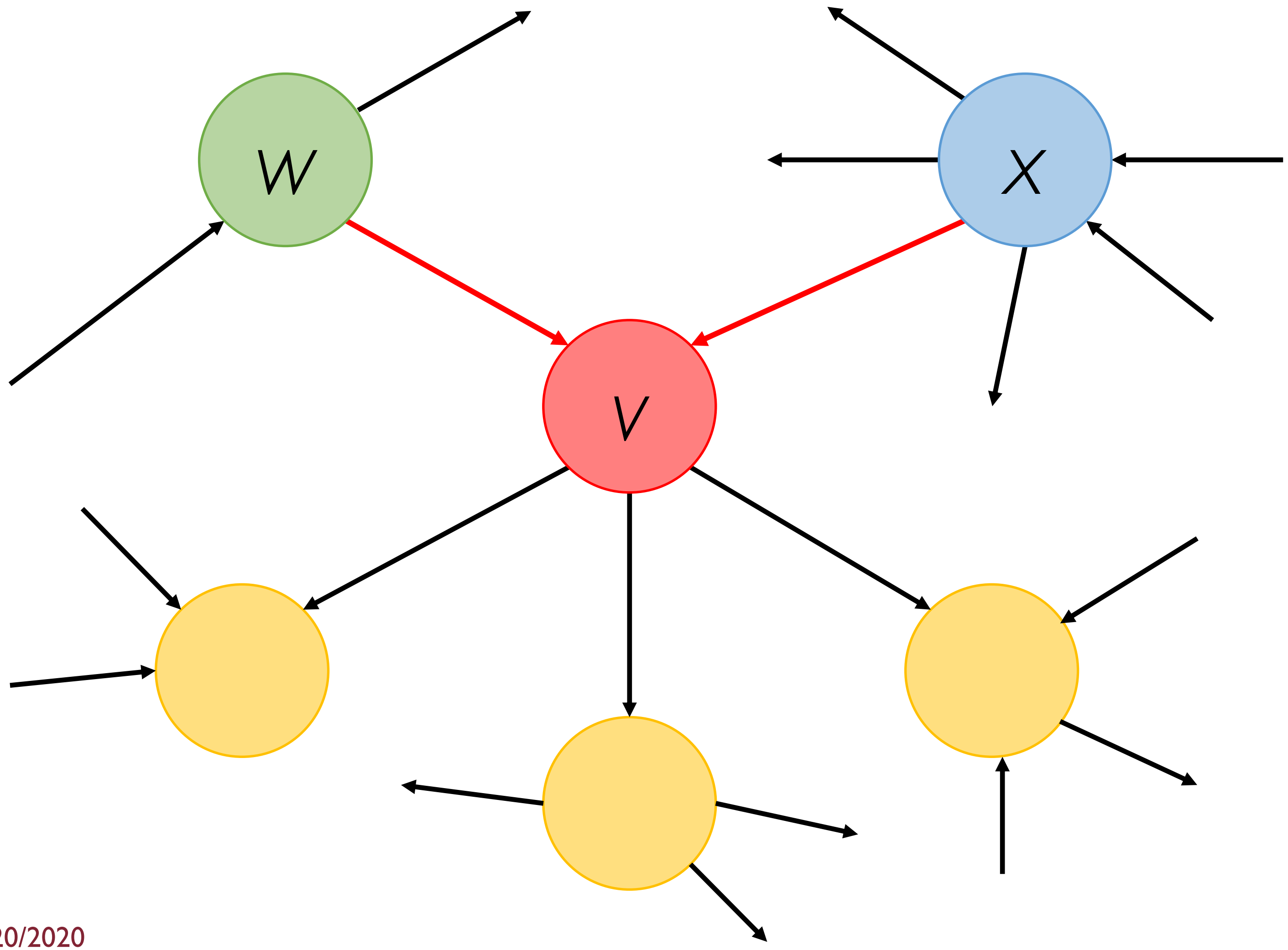
What is r_v?

PageRank: First Simple Recursive Formulation



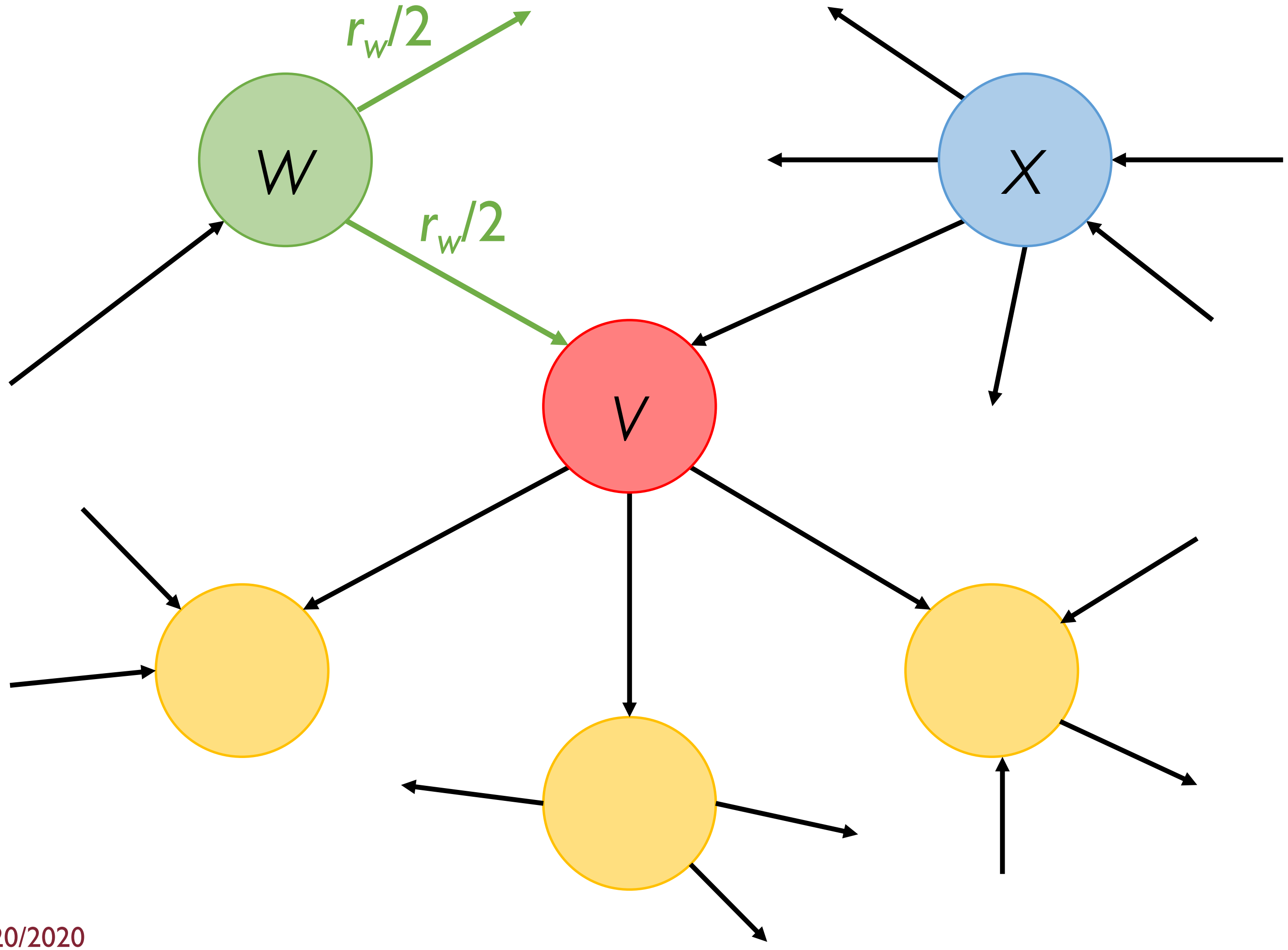
Suppose *v* has only **2** in-links coming from *w* and *x*

PageRank: First Simple Recursive Formulation



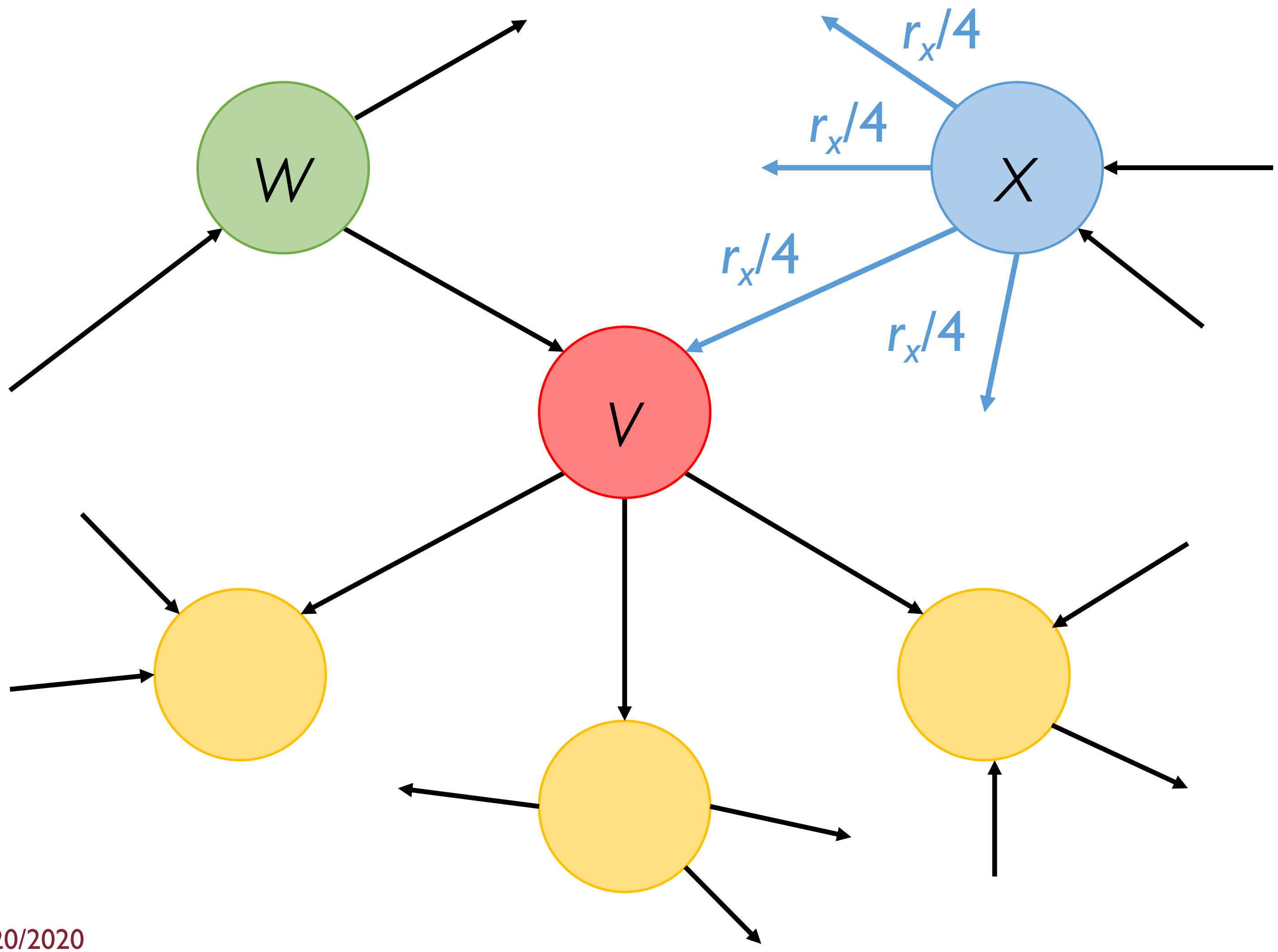
We must compute the in-link's **vote** from *w* and from *x*

PageRank: First Simple Recursive Formulation



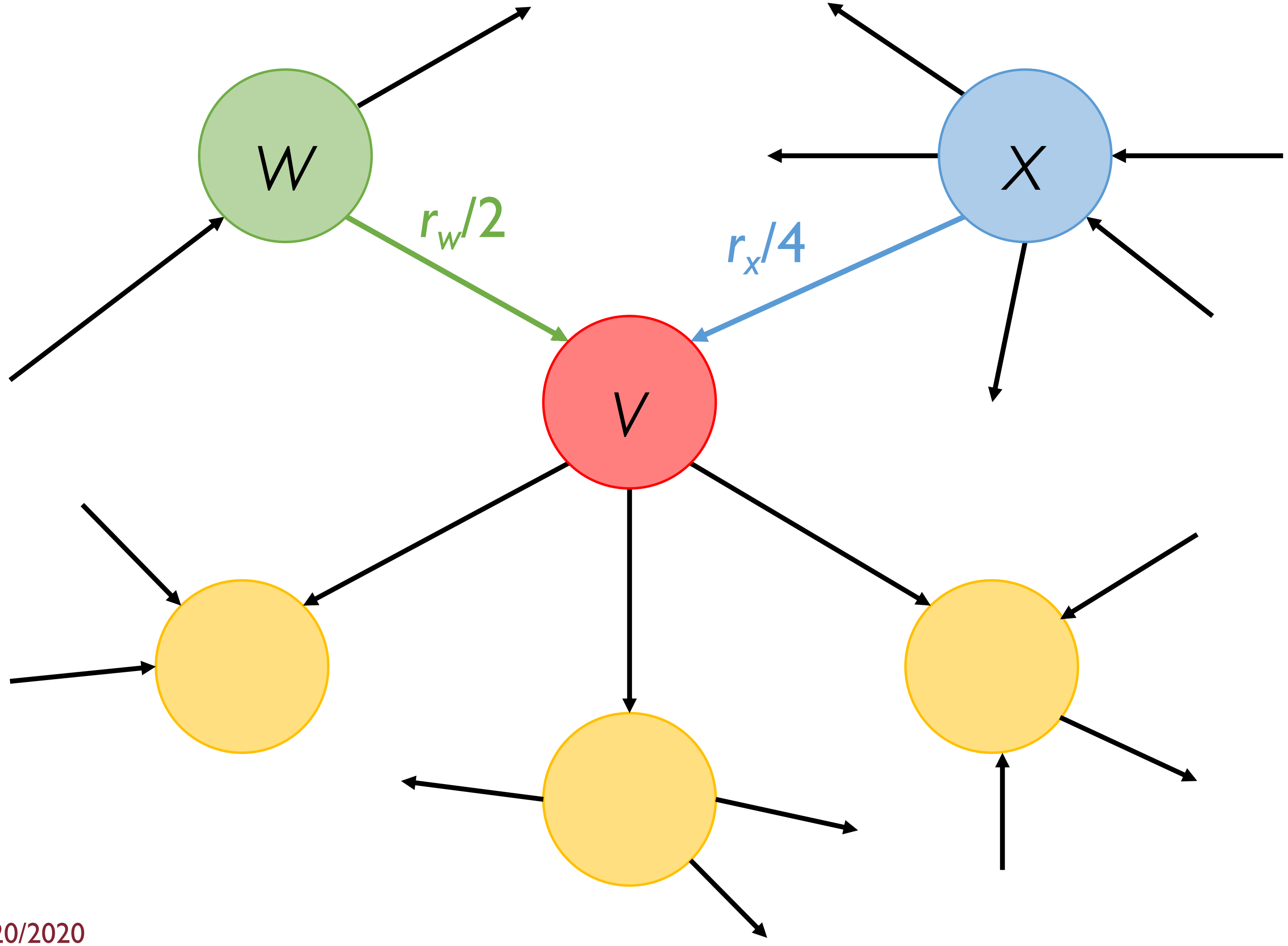
The importance of page w (r_w) is distributed across each of its 2 outgoing links

PageRank: First Simple Recursive Formulation



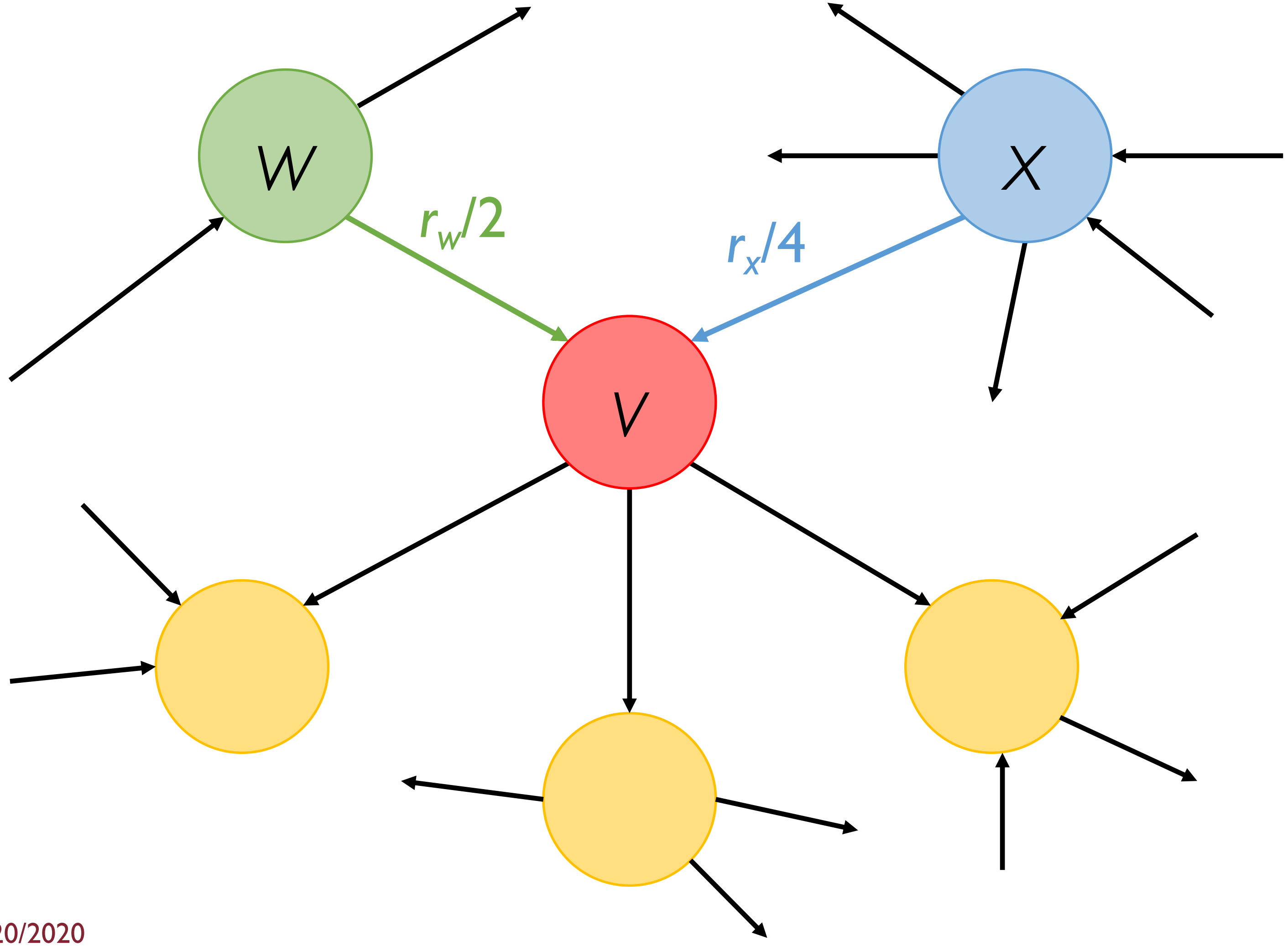
The importance of page x (r_x) is distributed across each of its 4 outgoing links

PageRank: First Simple Recursive Formulation



The importance of page v (r_v) is just the **sum** of its incoming links' votes

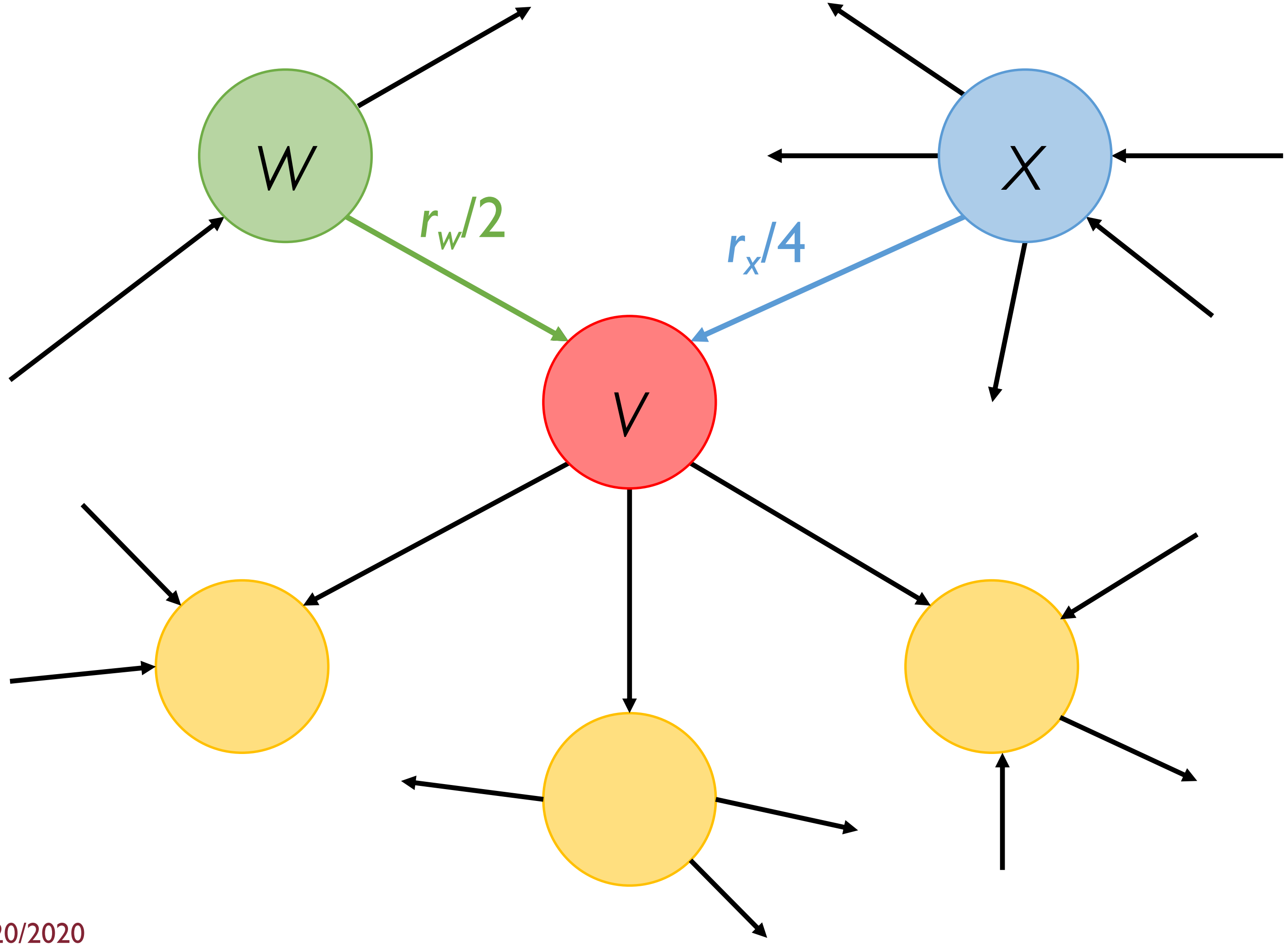
PageRank: First Simple Recursive Formulation



The importance of page v (r_v) is just the **sum** of its incoming links' votes

$$\boxed{r_v = r_w/2 + r_x/4}$$

PageRank: First Simple Recursive Formulation

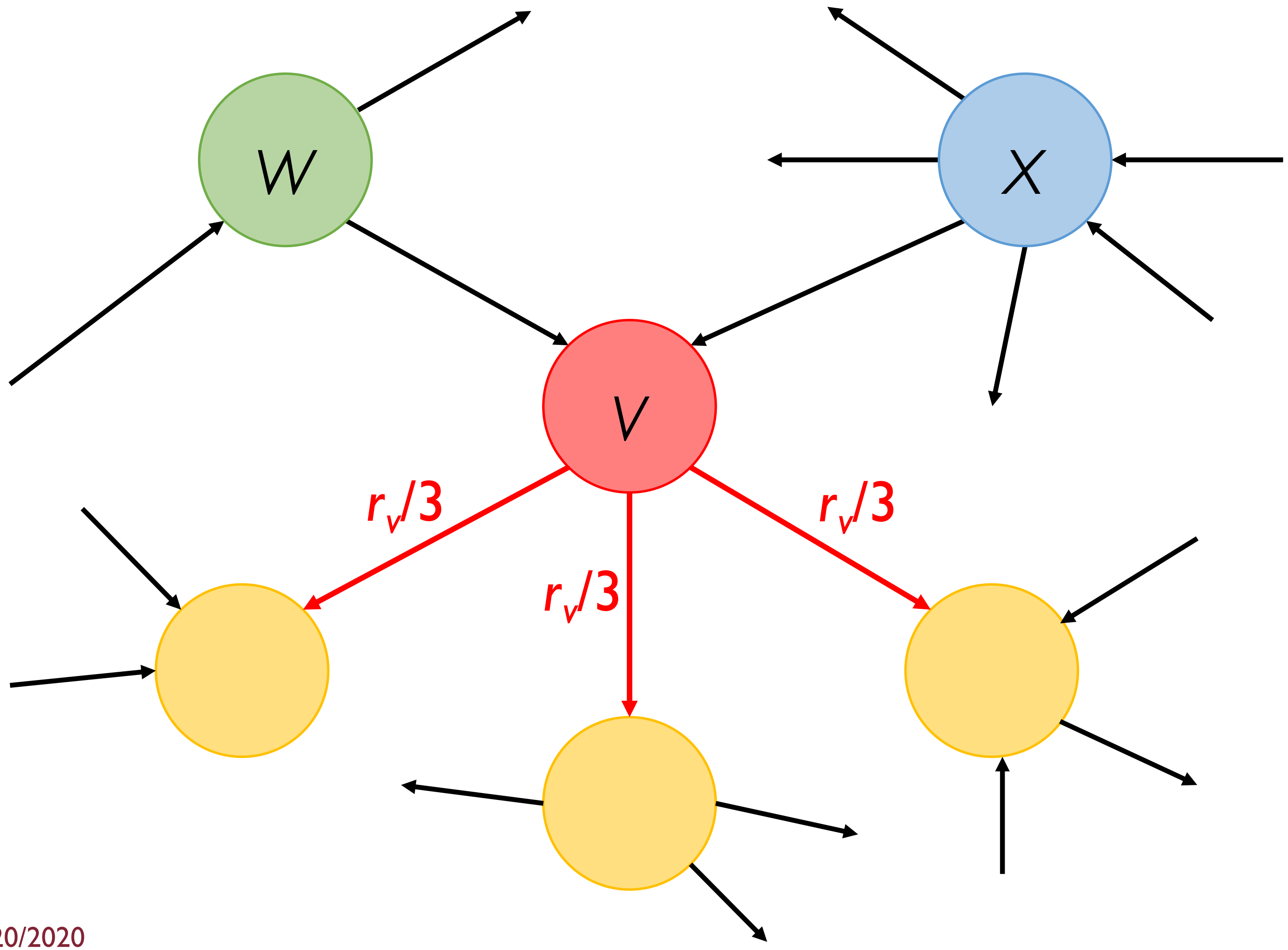


The importance of page v (r_v) is just the **sum** of its incoming links' votes

$$\boxed{r_v = r_w/2 + r_x/4}$$

$$\boxed{r_v = \sum_{u \in I_v} \frac{r_u}{o_u}}$$

PageRank: First Simple Recursive Formulation



Similarly, page v **uniformly** distributes its importance r_v to its outgoing links

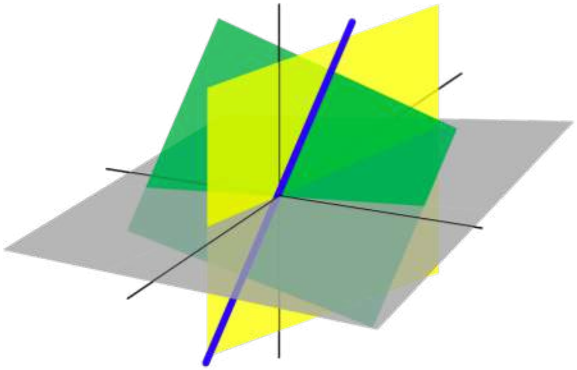
PageRank's Interpretations

$\boxed{\textbf{2} \text{ main perspectives}}$

PageRank's Interpretations

2 main perspectives

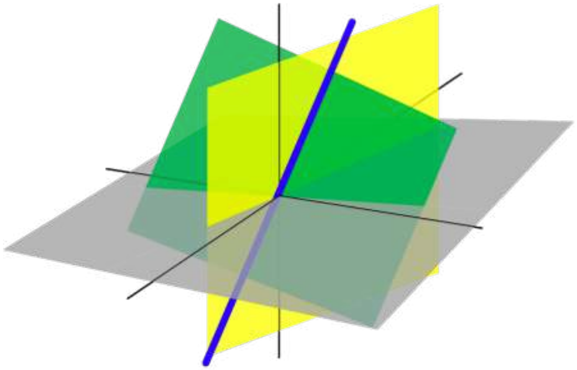
Linear Algebra



PageRank's Interpretations

2 main perspectives

Linear Algebra

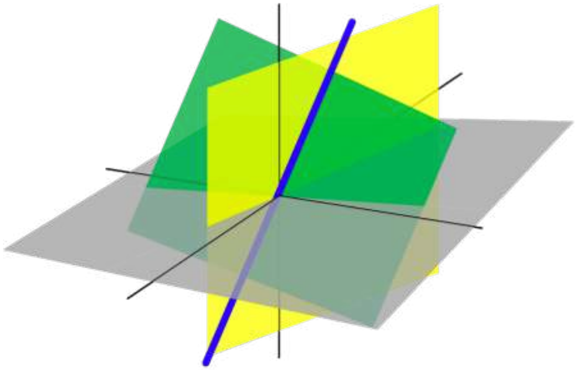


Probabilistic

PageRank's Interpretations

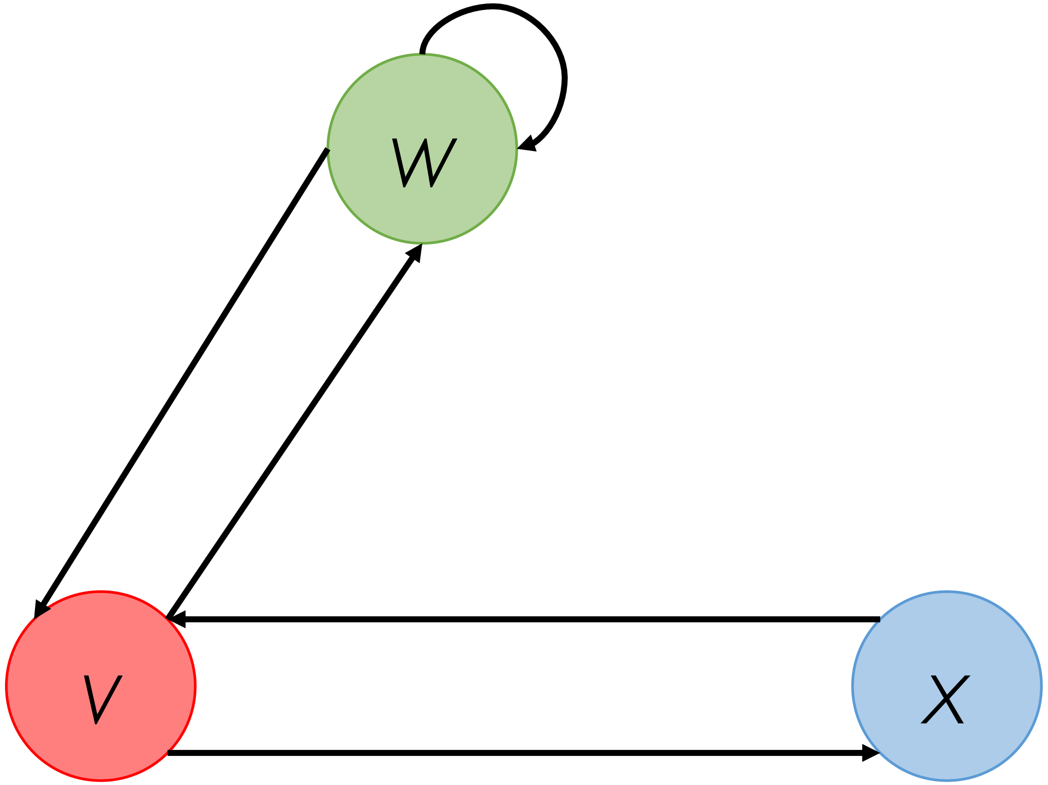
2 main perspectives

Linear Algebra

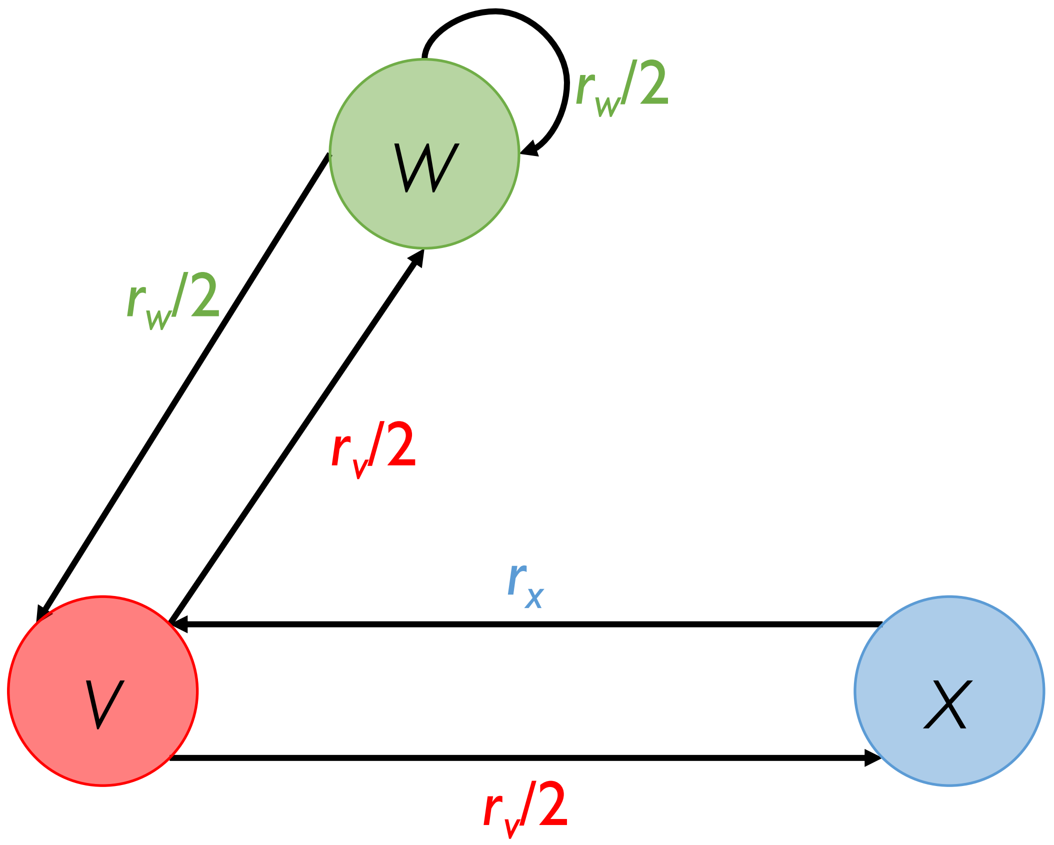


Probabilistic

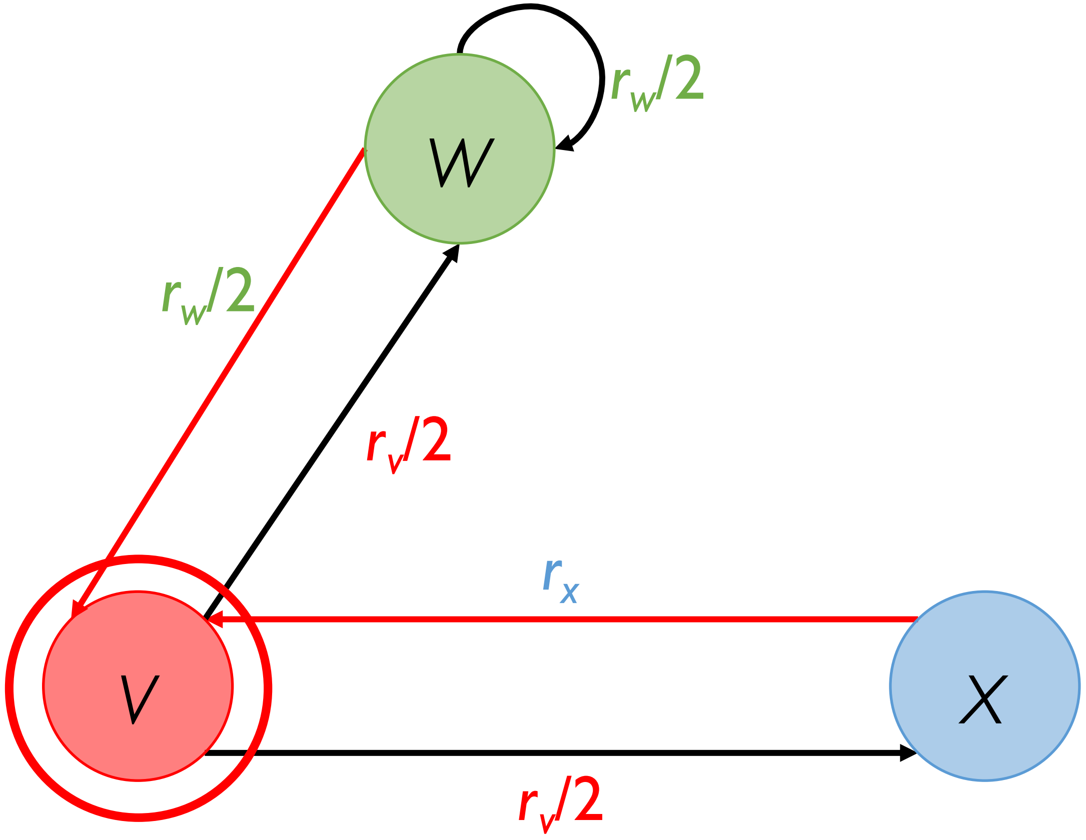
PageRank: The "Flow" Model



PageRank: The "Flow" Model

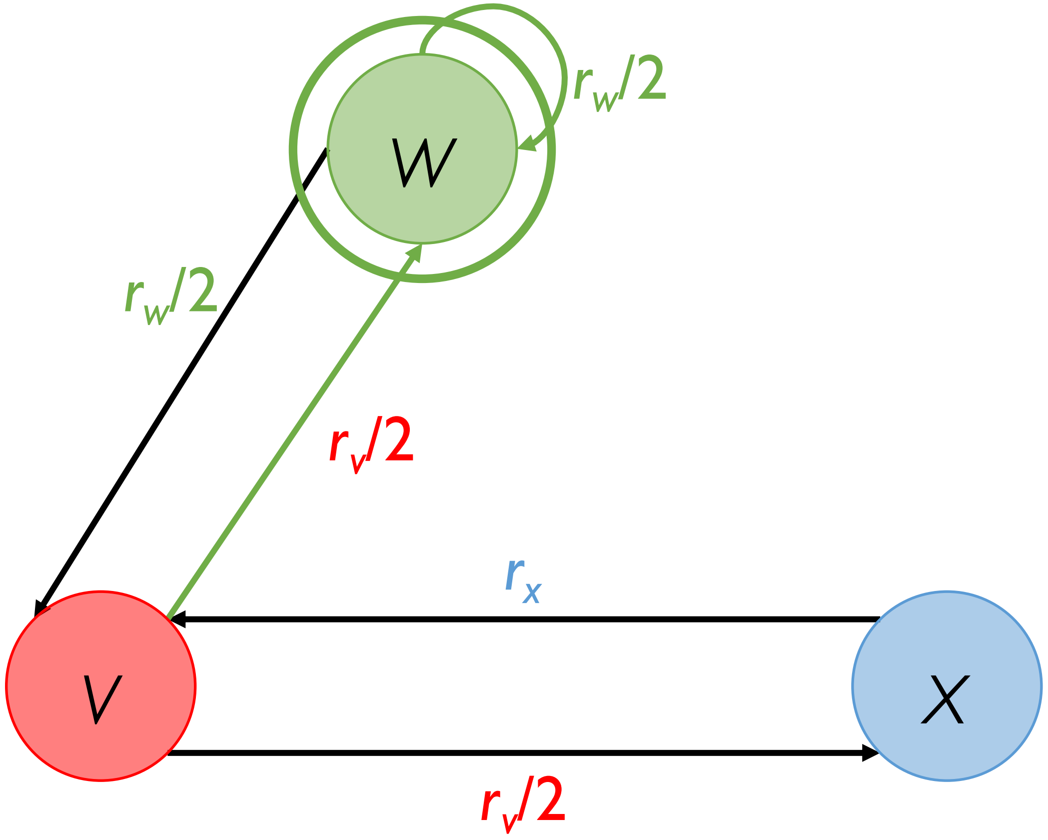


PageRank: The "Flow" Model



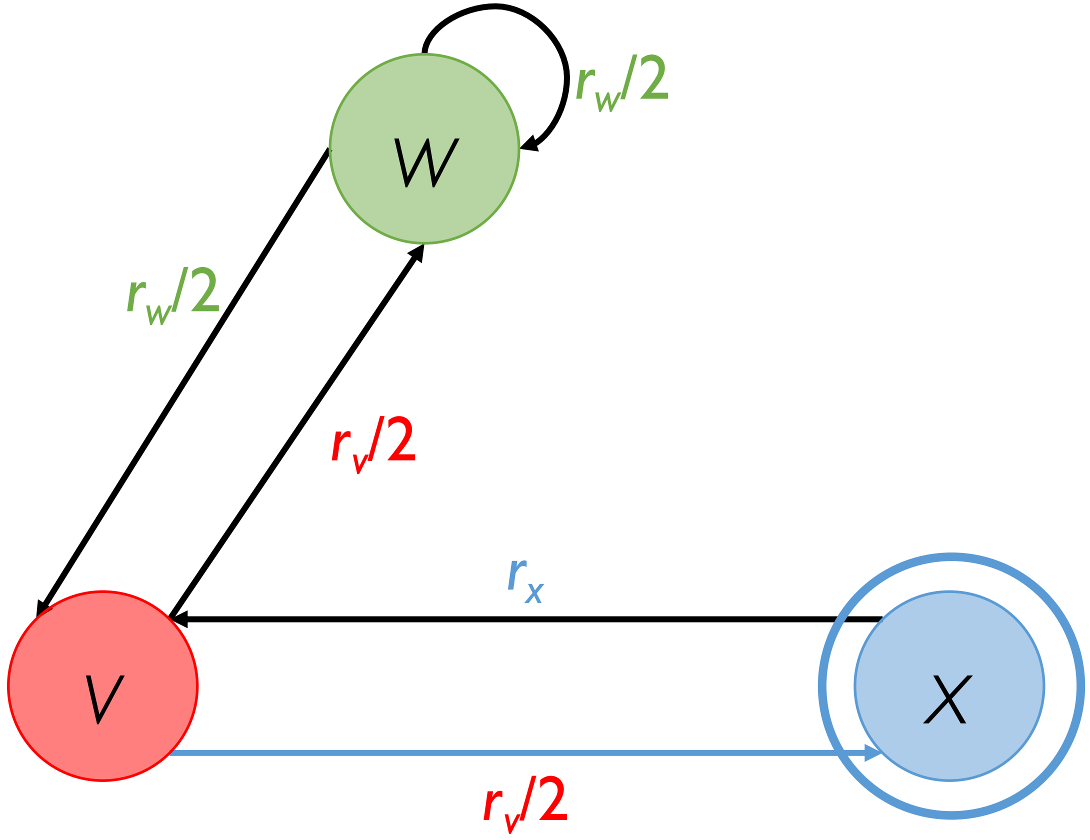
$$\begin{cases} \boxed{r_v = r_w/2 + r_x} \\ \\ \end{cases}$$

PageRank: The "Flow" Model



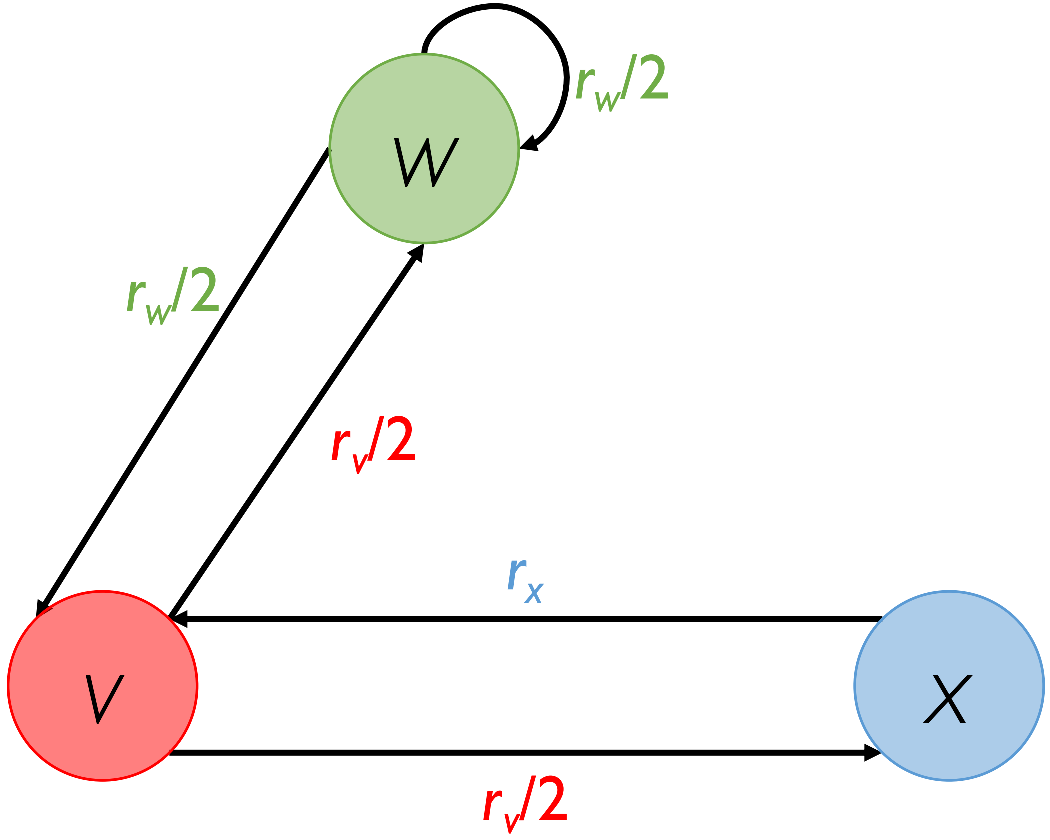
$$\begin{cases} r_v = r_w/2 + r_x \\ \boxed{r_w = r_v/2 + r_w/2} \end{cases}$$

PageRank: The "Flow" Model



$$\begin{cases} r_v = r_w/2 + r_x \\ r_w = r_v/2 + r_w/2 \\ \boxed{r_x = r_v/2} \end{cases}$$

PageRank: The "Flow" Model



$$\begin{cases} r_v = r_w/2 + r_x \\ r_w = r_v/2 + r_w/2 \\ r_x = r_v/2 \end{cases}$$

"Flow" Equations

Solving the System of "Flow" Equations

$$\begin{cases} r_v = r_w/2 + r_x \\ r_w = r_v/2 + r_w/2 \\ r_x = r_v/2 \end{cases}$$

3 equations with 3 unknowns: r_v, r_w, and r_x

Solving the System of "Flow" Equations

$$\begin{cases} r_v = r_w/2 + r_x \\ r_w = r_v/2 + r_w/2 \\ r_x = r_v/2 \end{cases}$$

3 equations with 3 unknowns: r_v, r_w, and r_x

But the first 2 equations are exactly the same if we substitute r_x

Solving the System of "Flow" Equations

$$\begin{cases} r_v = r_w/2 + r_x \\ r_w = r_v/2 + r_w/2 \\ r_x = r_v/2 \end{cases}$$

3 equations with 3 unknowns: r_v, r_w, and r_x

But the first 2 equations are exactly the same if we substitute r_x

No unique solution!

Infinitely many apart from a constant scale factor

Solving the System of "Flow" Equations

$$\begin{cases} r_v = r_w/2 + r_x \\ r_w = r_v/2 + r_w/2 \\ r_x = r_v/2 \\ \boxed{r_v + r_w + r_x = 1} \end{cases}$$

Additional constraint (equation) enforces the uniqueness of the solution

Solving the System of "Flow" Equations

$$\begin{cases} r_v = r_w/2 + r_x \\ r_w = r_v/2 + r_w/2 \\ r_x = r_v/2 \\ r_v + r_w + r_x = 1 \end{cases}$$

Additional constraint (equation) enforces the uniqueness of the solution

$$\boxed{r_v = r_w = \frac{2}{5} \quad r_x = \frac{1}{5}}$$

Solving the System of "Flow" Equations

$$\begin{cases} r_v = r_w/2 + r_x \\ r_w = r_v/2 + r_w/2 \\ r_x = r_v/2 \\ r_v + r_w + r_x = 1 \end{cases}$$

Additional constraint (equation) enforces the uniqueness of the solution

$$r_v = r_w = \frac{2}{5} \quad r_x = \frac{1}{5}$$

This may work for very small systems of linear equations (e.g., using Gaussian elimination)

Solving the System of "Flow" Equations

$$\begin{cases} r_v = r_w/2 + r_x \\ r_w = r_v/2 + r_w/2 \\ r_x = r_v/2 \\ r_v + r_w + r_x = 1 \end{cases}$$

Additional constraint (equation) enforces the uniqueness of the solution

$$r_v = r_w = \frac{2}{5} \quad r_x = \frac{1}{5}$$

In the case of web pages we might have **100s of billions** of equations!

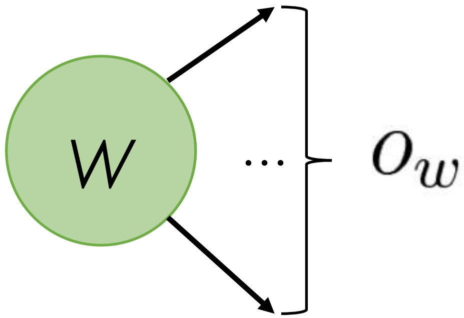
We need a new formulation

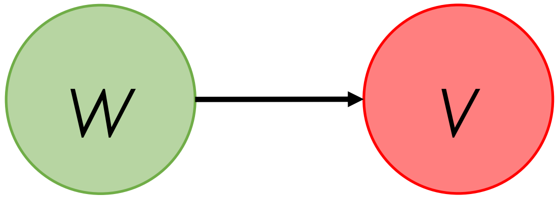
PageRank: The Matrix Formulation

Represent the Web graph of documents $G=(V, E)$ s.t. $|V|=N$
as a **column stochastic matrix** **M** of size NxN

PageRank: The Matrix Formulation

Represent the Web graph of documents G=(V, E) s.t. |V|=N
as a **column stochastic matrix M** of size NxN

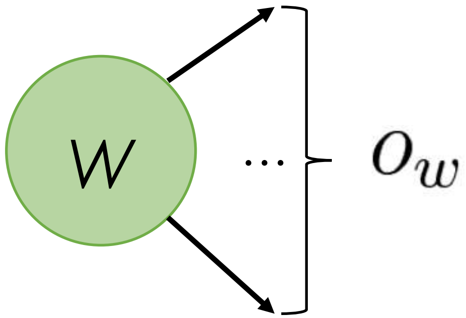


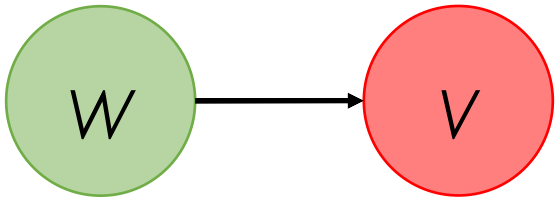


One of those o_w links points to page *v*

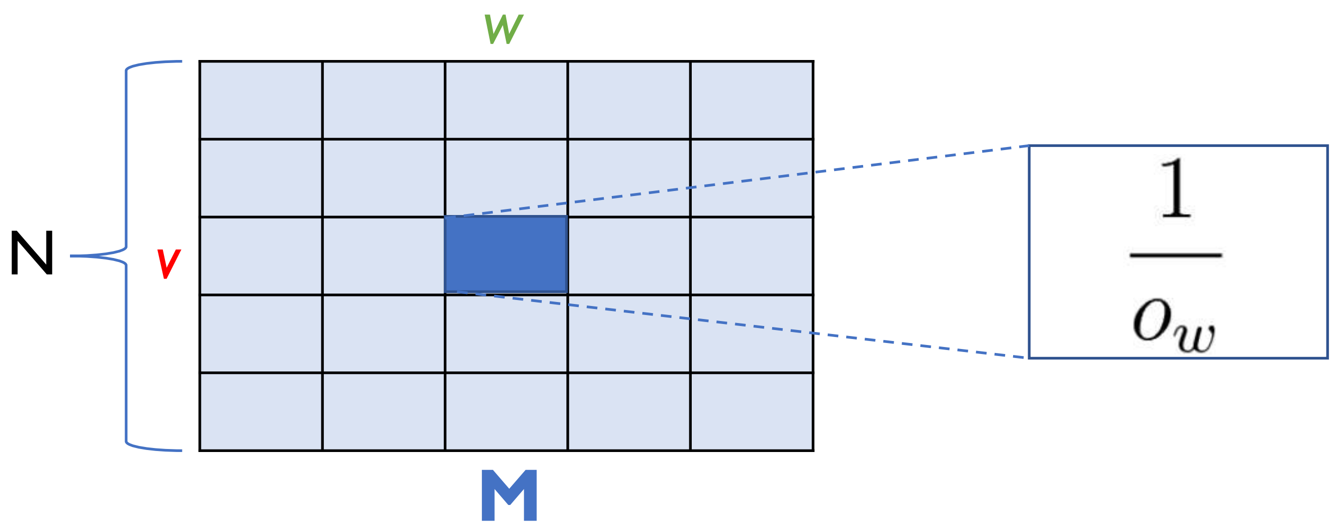
PageRank: The Matrix Formulation

Represent the Web graph of documents G=(V, E) s.t. |V|=N
as a **column stochastic matrix M** of size NxN



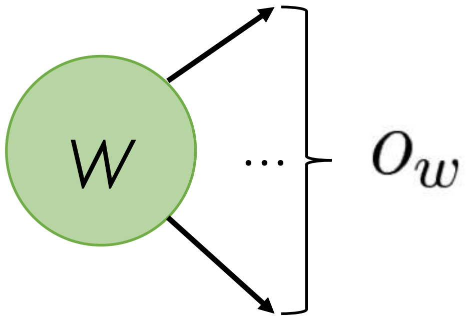


One of those o_w links
points to page *v*

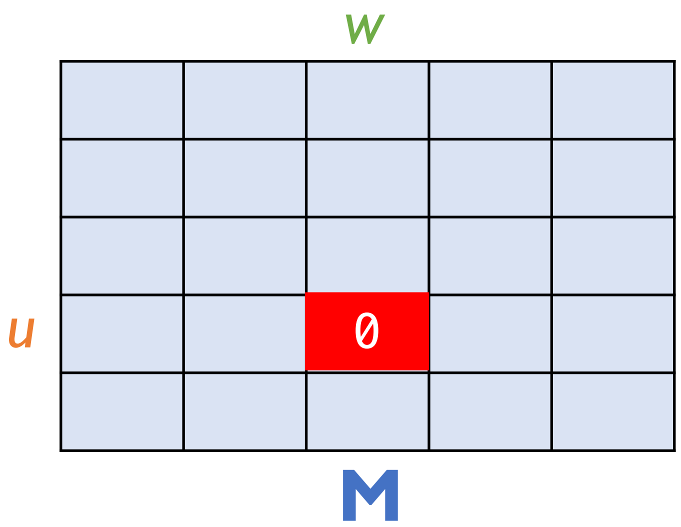


PageRank: The Matrix Formulation

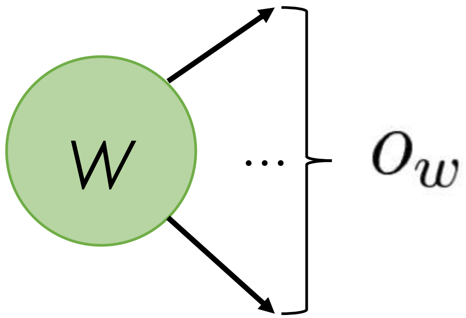
Represent the Web graph of documents G=(V, E) s.t. |V|=N
as a **column stochastic matrix** **M** of size NxN



For any other page *u* which *w* is not pointing to **M**[*u*, *w*] = 0

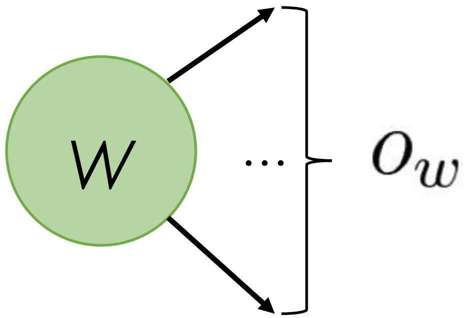


PageRank: The Matrix Formulation

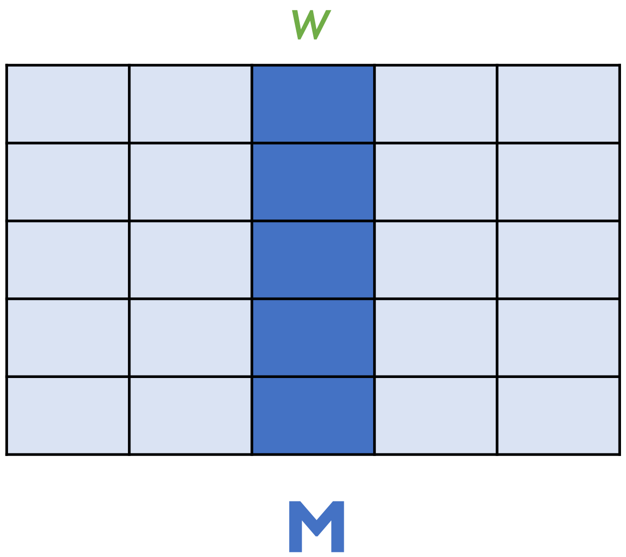


M is **column stochastic** because, by design, each of its **column sums up to 1**

PageRank: The Matrix Formulation



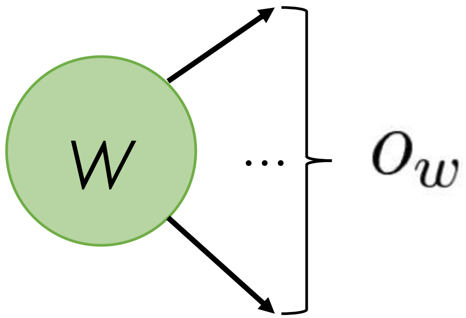
M is **column stochastic** because, by design, each of its **column sums up to 1**



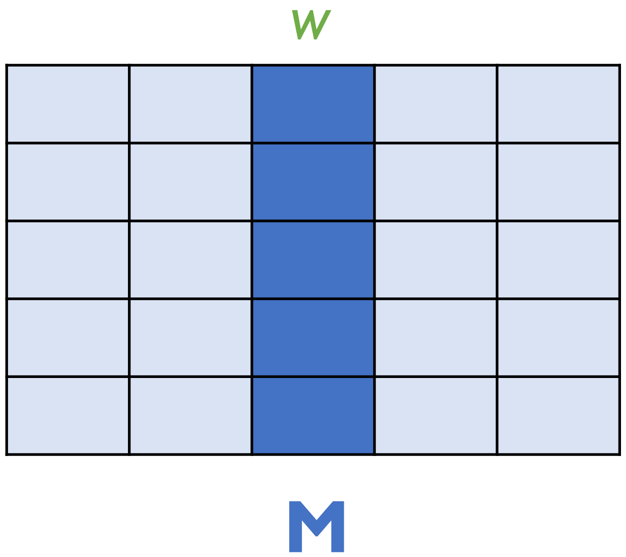
The *w*-th column will contain o_w <= N non-zero entries, each evaluating to $1/o_w$

$$\sum_{v=1}^{N} m_{v,w} = o_w \times \frac{1}{o_w} = 1$$

PageRank: The Matrix Formulation



M is **column stochastic** because, by design, each of its **column sums up to 1**



<u>Note:</u>
We are implicitly assuming there exists **at least one** outgoing link from each node

A Formal View of the Matrix **M**

$$\mathbf{A}_{N\times N} \quad a_{v,w} = \begin{cases} 1 & \text{if } w \in O_v \\ 0 & \text{otherwise} \end{cases}$$

Traditional adjacency matrix

A Formal View of the Matrix **M**

$$\mathbf{A}_{N\times N} \quad a_{v,w} = \begin{cases} 1 & \text{if } w \in O_v \\ 0 & \text{otherwise} \end{cases}$$

Traditional adjacency matrix

$$\mathbf{L}_{N\times N} \quad l_{v,w} = \begin{cases} o_v & \text{if } v = w \\ 0 & \text{otherwise} \end{cases}$$

Diagonal matrix of out-degrees

A Formal View of the Matrix **M**

$$\mathbf{A}_{N\times N} \quad a_{v,w} = \begin{cases} 1 & \text{if } w \in O_v \\ 0 & \text{otherwise} \end{cases}$$

Traditional adjacency matrix

$$\mathbf{L}_{N\times N} \quad l_{v,w} = \begin{cases} o_v & \text{if } v = w \\ 0 & \text{otherwise} \end{cases}$$

Diagonal matrix of out-degrees

$$\mathbf{M}_{N\times N} \quad m_{v,w} = \begin{cases} \frac{1}{o_w} & \text{if } v \in O_w \\ 0 & \text{otherwise} \end{cases}$$

Column stochastic matrix

$$\boxed{\mathbf{M} = (\mathbf{L}^{-1}\mathbf{A})^T}$$

PageRank: The Matrix Formulation

$\mathbf{r}$ Nx1 **rank vector** with an entry for each page

PageRank: The Matrix Formulation

$\mathbf{r}$ Nx1 **rank vector** with an entry for each page

r_v  Rank score of page *v*

PageRank: The Matrix Formulation

$\mathbf{r}$ Nx1 **rank vector** with an entry for each page

r_v

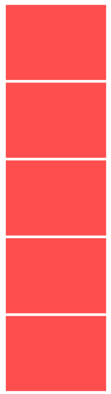
Rank score of page *v*

$$\sum_{v=1}^{N} r_v = 1$$

All the rank scores must sum up to 1

PageRank: The Matrix Formulation

$\mathbf{r}$ Nx1 **rank vector** with an entry for each page

r_v  Rank score of page *v*

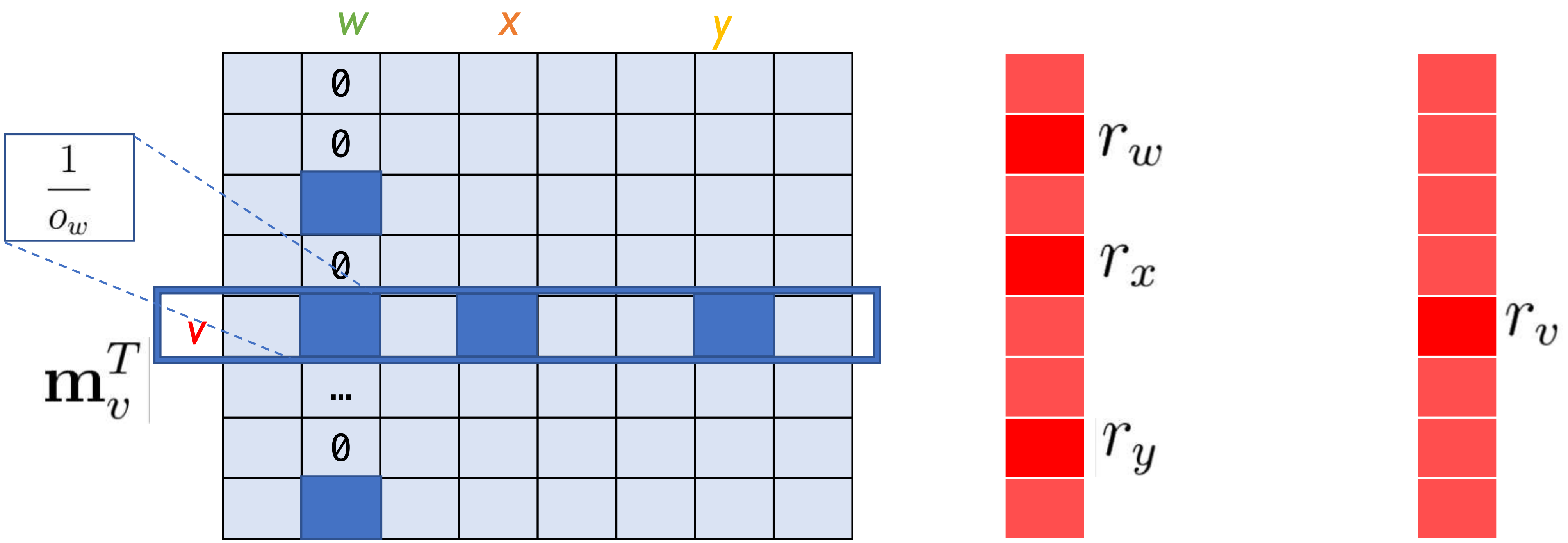
$$\sum_{v=1}^{N} r_v = 1$$

All the rank scores must sum up to 1

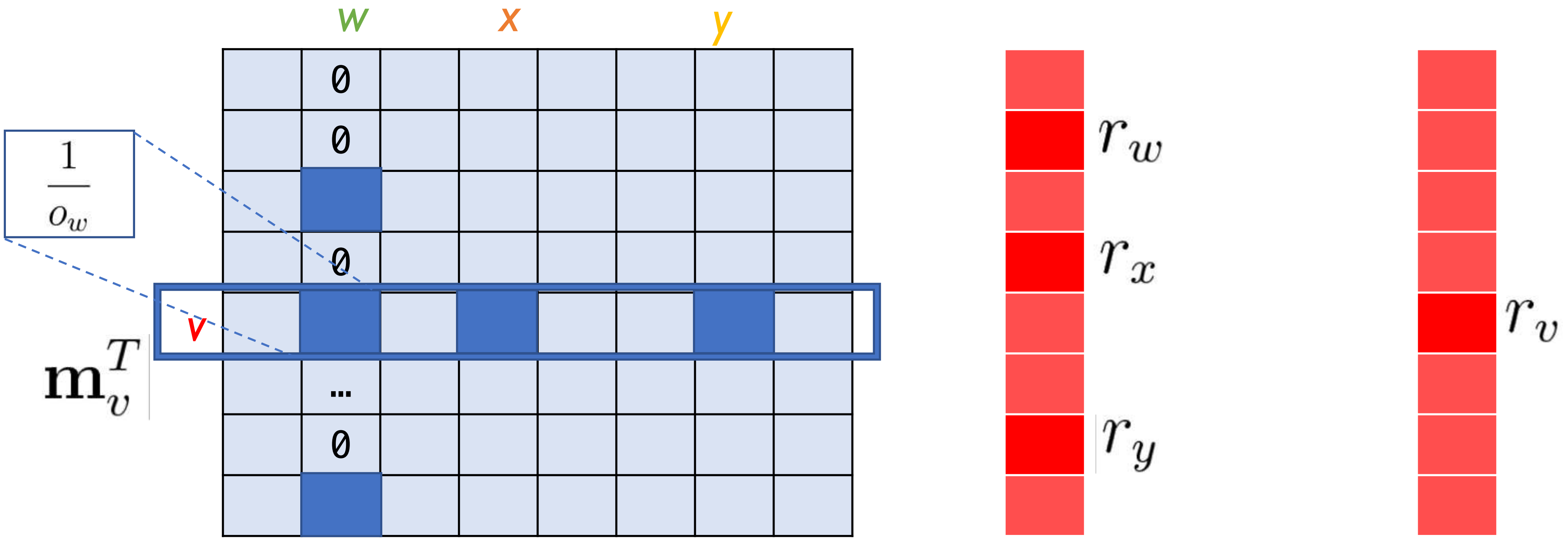
$$r_v = \sum_{w \in I_v} \frac{r_w}{o_w} \quad \Rightarrow \quad \mathbf{r} = \mathbf{M}\mathbf{r}$$

Flow equations in matrix form

PageRank: The Matrix Formulation

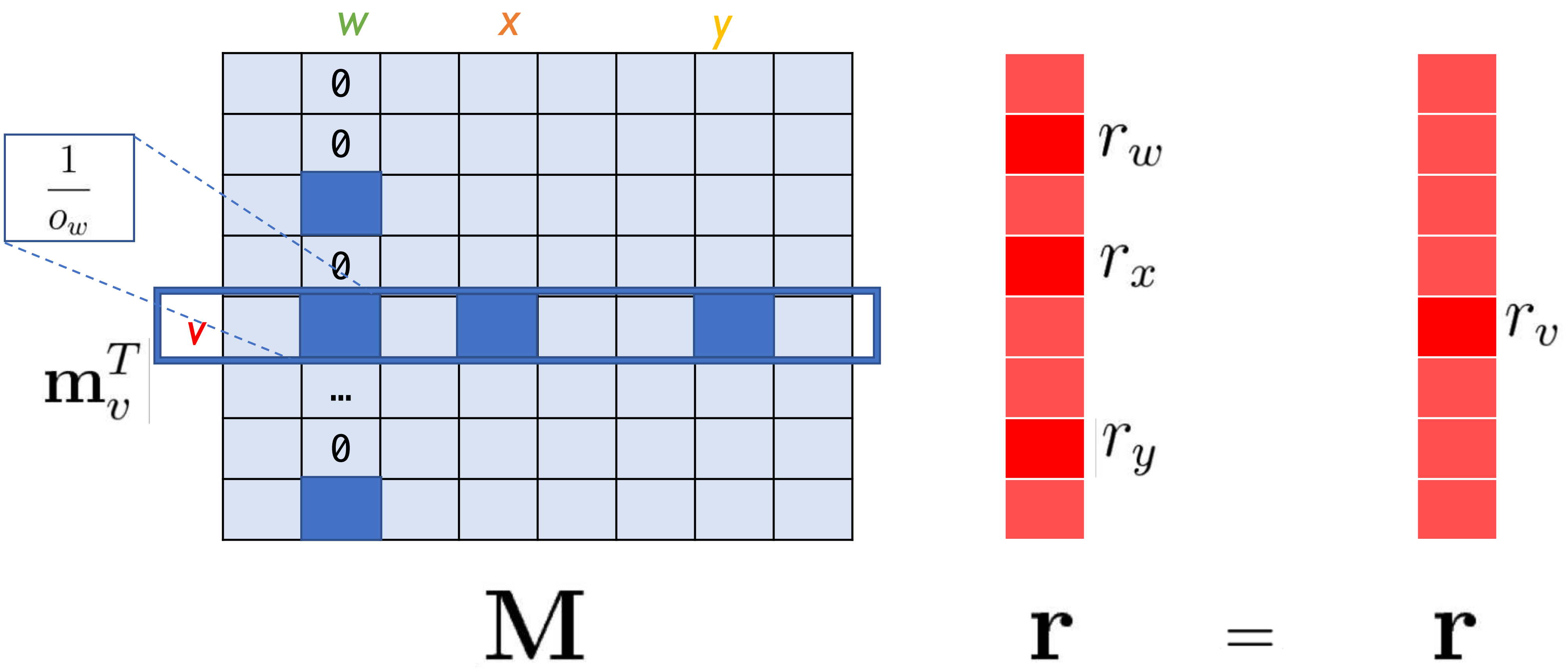


PageRank: The Matrix Formulation



$$r_v = \mathbf{m}_v^T \cdot \mathbf{r} = \sum_{w=1}^{N} m_{v,w} \times r_w = \sum_{w=1}^{N} \frac{1}{o_w} \times r_w = \sum_{w=1}^{N} \frac{r_w}{o_w} = \sum_{w \in I_v} \frac{r_w}{o_w}$$

PageRank: The Matrix Formulation



PageRank: The Eigenvector Formulation

$$\boxed{\mathbf{M}\mathbf{r} = \mathbf{r}}$$

Doesn't it look familiar?

PageRank: The Eigenvector Formulation

$$\boxed{\mathbf{Mr} = \mathbf{r}}$$

Doesn't it look familiar?

$$\mathbf{Ax} = \lambda\mathbf{x}$$

x is an eigenvector

λ is an eigenvalue

PageRank: The Eigenvector Formulation

$$\boxed{\mathbf{Mr} = \mathbf{r}}$$

Doesn't it look familiar?

$$\mathbf{Ax} = \lambda\mathbf{x}$$

x is an eigenvector

λ is an eigenvalue

So, the rank vector **r** is an **eigenvector** of the matrix **M**

PageRank: The Eigenvector Formulation

$$\boxed{\mathbf{Mr} = \mathbf{r}}$$

Doesn't it look familiar?

$$\mathbf{Ax} = \lambda\mathbf{x}$$

x is an eigenvector

λ is an eigenvalue

So, the rank vector **r** is an **eigenvector** of the matrix **M**

In fact, **r** is the eigenvector corresponding to the **eigenvalue** λ = 1

PageRank: The Eigenvector Formulation

$$\boxed{\mathbf{M}\mathbf{r} = \mathbf{r}}$$

For a fixed eigenvalue, eigenvectors are just scalar multiples of each other

PageRank: The Eigenvector Formulation

$$\boxed{\mathbf{Mr} = \mathbf{r}}$$

For a fixed eigenvalue, eigenvectors are just scalar multiples of each other

We can choose **any** of them to be our PageRank vector **r**

PageRank: The Eigenvector Formulation

$$\boxed{\mathbf{Mr} = \mathbf{r}}$$

For a fixed eigenvalue, eigenvectors are just scalar multiples of each other

We can choose **any** of them to be our PageRank vector **r**

Since PageRank should reflect only the relative importance of the nodes, choose $\mathbf{r} = \mathbf{r}^*$ as the eigenvector whose entries sum up to 1

PageRank: The Eigenvector Formulation

$$\boxed{\mathbf{Mr} = \mathbf{r}}$$

For a fixed eigenvalue, eigenvectors are just scalar multiples of each other

We can choose **any** of them to be our PageRank vector **r**

Since PageRank should reflect only the relative importance of the nodes, choose $\mathbf{r} = \mathbf{r}^*$ as the eigenvector whose entries sum up to 1

This may be referred to as the **probabilistic eigenvector** corresponding to the eigenvalue $\lambda = 1$

PageRank: The Eigenvector Formulation

$$\boxed{\mathbf{Mr} = \mathbf{r}}$$

We know from linear algebra theory that for any **stochastic** matrix **M** its **largest eigenvalue** is $\lambda = 1$

PageRank: The Eigenvector Formulation

$$\boxed{\mathbf{Mr} = \mathbf{r}}$$

We know from linear algebra theory that for any **stochastic** matrix **M** its **largest eigenvalue** is $\lambda = 1$

Therefore, $\mathbf{r} = \mathbf{r}^*$ is the **principal eigenvector** of **M** (i.e., the eigenvector associated with the largetst eigenvalue)

PageRank: The Eigenvector Formulation

$$\boxed{\mathbf{Mr} = \mathbf{r}}$$

We know from linear algebra theory that for any **stochastic** matrix **M** its **largest eigenvalue** is $\lambda = 1$

Therefore, $\mathbf{r} = \mathbf{r}^*$ is the **principal eigenvector** of **M** (i.e., the eigenvector associated with the largetst eigenvalue)

Note:

So far, we have assumed that **M** is (column) stochastic yet this may not be the case for the general Web graph…

PageRank: Quick Recap

We start from "flow" equations

PageRank: Quick Recap

We start from "flow" equations

We reformulate the system of linear equations using linear algebra
(i.e., stochastic matrix $\mathbf{M}$ and rank vector $\mathbf{r}$)

PageRank: Quick Recap

We start from "flow" equations

We reformulate the system of linear equations using linear algebra (i.e., stochastic matrix **M** and rank vector **r**)

We reduce the above to finding the **eigenvector** of the matrix **M**

PageRank: Quick Recap

We start from "flow" equations

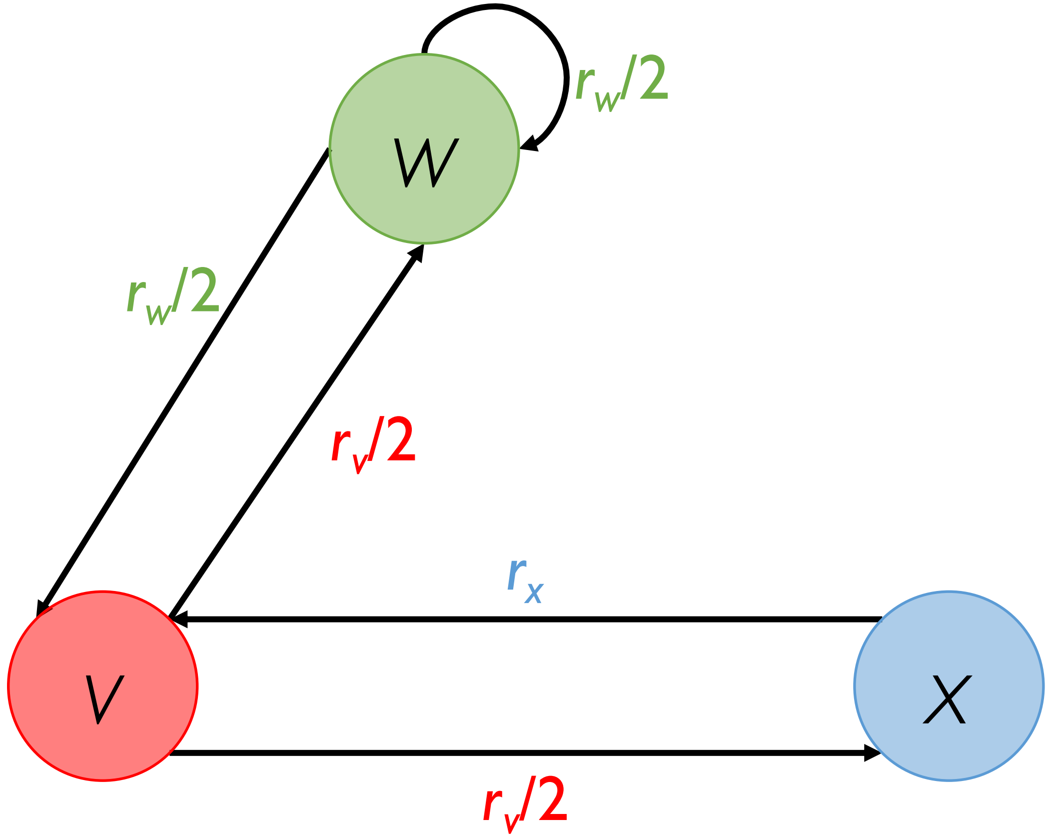
We reformulate the system of linear equations using linear algebra
(i.e., stochastic matrix **M** and rank vector **r**)

We reduce the above to finding the **eigenvector** of the matrix **M**

⇩

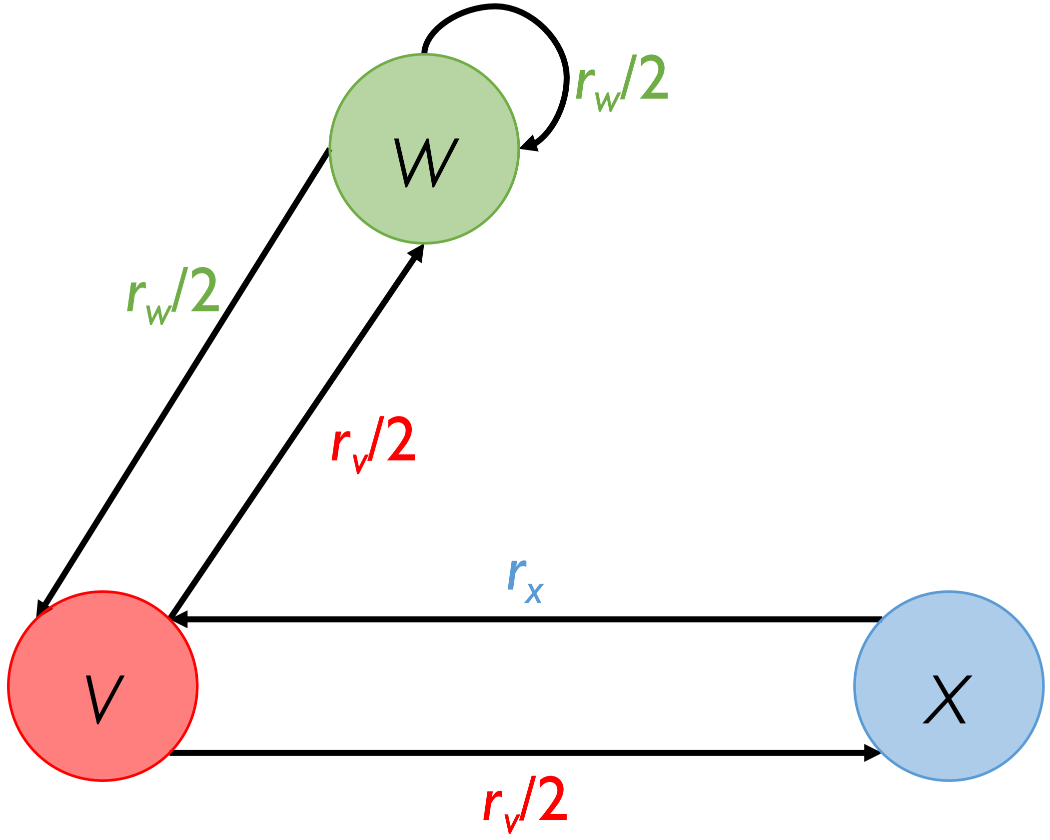
We know how to solve this efficiently using **power iteration** method

PageRank: The "Flow" Model



$$\begin{cases} r_v = r_w/2 + r_x \\ r_w = r_v/2 + r_w/2 \\ r_x = r_v/2 \end{cases}$$

PageRank: The "Flow" Model



$$
\begin{cases}
r_v = r_w/2 + r_x \\
r_w = r_v/2 + r_w/2 \\
r_x = r_v/2
\end{cases}
$$

0	1/2	1
1/2	1/2	0
1/2	0	0

$$\mathbf{r} = \mathbf{M} \quad \mathbf{r}$$

PageRank: Power Iteration Method

At the beginning, we assume all pages have the same rank score, **uniformly distributed** across the N pages

$$\textbf{init:}\ \ t = 0; \mathbf{r}(t) = (1/N, 1/N, \ldots, 1/N)^T$$

PageRank: Power Iteration Method

Keep updating the rank vector **r** **until convergence**

init: $t = 0; \mathbf{r}(t) = (1/N, 1/N, \ldots, 1/N)^T$

repeat:

$$\mathbf{r}(t+1) = \mathbf{M}\mathbf{r}(t)$$

until $\delta(\mathbf{r}(t+1), \mathbf{r}(t)) < \epsilon$

$\epsilon > 0$

PageRank: Power Iteration Method

$$\textbf{init: } t = 0; \mathbf{r}(t) = (1/N, 1/N, \ldots, 1/N)^T$$

repeat:

$$\mathbf{r}(t+1) = \mathbf{M}\mathbf{r}(t)$$

$$\textbf{until } \delta(\mathbf{r}(t+1), \mathbf{r}(t)) < \epsilon$$

$$\epsilon > 0$$

$$\left\{ \begin{array}{c} \delta(\mathbf{r}(t+1), \mathbf{r}(t)) = |\mathbf{r}(t+1) - \mathbf{r}(t)| \\ \text{or} \\ \delta(\mathbf{r}(t+1), \mathbf{r}(t)) = ||\mathbf{r}(t+1) - \mathbf{r}(t)|| \end{array} \right.$$

Power Iteration Method: Example

$$\begin{cases} r_v = r_w/2 + r_x \\ r_w = r_v/2 + r_w/2 \\ r_x = r_v/2 \end{cases}$$

$$\mathbf{r} = \mathbf{M}\,\mathbf{r}, \qquad \mathbf{M} = \begin{bmatrix} 0 & 1/2 & 1 \\ 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \end{bmatrix}$$

Power Iteration Method: Example

$$\begin{cases} r_v = r_w/2 + r_x \\ r_w = r_v/2 + r_w/2 \\ r_x = r_v/2 \end{cases}$$

$$\mathbf{r} = \mathbf{M} \, \mathbf{r}$$

0	1/2	1
1/2	1/2	0
1/2	0	0

$$\mathbf{r}(0) = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$$

Power Iteration Method: Example

$$\begin{cases} r_v = r_w/2 + r_x \\ r_w = r_v/2 + r_w/2 \\ r_x = r_v/2 \end{cases}$$

$$\mathbf{r} = \mathbf{M}\,\mathbf{r}, \quad \mathbf{M} = \begin{bmatrix} 0 & 1/2 & 1 \\ 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \end{bmatrix}$$

$$\underset{\mathbf{r}(0)}{\begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}} \qquad \underset{\mathbf{r}(1) = \mathbf{M}\mathbf{r}(0)}{\begin{bmatrix} 3/6 \\ 1/3 \\ 1/6 \end{bmatrix}}$$

Power Iteration Method: Example

$$\begin{cases} r_v = r_w/2 + r_x \\ r_w = r_v/2 + r_w/2 \\ r_x = r_v/2 \end{cases}$$

$$\underset{\mathbf{r}}{\begin{bmatrix} \, \\ \, \\ \, \end{bmatrix}} = \underset{\mathbf{M}}{\begin{bmatrix} 0 & 1/2 & 1 \\ 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \end{bmatrix}} \underset{\mathbf{r}}{\begin{bmatrix} \, \\ \, \\ \, \end{bmatrix}}$$

$$\underset{\mathbf{r}(0)}{\begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}} \qquad \underset{\mathbf{r}(1) = \mathbf{M}\mathbf{r}(0)}{\begin{bmatrix} 3/6 \\ 1/3 \\ 1/6 \end{bmatrix}} \qquad \underset{\mathbf{r}(2) = \mathbf{M}\mathbf{r}(1)}{\begin{bmatrix} 1/3 \\ 5/12 \\ 3/12 \end{bmatrix}}$$

Power Iteration Method: Example

$$\begin{cases} r_v = r_w/2 + r_x \\ r_w = r_v/2 + r_w/2 \\ r_x = r_v/2 \end{cases}$$

$$\mathbf{r} = \mathbf{M}\mathbf{r}, \quad \mathbf{M} = \begin{bmatrix} 0 & 1/2 & 1 \\ 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \end{bmatrix}$$

$$\underset{\mathbf{r}(0)}{\begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}} \quad \underset{\mathbf{r}(1) = \mathbf{M}\mathbf{r}(0)}{\begin{bmatrix} 3/6 \\ 1/3 \\ 1/6 \end{bmatrix}} \quad \underset{\mathbf{r}(2) = \mathbf{M}\mathbf{r}(1)}{\begin{bmatrix} 1/3 \\ 5/12 \\ 3/12 \end{bmatrix}} \quad \ldots \quad \underset{\ldots\ \mathbf{r}(t+1) = \mathbf{M}\mathbf{r}(t)}{\begin{bmatrix} 6/15 \\ 6/15 \\ 3/15 \end{bmatrix}} \quad \begin{matrix} 2/5 \\ 2/5 \\ 1/5 \end{matrix}$$

Power Iteration Method: Example

$$\begin{cases} r_v = r_w/2 + r_x \\ r_w = r_v/2 + r_w/2 \\ r_x = r_v/2 \end{cases}$$

$$\underset{\mathbf{r}}{\begin{bmatrix} \\ \\ \\ \end{bmatrix}} = \underset{\mathbf{M}}{\begin{bmatrix} 0 & 1/2 & 1 \\ 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \end{bmatrix}} \underset{\mathbf{r}}{\begin{bmatrix} \\ \\ \\ \end{bmatrix}}$$

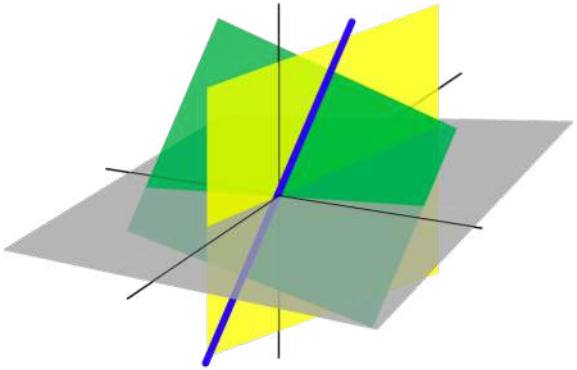
$$\underset{\mathbf{r}(0)}{\begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}} \quad \underset{\mathbf{r}(1) = \mathbf{M}\mathbf{r}(0)}{\begin{bmatrix} 3/6 \\ 1/3 \\ 1/6 \end{bmatrix}} \quad \underset{\mathbf{r}(2) = \mathbf{M}\mathbf{r}(1)}{\begin{bmatrix} 1/3 \\ 5/12 \\ 3/12 \end{bmatrix}} \quad \ldots \quad \ldots \quad \underset{\mathbf{r}(t+1) = \mathbf{M}\mathbf{r}(t)}{\begin{bmatrix} 6/15 \\ 6/15 \\ 3/15 \end{bmatrix}} \quad \begin{matrix} 2/5 \\ 2/5 \\ 1/5 \end{matrix}$$

We came up with the same set of solutions for r_v, r_w, and r_x without explicitly solving the system of equations

PageRank's Interpretations

2 main perspectives

Linear Algebra



Probabilistic

Random Walk Interpretation of Page Rank

Imagine a **random surfer** navigating through the pages of the Web graph

Random Walk Interpretation of Page Rank

Initially, at time $t=0$ the surfer can be on **any** web page

www.donuts.com

www.krustyburger.com

www.duffbeer.com

…

www.moes.com

Random Walk Interpretation of Page Rank

Initially, at time $t=0$ the surfer can be on **any** web page

www.donuts.com

www.krustyburger.com

www.duffbeer.com

...

www.moes.com

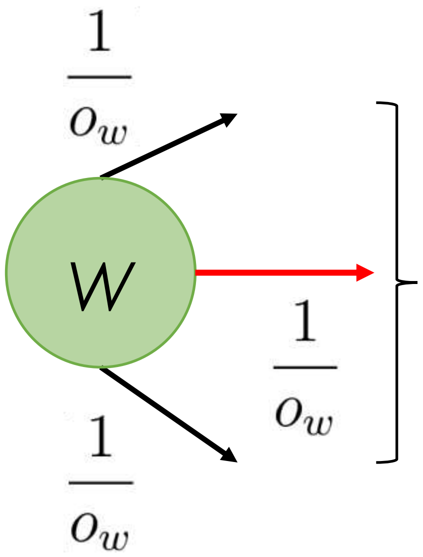
Each web page has **equal probability** 1/N to be chosen as starting point

Random Walk Interpretation of Page Rank

At any given time t, the surfer is on some web page w

Random Walk Interpretation of Page Rank

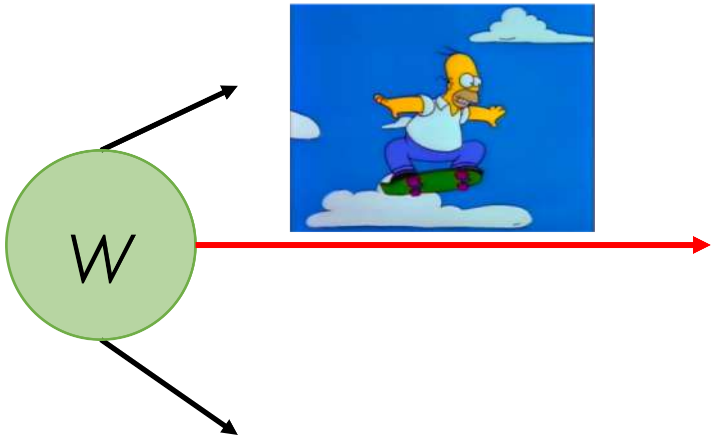
At time *t+1*, the surfer follows one of the outgoing links from web page *w*, chosen **uniformly at random**



o_w out-degree of node *w*

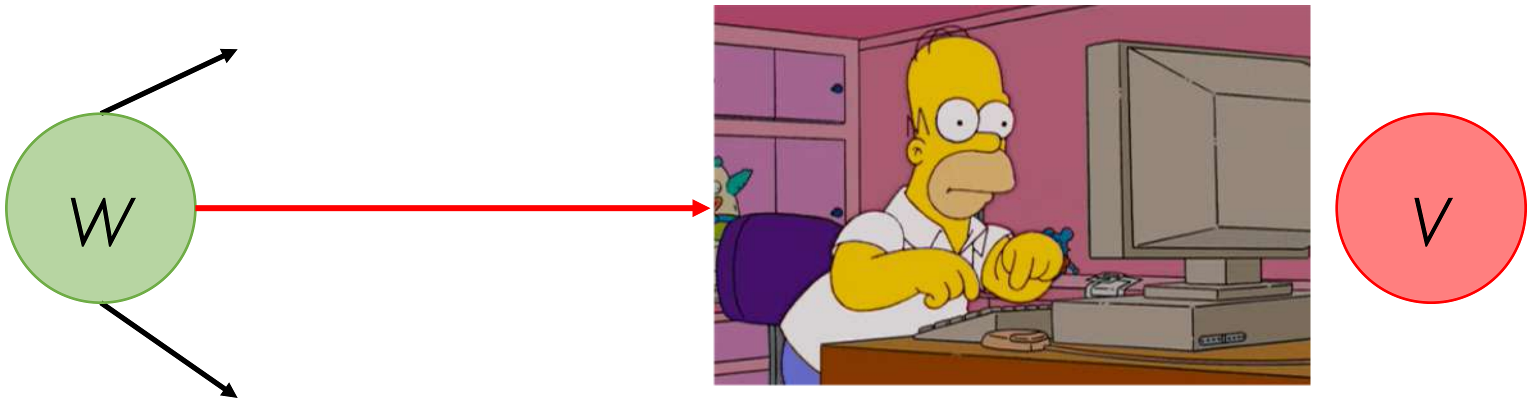
Random Walk Interpretation of Page Rank

The surfer ends up into some other web page *v* pointed by *w*



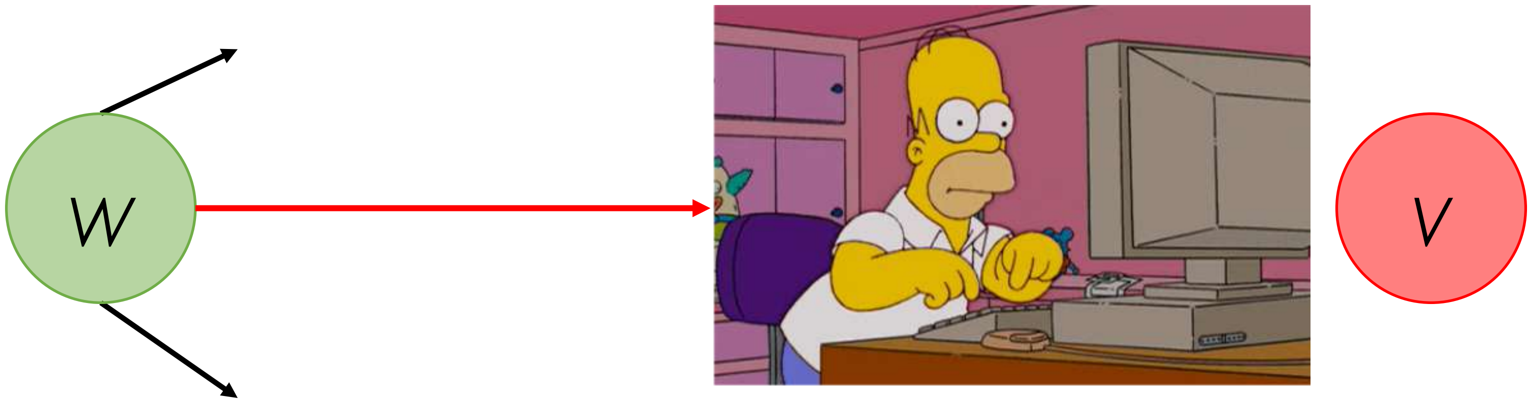
Random Walk Interpretation of Page Rank

The surfer ends up into some other web page *v* pointed by *w*



Random Walk Interpretation of Page Rank

The surfer ends up into some other web page *v* pointed by *w*



This process repeats indefinitely and is known as **random walk**

Transition Matrix **M**

$$\mathbf{M}_{N\times N} \quad m_{v,w} = \begin{cases} \frac{1}{o_w} & \text{if } v \in O_w \\ 0 & \text{otherwise} \end{cases}$$

Column stochastic matrix

Transition Matrix **M**

$$\mathbf{M}_{N\times N} \quad m_{v,w} = \begin{cases} \frac{1}{o_w} & \text{if } v \in O_w \\ 0 & \text{otherwise} \end{cases}$$

Column stochastic matrix

The *v*-th, *w*-th entry of **M** indicates the probability of a random surfer moving from page *w* to page *v*

Transition Matrix **M**

$$\mathbf{M}_{N\times N} \quad m_{v,w} = \begin{cases} \frac{1}{o_w} & \text{if } v \in O_w \\ 0 & \text{otherwise} \end{cases}$$

Column stochastic matrix

The *v*-th, *w*-th entry of **M** indicates the probability of a random surfer moving from page *w* to page *v*

Such a matrix describes a **Markov chain** over the finite state space V of nodes (i.e., pages) of the Web graph

Random Walk Interpretation of Page Rank

X Discrete-Valued Random Variable taking on |V| = N possible values

Random Walk Interpretation of Page Rank

X Discrete-Valued Random Variable taking on |V| = N possible values

$X = w$ Indicates a random surfer is on web page *w*

Random Walk Interpretation of Page Rank

X Discrete-Valued Random Variable taking on |V| = N possible values

$X = w$ Indicates a random surfer is on web page *w*

N-dimensional stochastic (i.e., probability) vector associated with X

$$\mathbf{p} \subseteq \mathbb{R}^N = (P(X=1), \ldots, P(X=w), \ldots, P(X=N))^T$$

Random Walk Interpretation of Page Rank

X Discrete-Valued Random Variable taking on |V| = N possible values

$X = w$ Indicates a random surfer is on web page *w*

N-dimensional stochastic (i.e., probability) vector associated with X

$$\mathbf{p} \subseteq \mathbb{R}^N = (P(X=1), \ldots, P(X=w), \ldots, P(X=N))^T$$

N-dimensional stochastic (i.e., probability) vector associated with X at time *t*

$$\mathbf{p}(t) \subseteq \mathbb{R}^N = (P(X_t=1), \ldots, P(X_t=w), \ldots, P(X_t=N))^T$$

Random Walk Interpretation of Page Rank

X Discrete-Valued Random Variable taking on |V| = N possible values

$X = w$ Indicates a random surfer is on web page *w*

N-dimensional stochastic (i.e., probability) vector associated with X

$$\mathbf{p} \subseteq \mathbb{R}^N = (P(X=1), \ldots, P(X=w), \ldots, P(X=N))^T$$

N-dimensional stochastic (i.e., probability) vector associated with X at time *t*

$$\mathbf{p}(t) \subseteq \mathbb{R}^N = (P(X_t=1), \ldots, P(X_t=w), \ldots, P(X_t=N))^T$$

Probability distribution over web pages at time *t*

Random Walks as Markov Chains

Random Walks are also known as **stochastic processes** with **Markov property** (i.e., **Markov chains**)

Random Walks as Markov Chains

Random Walks are also known as **stochastic processes** with **Markov property** (i.e., **Markov chains**)

The **transition probability** of moving to the next state **depends** only on the **present state** and not on the previous states

$$P(X_{t+1} = v | X_1 = x_1, X_2 = x_2, \ldots, X_t = x_t) = P(X_{t+1} = v | X_t = x_t)$$

Random Walks as Markov Chains

Random Walks are also known as **stochastic processes** with **Markov property** (i.e., **Markov chains**)

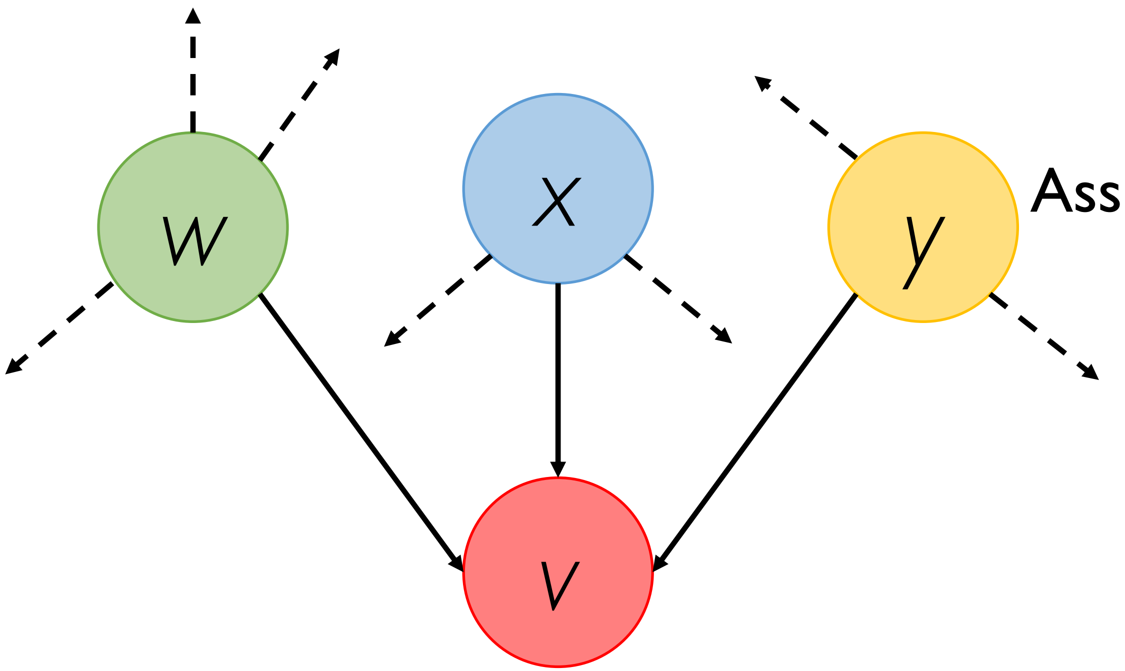
The **transition probability** of moving to the next state **depends** only on the **present state** and not on the previous states

$$P(X_{t+1} = v | X_1 = x_1, X_2 = x_2, \ldots, X_t = x_t) = P(X_{t+1} = v | X_t = x_t)$$

The probability that the random surfer will be on page *v* at time *t+1* depends only on where the surfer was at time *t*

Random Walk Interpretation of Page Rank

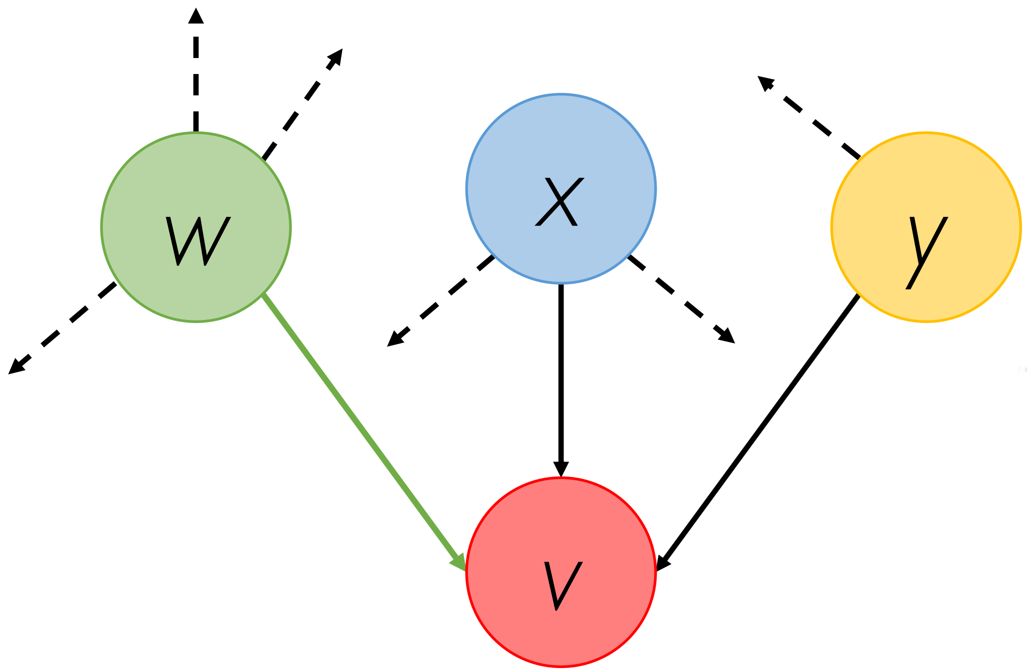
Where is the random surfer at time $t+1$ knowing where he was at time t?
Suppose we want to estimate $P(X_{t+1} = v)$



Assume *v* has only **3 incoming links** from *w*, *x*, and *y*

Random Walk Interpretation of Page Rank

Where is the random surfer at time *t+1* knowing where he was at time *t*?
Suppose we want to estimate $P(X_{t+1} = v)$

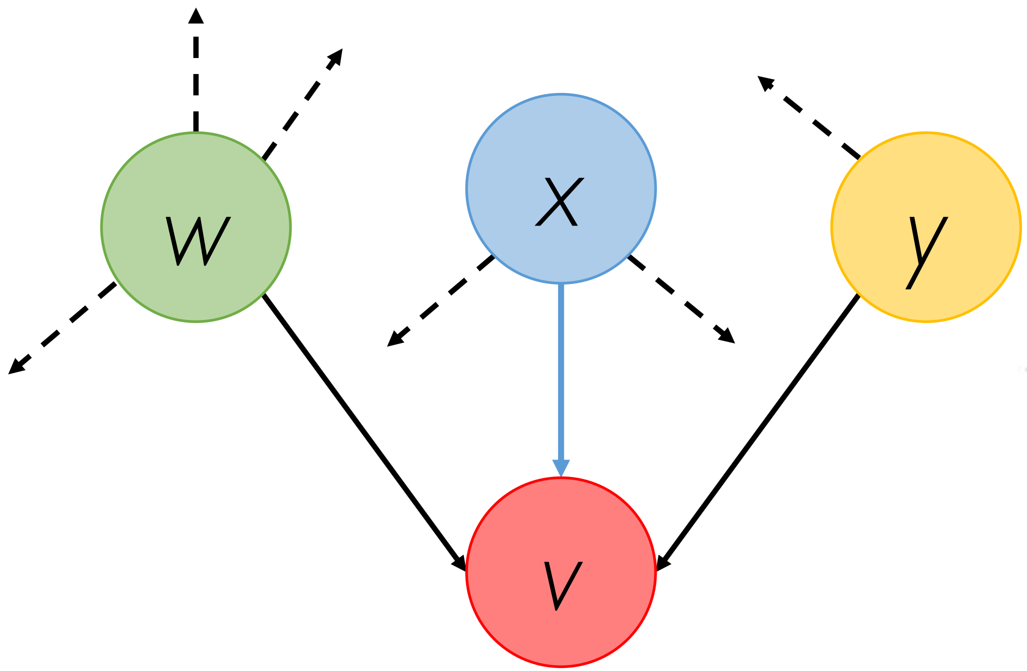


Assume *v* has only **3 incoming links** from *w*, *x*, and *y*

$$P(X_{t+1} = v) = \boxed{P(X_t = w, Z_w = v) +}$$

Random Walk Interpretation of Page Rank

Where is the random surfer at time $t+1$ knowing where he was at time t?
Suppose we want to estimate $P(X_{t+1} = v)$

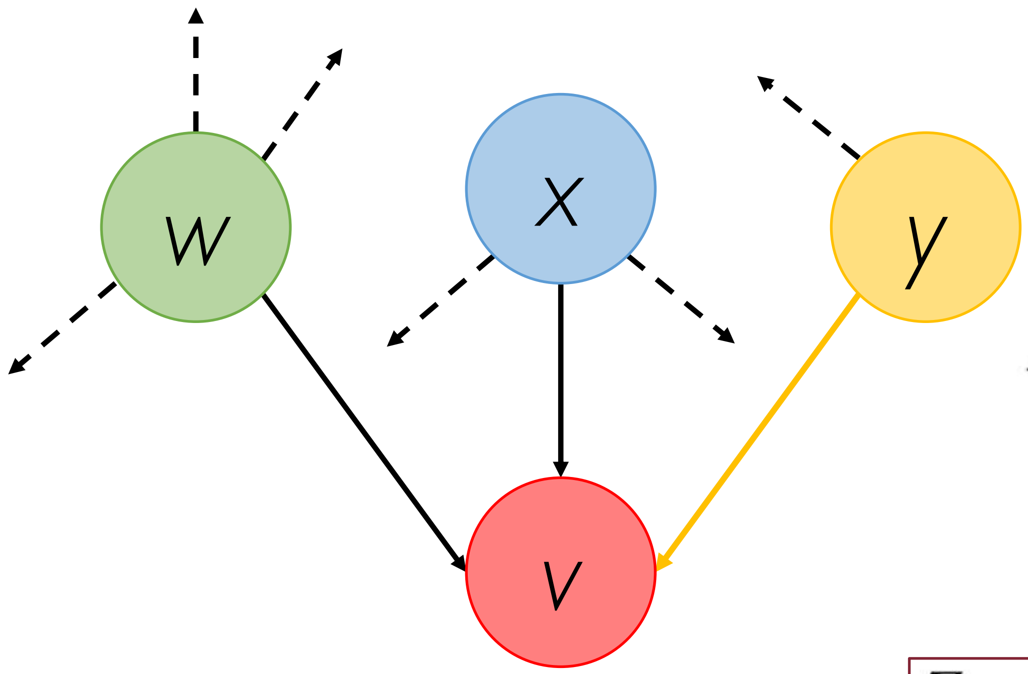


Assume v has only **3 incoming links** from w, x, and y

$$P(X_{t+1} = v) = P(X_t = w, Z_w = v) +$$
$$\boxed{P(X_t = x, Z_x = v) +}$$

Random Walk Interpretation of Page Rank

Where is the random surfer at time *t+1* knowing where he was at time *t*?
Suppose we want to estimate $P(X_{t+1} = v)$

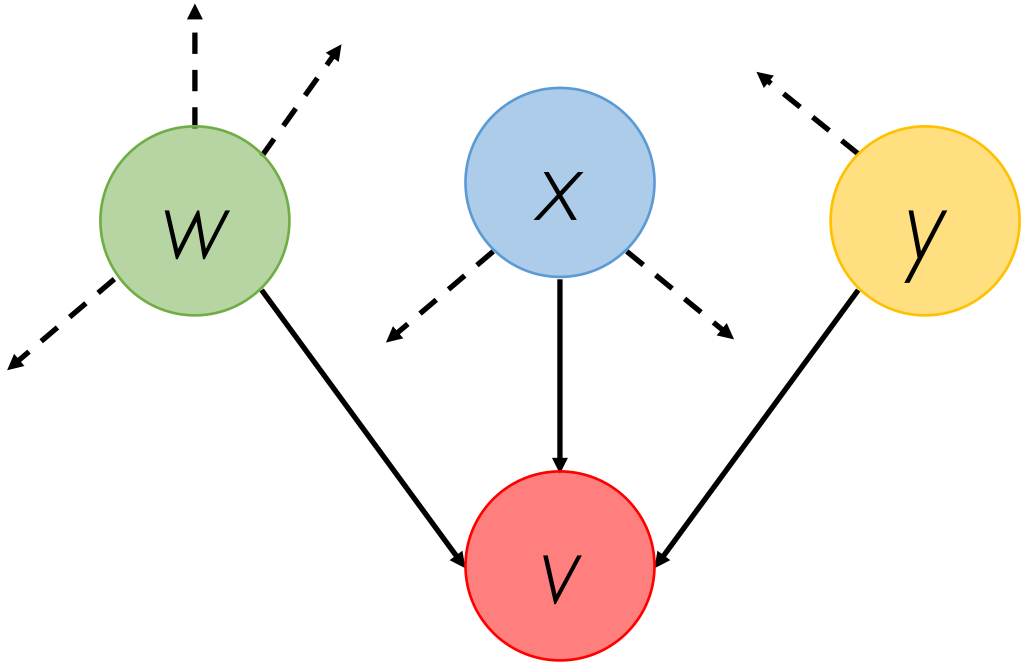


Assume *v* has only **3 incoming links** from *w*, *x*, and *y*

$$P(X_{t+1} = v) = P(X_t = w, Z_w = v) +$$
$$P(X_t = x, Z_x = v) +$$
$$\boxed{P(X_t = y, Z_y = v)}$$

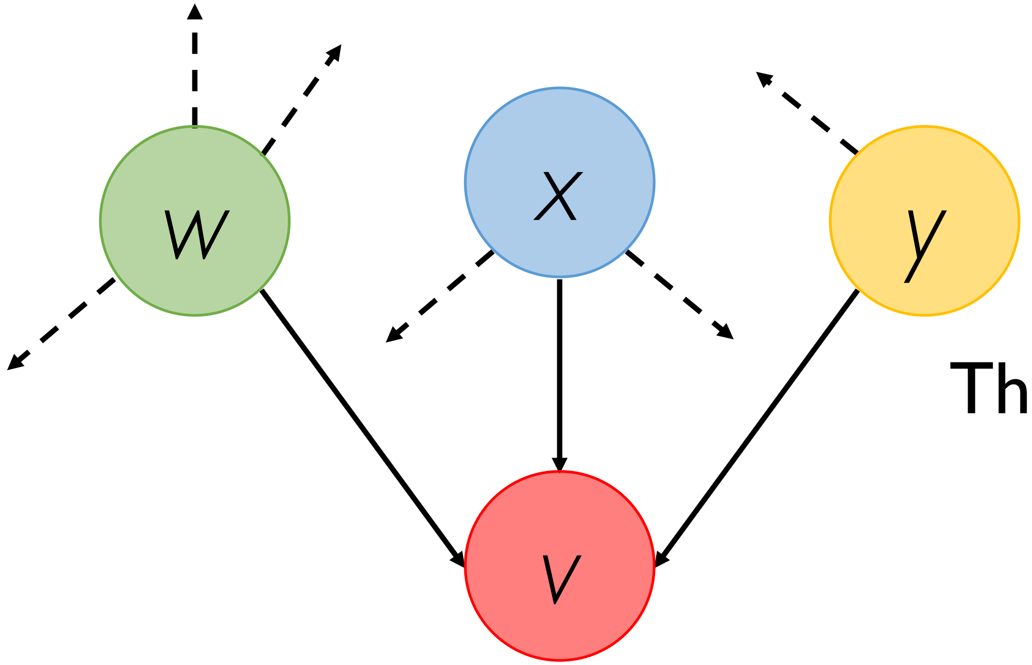
$$\boxed{Z_u \sim \text{Uniform}(1, o_u)}$$

Random Walk Interpretation of Page Rank



$$\boxed{\mathbf{p}(t+1) = \mathbf{M}\mathbf{p}(t)}$$

Random Walk Interpretation of Page Rank

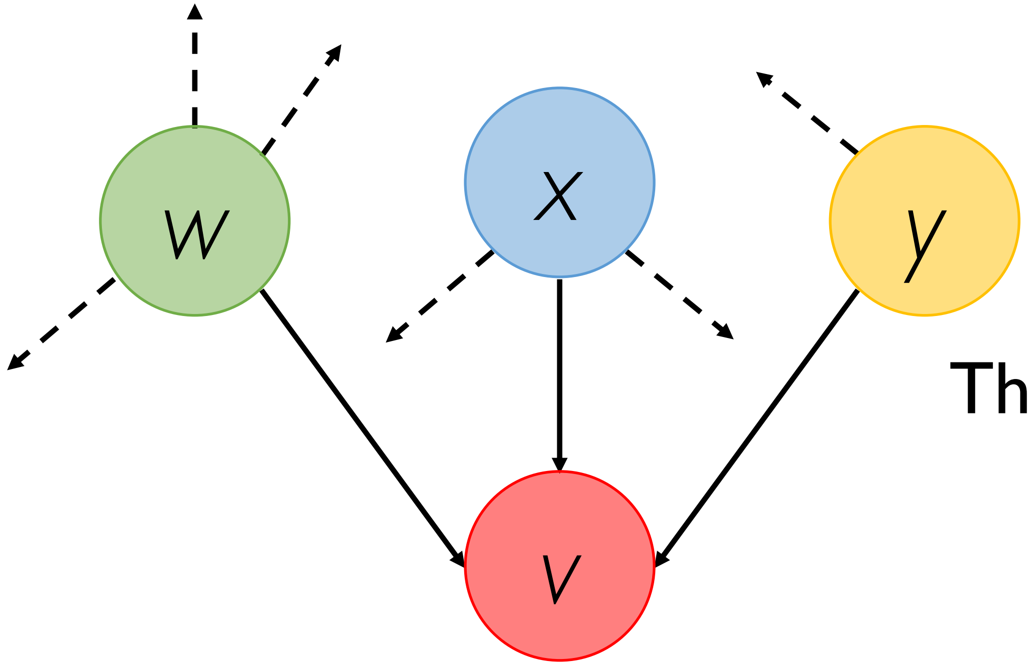


$$\boxed{\mathbf{p}(t+1) = \mathbf{M}\mathbf{p}(t)}$$

This resembles our PageRank equation

$$\boxed{\mathbf{r}(t+1) = \mathbf{M}\mathbf{r}(t)}$$

Random Walk Interpretation of Page Rank



$$\boxed{\mathbf{p}(t+1) = \mathbf{M}\mathbf{p}(t)}$$

This resembles our PageRank equation

$$\boxed{\mathbf{r}(t+1) = \mathbf{M}\mathbf{r}(t)}$$

Solving the former is equivalent to solving the latter!

Random Walk Interpretation of Page Rank

Initially, the stochastic vector **p**(0) is a **uniform probability distribution**

Random Walk Interpretation of Page Rank

Initially, the stochastic vector **p**(0) is a **uniform probability distribution**

The probability that page *i* will be visited after one step corresponds to the *i*-th entry of **p**(1), obtained as follows:

$$\mathbf{p}(1) = \mathbf{M}\mathbf{p}(0)$$

Random Walk Interpretation of Page Rank

Initially, the stochastic vector **p**(0) is a **uniform probability distribution**

The probability that page *i* will be visited after one step corresponds to the *i*-th entry of **p**(1), obtained as follows:

$$\mathbf{p}(1) = \mathbf{M}\mathbf{p}(0)$$

More generally, the probability of visiting *any* web page after *t* steps is:

$$\boxed{\mathbf{p}(t) = \mathbf{M}^t\mathbf{p}(0)}$$

Random Walk Interpretation of Page Rank

$$\mathbf{p}(0) = (\underbrace{1/N}_{P(X_0=1)}, \ldots, \underbrace{1/N}_{P(X_0=w)}, \ldots, \underbrace{1/N}_{P(X_0=N)})^T$$

Random Walk Interpretation of Page Rank

$$\mathbf{p}(0) = (\underbrace{1/N}_{P(X_0=1)}, \ldots, \underbrace{1/N}_{P(X_0=w)}, \ldots, \underbrace{1/N}_{P(X_0=N)})^T$$

$$\mathbf{p}(1) = \mathbf{M}\mathbf{p}(0)$$

Random Walk Interpretation of Page Rank

$$\mathbf{p}(0) = (\underbrace{1/N}_{P(X_0=1)} , \ldots , \underbrace{1/N}_{P(X_0=w)} , \ldots , \underbrace{1/N}_{P(X_0=N)})^T$$

$$\mathbf{p}(1) = \mathbf{M}\mathbf{p}(0)$$

$$\mathbf{p}(2) = \mathbf{M}\mathbf{p}(1) = \underbrace{\mathbf{M} \times \mathbf{M}}_{\mathbf{M}^2} \mathbf{p}(0)$$

Random Walk Interpretation of Page Rank

$$\mathbf{p}(0) = (\underbrace{1/N}_{P(X_0=1)}, \ldots, \underbrace{1/N}_{P(X_0=w)}, \ldots, \underbrace{1/N}_{P(X_0=N)})^T$$

$$\mathbf{p}(1) = \mathbf{M}\mathbf{p}(0)$$

$$\mathbf{p}(2) = \mathbf{M}\mathbf{p}(1) = \underbrace{\mathbf{M} \times \mathbf{M}}_{\mathbf{M}^2}\mathbf{p}(0)$$

$$\vdots$$

$$\mathbf{p}(k) = \mathbf{M}\mathbf{p}(k-1) = \underbrace{\mathbf{M} \times \mathbf{M} \times \ldots \times \mathbf{M}}_{\mathbf{M}^k}\mathbf{p}(0)$$

$$\vdots$$

Random Walk Interpretation of Page Rank

$\{\mathbf{p}(t)\}_{t=0,1,\ldots,T}$

Discrete Stochastic Process

Markov chain

$$\mathbf{p}(0) = (\underbrace{1/N}_{P(X_0=1)}, \ldots, \underbrace{1/N}_{P(X_0=w)}, \ldots, \underbrace{1/N}_{P(X_0=N)})^T$$

$$\mathbf{p}(1) = \mathbf{M}\mathbf{p}(0)$$

$$\mathbf{p}(2) = \mathbf{M}\mathbf{p}(1) = \underbrace{\mathbf{M} \times \mathbf{M}}_{\mathbf{M}^2}\mathbf{p}(0)$$

$$\vdots$$

$$\mathbf{p}(k) = \mathbf{M}\mathbf{p}(k-1) = \underbrace{\mathbf{M} \times \mathbf{M} \times \ldots \times \mathbf{M}}_{\mathbf{M}^k}\mathbf{p}(0)$$

$$\vdots$$

The Stationary Distribution

Suppose that our random surfer reaches a so-called **steady state**

The Stationary Distribution

Suppose that our random surfer reaches a so-called **steady state**

A steady state indicates a situation where the stochastic vector $\mathbf{p}^*$ does not change anymore

$$\mathbf{p}(t+1) = \mathbf{M}\mathbf{p}(t) = \underbrace{\mathbf{p}(t)}_{\mathbf{p}^*}$$

The Stationary Distribution

Suppose that our random surfer reaches a so-called **steady state**

A steady state indicates a situation where the stochastic vector $\mathbf{p}^*$ does not change anymore

$$\mathbf{p}(t+1) = \mathbf{M}\mathbf{p}(t) = \underbrace{\mathbf{p}(t)}_{\mathbf{p}^*}$$

The sequence coverges to $\mathbf{p}^*$: $\mathbf{M}\mathbf{p}(0), \mathbf{M}^2\mathbf{p}(0), \ldots, \mathbf{M}^t\mathbf{p}(0) \rightsquigarrow \mathbf{p}^*$

The Stationary Distribution

Suppose that our random surfer reaches a so-called **steady state**

A steady state indicates a situation where the stochastic vector $\mathbf{p}^*$ does not change anymore

$$\mathbf{p}(t+1) = \mathbf{M}\mathbf{p}(t) = \underbrace{\mathbf{p}(t)}_{\mathbf{p}^*}$$

The sequence coverges to $\mathbf{p}^*$: $\mathbf{M}\mathbf{p}(0), \mathbf{M}^2\mathbf{p}(0), \ldots, \mathbf{M}^t\mathbf{p}(0) \rightsquigarrow \mathbf{p}^*$

$\mathbf{p}^*$ is the **stationary distribution** of the random walk

Equivalence between Formulations

Linear Algebra

Probabilistic

Equivalence between Formulations

Linear Algebra

System of linear "flow" equations

$$\boxed{\mathbf{r}(t+1) = \mathbf{M}\mathbf{r}(t)}$$

Probabilistic

Equivalence between Formulations

Linear Algebra

System of linear "flow" equations

$$\boxed{\mathbf{r}(t+1) = \mathbf{M}\mathbf{r}(t)}$$

Probabilistic

Random walk over web pages (Markov chain)

$$\boxed{\mathbf{p}(t+1) = \mathbf{M}\mathbf{p}(t)}$$

Equivalence between Formulations

Linear Algebra

System of linear "flow" equations

$$\boxed{\mathbf{r}(t+1) = \mathbf{M}\mathbf{r}(t)}$$

Use **power iteration** method to find the eigenvector $\mathbf{r}^*$ associated with the largest eigenvalue of **M** (λ = 1)

Probabilistic

Random walk over web pages (Markov chain)

$$\boxed{\mathbf{p}(t+1) = \mathbf{M}\mathbf{p}(t)}$$

Equivalence between Formulations

Linear Algebra

System of linear "flow" equations

$$\boxed{\mathbf{r}(t+1) = \mathbf{M}\mathbf{r}(t)}$$

Use **power iteration** method to find the eigenvector $\mathbf{r}^*$ associated with the largest eigenvalue of **M** ($\lambda = 1$)

Probabilistic

Random walk over web pages (Markov chain)

$$\boxed{\mathbf{p}(t+1) = \mathbf{M}\mathbf{p}(t)}$$

Model a random surfer who moves from one page to the other according to the state transition probabilities in **M** converging to a steady-state $\mathbf{p}^*$

Equivalence between Formulations

Linear Algebra

System of linear "flow" equations

$$\boxed{\mathbf{r}(t+1) = \mathbf{M}\mathbf{r}(t)}$$

Use **power iteration** method to find the eigenvector $\mathbf{r}^*$ associated with the largest eigenvalue of **M** ($\lambda = 1$)

Probabilistic

Random walk over web pages (Markov chain)

$$\boxed{\mathbf{p}(t+1) = \mathbf{M}\mathbf{p}(t)}$$

Model a random surfer who moves from one page to the other according to the state transition probabilities in **M** converging to a steady-state $\mathbf{p}^*$

$$\boxed{\mathbf{r}^* = \mathbf{p}^*}$$

Equivalence between Formulations

So the PageRank vector $\mathbf{r}^*$ corresponds to the **stationary distribution** $\mathbf{p}^*$ for the random walk on the graph encoded by $\mathbf{M}$!

Equivalence between Formulations

So the PageRank vector $\mathbf{r}^*$ corresponds to the **stationary distribution** $\mathbf{p}^*$ for the random walk on the graph encoded by **M**!

Intuitively, the PageRank vector indicates for each web page the probability that a random surfer will eventually get to that page

Hang on a second…

Linear Algebra

Probabilistic

Hang on a second…

Linear Algebra

How do we know that the power iteration method always converge to $\mathbf{r}^*$?

existence

Probabilistic

How do we know that a Markov chain always converge to a steady-state $\mathbf{p}^*$?

existence

Hang on a second…

Linear Algebra

How do we know that the power iteration method always converge to $\mathbf{r}^*$?

existence

How do we know that $\mathbf{r}^*$ is unique?

uniqueness

Probabilistic

How do we know that a Markov chain always converge to a steady-state $\mathbf{p}^*$?

existence

How do we know that $\mathbf{p}^*$ is unique?

uniqueness

Hang on a second…

Linear Algebra

How do we know that the power iteration method always converge to $\mathbf{r}^*$?

existence

How do we know that $\mathbf{r}^*$ is unique?

uniqueness

Probabilistic

How do we know that a Markov chain always converge to a steady-state $\mathbf{p}^*$?

existence

How do we know that $\mathbf{p}^*$ is unique?

uniqueness

existence and **uniqueness** of $\mathbf{r}^*$ ($\mathbf{p}^*$) are guaranteed under certain conditions on the matrix $\mathbf{M}$

Existence and Uniqueness of PageRank

If **M** is a **column stochastic** matrix with **all positive entries**:

- $\lambda = 1$ is an eigenvalue of **M** with multiplicity one
- $\lambda = 1$ is the largest eigenvalue of **M**
- There exists a unique (right) eigenvector $\mathbf{r}^*$ associated with the eigenvalue $\lambda = 1$ with the sum of its entries equal to 1

Perron-Frobenius theorem (circa 1910)

Existence and Uniqueness of PageRank

If **M** is a **column stochastic** matrix with **all positive entries**, then **M has** a **unique** steady-state vector $\mathbf{p}^*$ such that for any $\mathbf{p}(0)$

$\mathbf{p}(t) = \mathbf{M}^t\mathbf{p}(0)$ converges to $\mathbf{p}^*$ as $t \to \infty$

Perron-Frobenius theorem (circa 1910)

Existence and Uniqueness of PageRank

The Perron-Frobenius theorem ensures that the steady-state vector $\mathbf{p}^*$ exists and is unique

Existence and Uniqueness of PageRank

The Perron-Frobenius theorem ensures that the steady-state vector $\mathbf{p}^*$ exists and is unique

Such a steady-state vector is actually an **eigenvector** of a **positive stochastic** matrix **M** which corresponds to the **eigenvalue** $\lambda = 1$

Existence and Uniqueness of PageRank

The Perron-Frobenius theorem ensures that the steady-state vector $\mathbf{p}^*$ exists and is unique

Such a steady-state vector is actually an **eigenvector** of a **positive stochastic** matrix **M** which corresponds to the **eigenvalue** $\lambda = 1$

We know that the largest eigenvalue of a stochastic matrix is $\lambda = 1$ (though we haven't proved it)

Existence and Uniqueness of PageRank

The Perron-Frobenius theorem ensures that the steady-state vector $\mathbf{p}^*$ exists and is unique

Such a steady-state vector is actually an **eigenvector** of a **positive stochastic** matrix **M** which corresponds to the **eigenvalue** $\lambda = 1$

We know that the largest eigenvalue of a stochastic matrix is $\lambda = 1$ (though we haven't proved it)

The steady-state vector is the unique eigenvector associated with the largest eigenvalue $\lambda = 1$

Are We Done, Then?

Problem: We cannot apply the Perron-Frobenius theorem to the matrix **M** as we originally defined it

Are We Done, Then?

Problem: We cannot apply the Perron-Frobenius theorem to the matrix **M** as we originally defined it

M is column stochastic but it may not be strictly positive (i.e., it may contain some 0s)

Are We Done, Then?

Problem: We cannot apply the Perron-Frobenius theorem to the matrix **M** as we originally defined it

M is column stochastic but it may not be strictly positive (i.e., it may contain some 0s)

$$\mathbf{M}_1 = \begin{bmatrix} 0.6 & 0.5 & 0 \\ 0.4 & 0.3 & 1 \\ 0 & 0.2 & 0 \end{bmatrix} \qquad \mathbf{M}_2 = \begin{bmatrix} 0.6 & 0.5 & 0.1 \\ 0.2 & 0.3 & 0.4 \\ 0.2 & 0.2 & 0.5 \end{bmatrix}$$

Are We Done, Then?

Problem: We cannot apply the Perron-Frobenius theorem to the matrix **M** as we originally defined it

M is column stochastic but it may not be strictly positive (i.e., it may contain some 0s)

$$\mathbf{M}_1 = \begin{bmatrix} 0.6 & 0.5 & 0 \\ 0.4 & 0.3 & 1 \\ 0 & 0.2 & 0 \end{bmatrix} \qquad \mathbf{M}_2 = \begin{bmatrix} 0.6 & 0.5 & 0.1 \\ 0.2 & 0.3 & 0.4 \\ 0.2 & 0.2 & 0.5 \end{bmatrix}$$

Both $\mathbf{M}_1$ and $\mathbf{M}_2$ are column stochastic, but only $\mathbf{M}_2$ is positive

So? Should We Give Up?

Here is where Brin and Page, in fact Google, comes in!

So? Should We Give Up?

Here is where Brin and Page, in fact Google, comes in!

We show how they fixed the issues with the original definition of **M** to accommodate for the heterogeneity of the Web graph

So? Should We Give Up?

Here is where Brin and Page, in fact Google, comes in!

We show how they fixed the issues with the original definition of **M** to accommodate for the heterogeneity of the Web graph

By doing so, we know that a solution to our PageRank problem **exists** and is **unique**!

Google's PageRank

Problems with Original PageRank Formulation

We cannot directly apply the Perron-Frobenius theorem to the original Web graph matrix $\mathbf{M}$

Problems with Original PageRank Formulation

We cannot directly apply the Perron-Frobenius theorem to the original Web graph matrix **M**

M is non-negative but **not strictly positive**, and as we will see it may not even be column stochastic!

Problems with Original PageRank Formulation

We cannot directly apply the Perron-Frobenius theorem to the original Web graph matrix **M**

M is non-negative but **not strictly positive**, and as we will see it may not even be column stochastic!

We show why this causes the problem of **existence** and **convergence** of PageRank when applied to the original matrix **M**

Problems with Original PageRank Formulation

We cannot directly apply the Perron-Frobenius theorem to the original Web graph matrix **M**

M is non-negative but **not strictly positive**, and as we will see it may not even be column stochastic!

We show why this causes the problem of **existence** and **convergence** of PageRank when applied to the original matrix **M**

Then we discuss how Brin and Page fixed this in their seminal paper which sets up the rising of Google

Problems with Original PageRank Formulation

2 main **issues** to solve:

Problems with Original PageRank Formulation

2 main **issues** to solve:

Dead End

Pages with no outlinks cause PageRank to leak out

Problems with Original PageRank Formulation

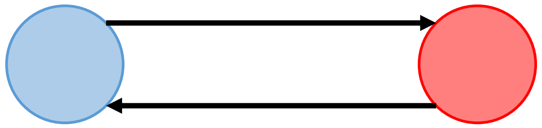
2 main **issues** to solve:

Dead End

Pages with no outlinks cause PageRank to leak out

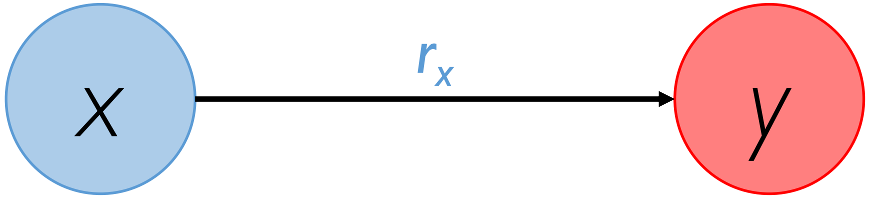
Spider Trap

Not every node is reachable and PageRank gets eventually absorbed by small group of pages



The "Dead End" Problem (Dangling Nodes)

Example:



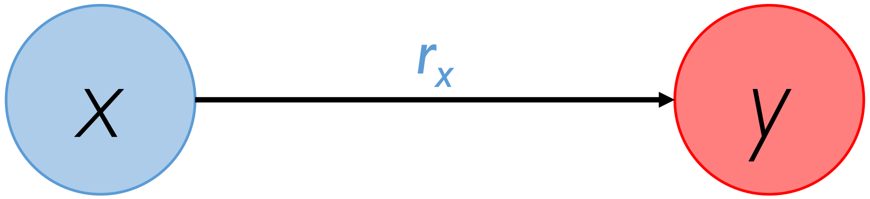
$$r_y = r_x$$

$\mathbf{M}$

0	0
1	0

The "Dead End" Problem (Dangling Nodes)

Example:



$$r_y = r_x$$

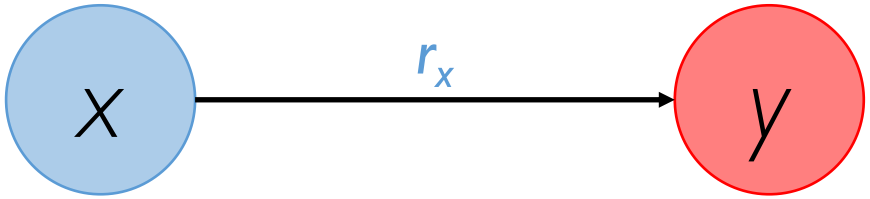
$$\mathbf{M} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}$$

When a web page has no outgoing links (**dangling node**) the resulting column vector in the matrix **M** is **not stochastic** anymore!

*Previously, we assumed each web page has at least one outgoing link, and therefore **M** was stochastic*

The "Dead End" Problem (Dangling Nodes)

Example:



$$r_y = r_x$$

$\mathbf{M}$

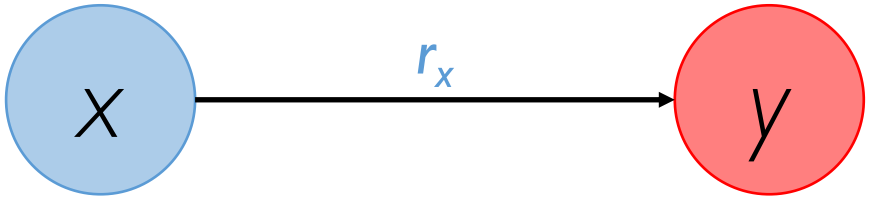
0	0
1	0

Assume the following initialization for **r**:

$$\mathbf{r}(0) = \begin{bmatrix} r_x^{(0)} \\ r_y^{(0)} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

The "Dead End" Problem (Dangling Nodes)

Example:



$$r_y = r_x$$

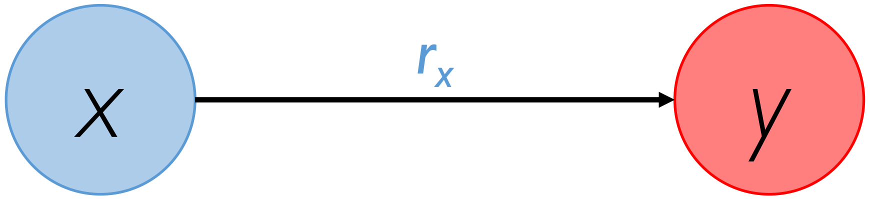
$$\begin{array}{c}\mathbf{M}\\ \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}\end{array}$$

$$\mathbf{r}(1) = \mathbf{M} \quad \mathbf{r}(0)$$

$$\begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

The "Dead End" Problem (Dangling Nodes)

Example:



$$r_y = r_x$$

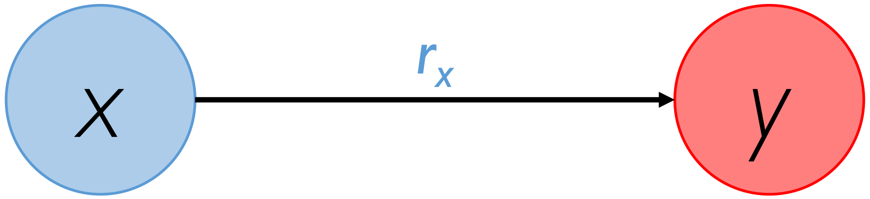
$$\mathbf{M} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}$$

$$\mathbf{r}(2) = \mathbf{M} \quad \mathbf{r}(1)$$

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

The "Dead End" Problem (Dangling Nodes)

Example:



$$r_y = r_x$$

M

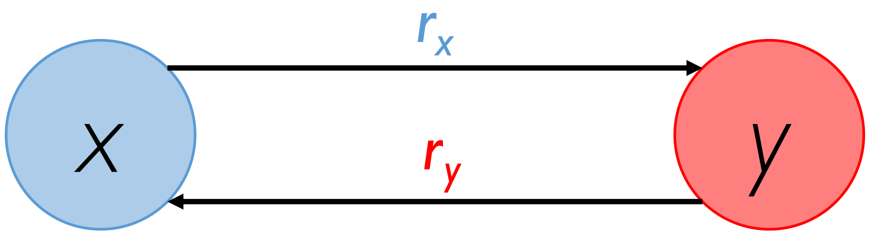
0	0
1	0

$$\mathbf{r}(0) = \begin{bmatrix}1\\0\end{bmatrix} \quad \mathbf{r}(1) = \begin{bmatrix}0\\1\end{bmatrix} \quad \mathbf{r}(2) = \begin{bmatrix}0\\0\end{bmatrix} \quad \cdots \quad \mathbf{r}(t-1) = \begin{bmatrix}0\\0\end{bmatrix} \quad \mathbf{r}(t) = \begin{bmatrix}0\\0\end{bmatrix}$$

The PageRank vector vanishes to **0**!

The "Spider Trap" Problem

Example:



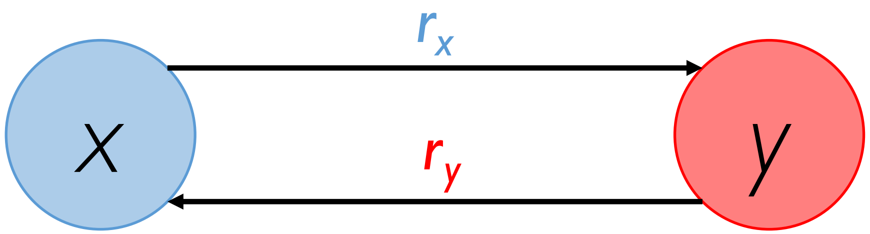
$$\begin{cases} r_x = r_y \\ r_y = r_x \end{cases}$$

M

0	1
1	0

The "Spider Trap" Problem

Example:



$$\begin{cases} r_x = r_y \\ r_y = r_x \end{cases}$$

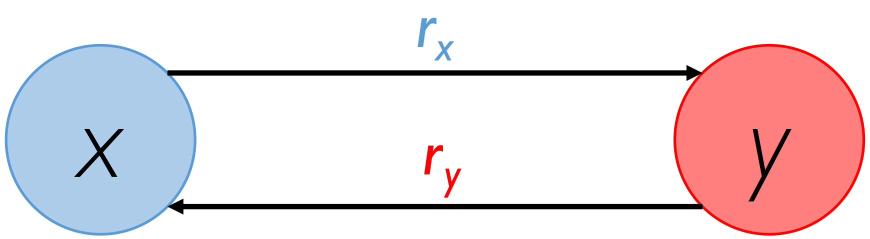
$$\mathbf{M} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

M is column stochastic non-negative (but **not strictly positive**)

Does PageRank converge regardless of the initialization of **r**?

The "Spider Trap" Problem

Example:



$$\begin{cases} r_x = r_y \\ r_y = r_x \end{cases}$$

M

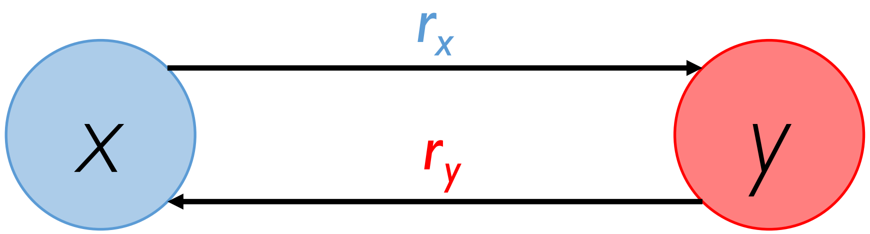
0	1
1	0

Assume the same initialization as before for **r**:

$$\mathbf{r}(0) = \begin{bmatrix} r_x^{(0)} \\ r_y^{(0)} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

The "Spider Trap" Problem

Example:



$$\begin{cases} r_x = r_y \\ r_y = r_x \end{cases}$$

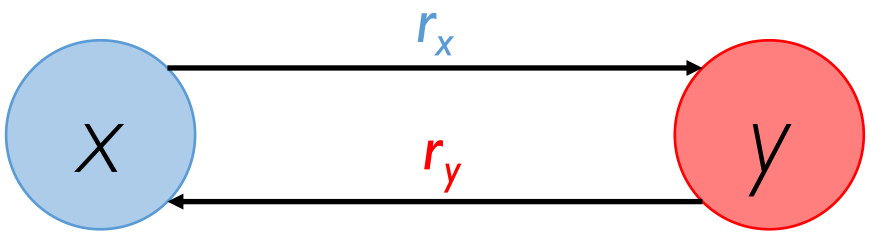
$$\mathbf{M} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$\mathbf{r}(1) = \mathbf{M} \quad \mathbf{r}(0)$$

$$\begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

The "Spider Trap" Problem

Example:



$$\begin{cases} r_x = r_y \\ r_y = r_x \end{cases}$$

M

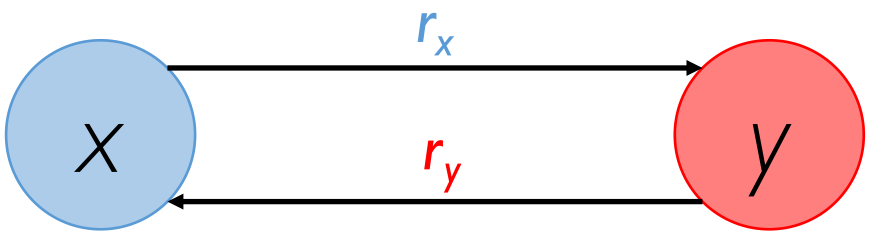
0	1
1	0

$$\mathbf{r}(2) = \mathbf{M} \quad \mathbf{r}(1)$$

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

The "Spider Trap" Problem

Example:



$$\begin{cases} r_x = r_y \\ r_y = r_x \end{cases}$$

$$\mathbf{M} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$\underset{\mathbf{r}(0)}{\begin{bmatrix} 1 \\ 0 \end{bmatrix}} \quad \underset{\mathbf{r}(1)}{\begin{bmatrix} 0 \\ 1 \end{bmatrix}} \quad \underset{\mathbf{r}(2)}{\begin{bmatrix} 1 \\ 0 \end{bmatrix}} \quad \cdots \quad \underset{\mathbf{r}(t-1)}{\begin{bmatrix} 0 \\ 1 \end{bmatrix}} \quad \underset{\mathbf{r}(t)}{\begin{bmatrix} 1 \\ 0 \end{bmatrix}}$$

The PageRank vector keeps alternating its components and **never** converges!

Problems with Original PageRank Formulation

2 main **issues** to solve:

Dead End

Pages with no outlinks cause PageRank to leak out

Spider Trap

Not every node is reachable and PageRank gets eventually absorbed by small group of pages

Problems with Original PageRank Formulation

2 main **issues** to solve:

Dead End

Pages with no outlinks cause PageRank to leak out

M is **not** column stochastic as some nodes have no outlinks

Spider Trap

Not every node is reachable and PageRank gets eventually absorbed by small group of pages

Problems with Original PageRank Formulation

2 main **issues** to solve:

Dead End

Pages with no outlinks cause PageRank to leak out

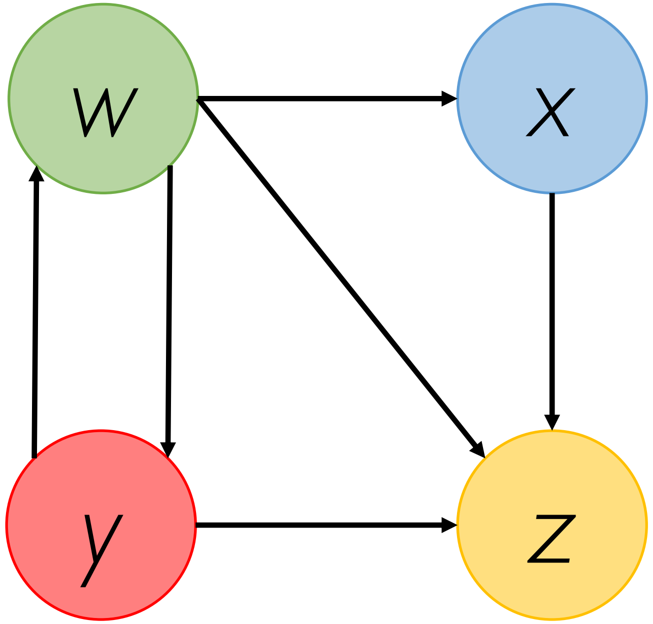
M is **not** column stochastic as some nodes have no outlinks

Spider Trap

Not every node is reachable and PageRank gets eventually absorbed by small group of pages

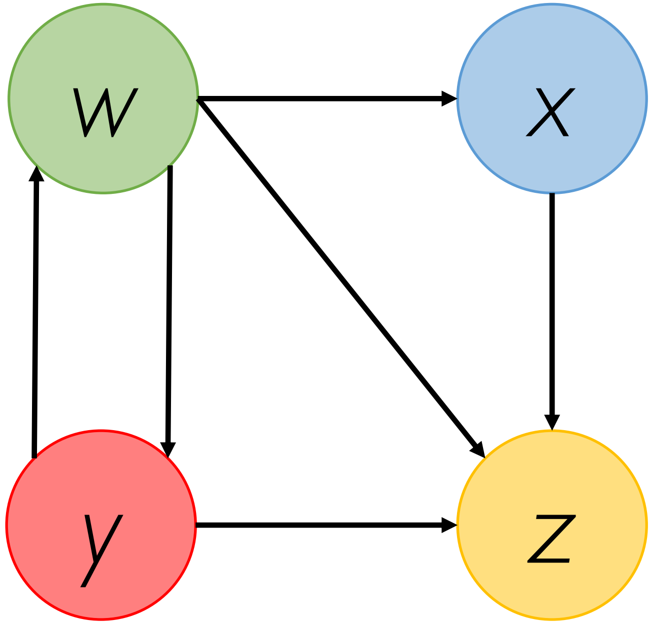
M is stochastic but **not** strictly positive

Deal with Dangling Nodes



$$\mathbf{M} = \begin{array}{c} \\ w \\ x \\ y \\ z \end{array} \begin{array}{c} \begin{array}{cccc} w & x & y & z \end{array} \\ \begin{bmatrix} 0 & 0 & 1/2 & 0 \\ 1/3 & 0 & 0 & 0 \\ 1/3 & 0 & 0 & 0 \\ 1/3 & 1 & 1/2 & 0 \end{bmatrix} \end{array}$$

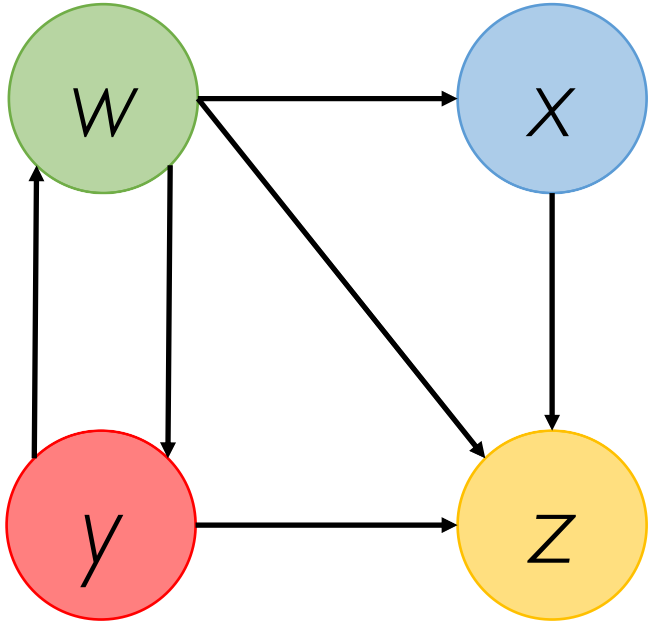
Deal with Dangling Nodes



z is a dangling node

$$\mathbf{M} = \begin{array}{c c} & \begin{array}{cccc} w & x & y & z \end{array} \\ \begin{array}{c} w \\ x \\ y \\ z \end{array} & \begin{bmatrix} 0 & 0 & 1/2 & \boxed{0} \\ 1/3 & 0 & 0 & \boxed{0} \\ 1/3 & 0 & 0 & \boxed{0} \\ 1/3 & 1 & 1/2 & \boxed{0} \end{bmatrix} \end{array}$$

Deal with Dangling Nodes

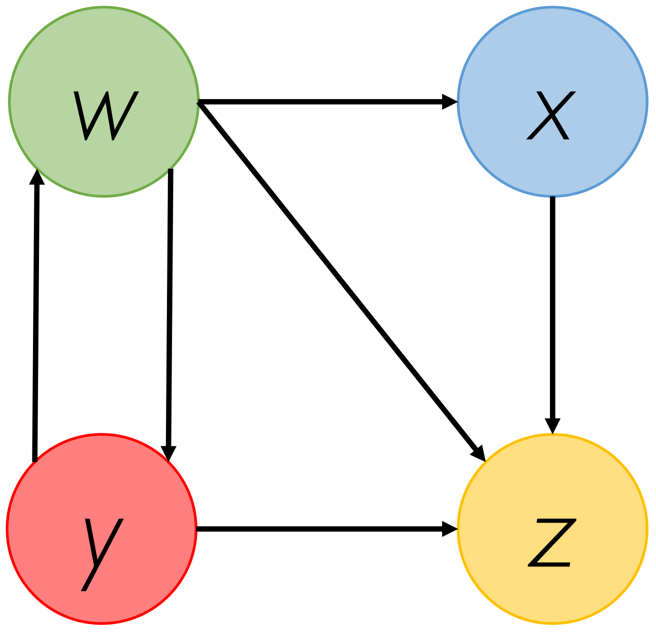


z is a dangling node

$$\mathbf{M} = \begin{array}{c} \\ w \\ x \\ y \\ z \end{array} \begin{array}{c} \begin{array}{cccc} w & x & y & z \end{array} \\ \begin{bmatrix} 0 & 0 & 1/2 & 0 \\ 1/3 & 0 & 0 & 0 \\ 1/3 & 0 & 0 & 0 \\ 1/3 & 1 & 1/2 & 0 \end{bmatrix} \end{array}$$

M is **not**
(column) stochastic

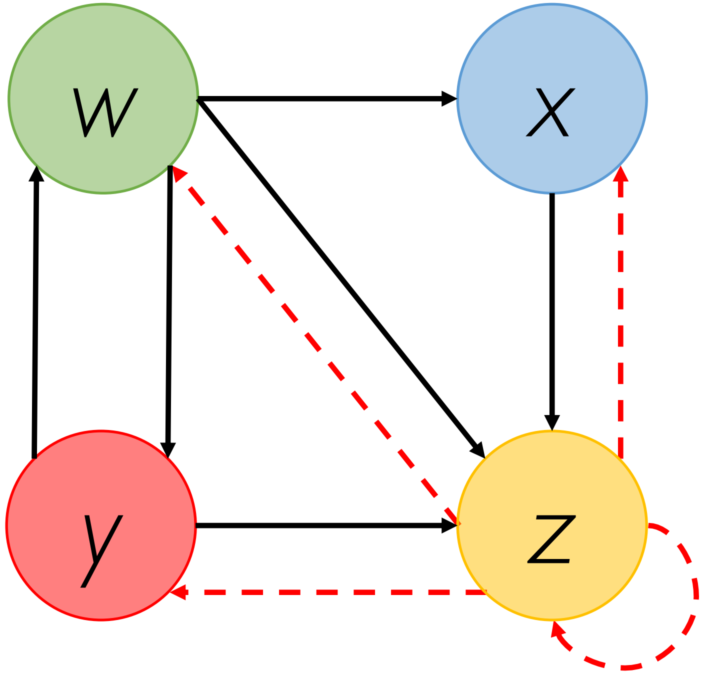
Deal with Dangling Nodes



$$\mathbf{M} = \begin{array}{c} \\ w \\ x \\ y \\ z \end{array} \begin{array}{c} \begin{array}{cccc} w & x & y & z \end{array} \\ \begin{bmatrix} 0 & 0 & 1/2 & 0 \\ 1/3 & 0 & 0 & 0 \\ 1/3 & 0 & 0 & 0 \\ 1/3 & 1 & 1/2 & 0 \end{bmatrix} \end{array}$$

If we apply simplified PageRank to **M** the rank vector **r** will eventually vanish to **0**

Deal with Dangling Nodes



$$
\begin{array}{c c}
 & \begin{array}{cccc} w & x & y & z \end{array} \\
\mathbf{M}' = \begin{array}{c} w \\ x \\ y \\ z \end{array} & \begin{bmatrix} 0 & 0 & 1/2 & 1/4 \\ 1/3 & 0 & 0 & 1/4 \\ 1/3 & 0 & 0 & 1/4 \\ 1/3 & 1 & 1/2 & 1/4 \end{bmatrix}
\end{array}
$$

Solution: Teleporting

Create **artificial links** from any dangling node to any other node

Deal with Dangling Nodes: Teleporting

This adjustment is justified by modeling the behaviour of a web surfer

Deal with Dangling Nodes: Teleporting

This adjustment is justified by modeling the behaviour of a web surfer

After reading a page with no out-going link, jump to a page picked **uniformly at random** amongst the N

Deal with Dangling Nodes: Teleporting

Initially, we set $\mathbf{M}_{N\times N}$ $m_{v,w} = \begin{cases} \frac{1}{o_w} & \text{if } v \in O_w \\ 0 & \text{otherwise} \end{cases}$

Deal with Dangling Nodes: Teleporting

Initially, we set $\mathbf{M}_{N\times N}$ $m_{v,w} = \begin{cases} \frac{1}{o_w} & \text{if } v \in O_w \\ 0 & \text{otherwise} \end{cases}$

Now we change it to $\mathbf{M}'_{N\times N}$ $m'_{v,w} = \begin{cases} \frac{1}{o_w} & \text{if } v \in O_w \\ \frac{1}{N} & \text{if } \sum_{v=1}^{N} m_{v,w} = 0 \\ 0 & \text{otherwise} \end{cases}$

Deal with Dangling Nodes: Teleporting

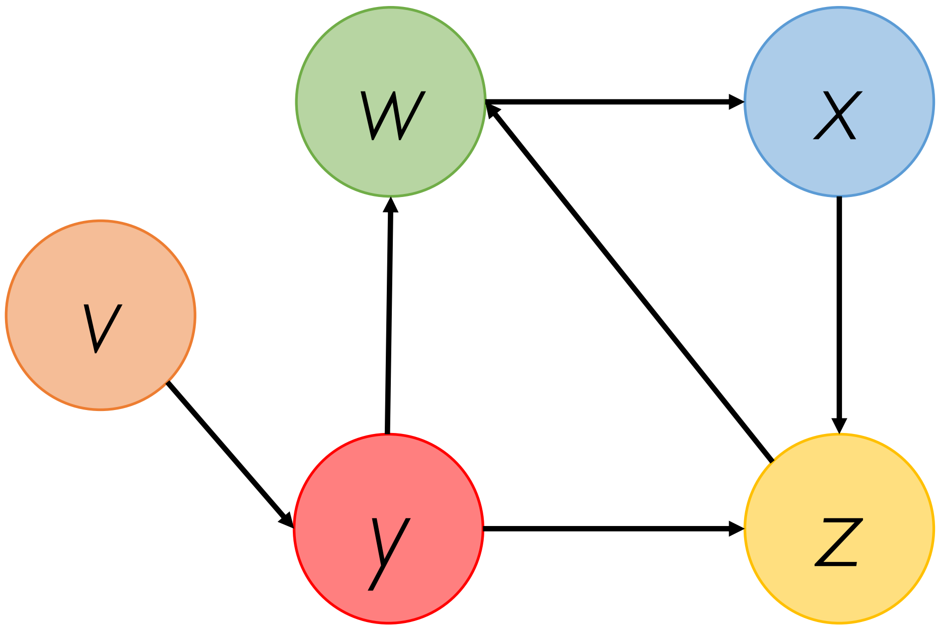
Initially, we set $\mathbf{M}_{N\times N}$ $m_{v,w} = \begin{cases} \frac{1}{o_w} & \text{if } v \in O_w \\ 0 & \text{otherwise} \end{cases}$

Now we change it to $\mathbf{M}'_{N\times N}$ $m'_{v,w} = \begin{cases} \frac{1}{o_w} & \text{if } v \in O_w \\ \frac{1}{N} & \text{if } \sum_{v=1}^{N} m_{v,w} = 0 \\ 0 & \text{otherwise} \end{cases}$

$$\boxed{\mathbf{M} \rightsquigarrow \mathbf{M}'}$$

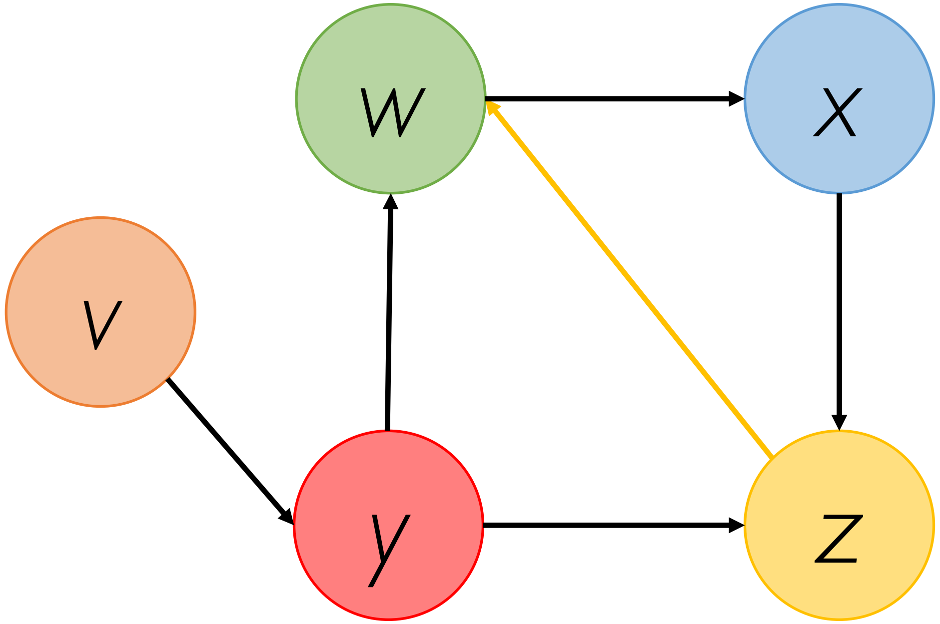
This transformation allows **M'** to be **column stochastic**

Deal with Spider Traps



$$\mathbf{M} = \begin{array}{c} \\ v \\ w \\ x \\ y \\ z \end{array} \begin{array}{c} \begin{array}{ccccc} v & w & x & y & z \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/2 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1/2 & 0 \end{bmatrix} \end{array}$$

Deal with Spider Traps

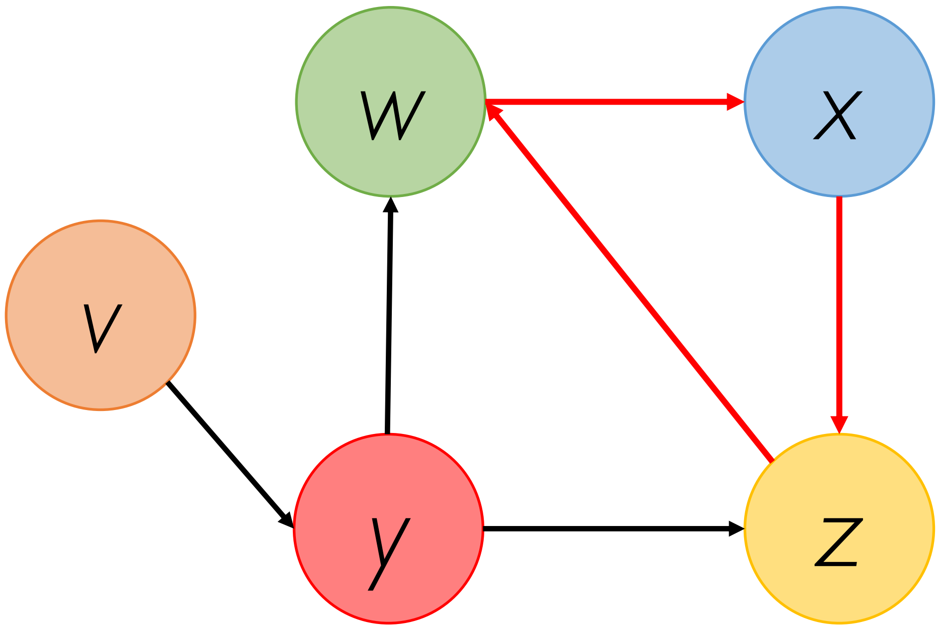


$$\mathbf{M} = \begin{array}{c} \\ v \\ w \\ x \\ y \\ z \end{array} \begin{array}{c} \begin{array}{ccccc} v & w & x & y & z \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/2 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1/2 & 0 \end{bmatrix} \end{array}$$

z is not a dangling node anymore

M is (column) stochastic

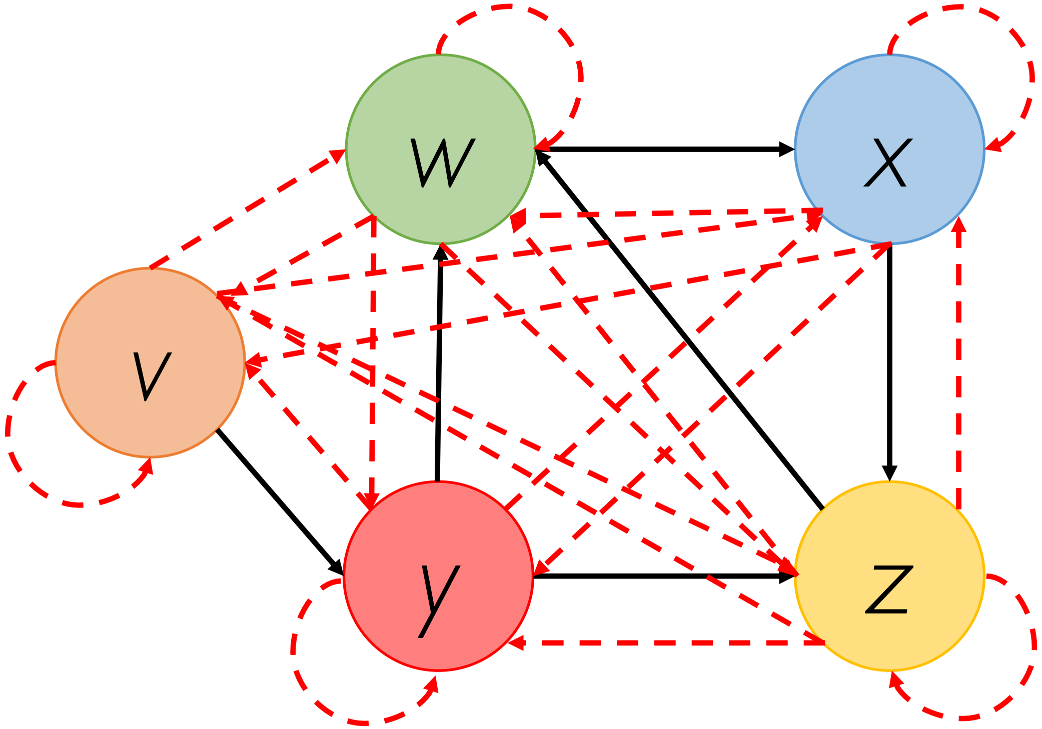
Deal with Spider Traps



$$\mathbf{M} = \begin{array}{c|ccccc} & v & w & x & y & z \\ \hline v & 0 & 0 & 0 & 0 & 0 \\ w & 0 & 0 & 0 & 1/2 & 1 \\ x & 0 & 1 & 0 & 0 & 0 \\ y & 1 & 0 & 0 & 0 & 0 \\ z & 0 & 0 & 1 & 1/2 & 0 \end{array}$$

If we apply simplified PageRank to **M** some entries of the rank vector **r** will eventually drop to 0, as we get stuck in *w*, *x*, *z*

Deal with Spider Traps: Teleporting (Again!)



$$\mathbf{M}' = \begin{array}{c} \\ v \\ w \\ x \\ y \\ z \end{array} \begin{array}{c} \begin{array}{ccccc} v & w & x & y & z \end{array} \\ \begin{bmatrix} 0.03 & 0.03 & 0.03 & 0.03 & 0.03 \\ 0.03 & 0.03 & 0.03 & 0.455 & 0.88 \\ 0.03 & 0.88 & 0.03 & 0.03 & 0.03 \\ 0.88 & 0.03 & 0.03 & 0.03 & 0.03 \\ 0.03 & 0.03 & 0.88 & 0.455 & 0.03 \end{bmatrix} \end{array}$$

Solution: Probabilistic Teleporting

Create **artificial links** from each node to every other node and follow each of it with probability (1-*d*)/N

Deal with Spider Traps: Probabilistic Teleporting

To avoid the surfer to get stuck in a spider trap

Deal with Spider Traps: Probabilistic Teleporting

To avoid the surfer to get stuck in a spider trap

On each page *w* the surfer will either follow one of its outgoing links with probability *d* or jump to another page with probability (1-*d*)

Deal with Spider Traps: Probabilistic Teleporting

To avoid the surfer to get stuck in a spider trap

On each page *w* the surfer will either follow one of its outgoing links with probability *d* or jump to another page with probability (1-*d*)

d is called **damping factor**

d = **0.85** in the original Google formulation

The Google's PageRank Formulation

$$\mathbf{M}_{N\times N} \quad m_{v,w} = \begin{cases} \frac{1}{o_w} & \text{if } v \in O_w \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{M}'_{N\times N} \quad m'_{v,w} = \begin{cases} \frac{1}{o_w} & \text{if } v \in O_w \\ \frac{1}{N} & \text{if } \sum_{v=1}^{N} m_{v,w} = 0 \\ 0 & \text{otherwise} \end{cases}$$

$\boxed{\mathbf{M} \leadsto \mathbf{M}'}$

Ensure the matrix is **stochastic**

The Google's PageRank Formulation

$$\mathbf{M}_{N\times N} \quad m_{v,w} = \begin{cases} \frac{1}{o_w} & \text{if } v \in O_w \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{M}'_{N\times N} \quad m'_{v,w} = \begin{cases} \frac{1}{o_w} & \text{if } v \in O_w \\ \frac{1}{N} & \text{if } \sum_{v=1}^{N} m_{v,w} = 0 \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{G} = d\mathbf{M}' + \frac{1-d}{N} \underbrace{\begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & 1 & \dots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \dots & 1 \end{bmatrix}}_{\mathbf{1}_{N\times N}}$$

$\boxed{\mathbf{M} \rightsquigarrow \mathbf{M}'}$

Ensure the matrix is **stochastic**

$\boxed{\mathbf{M}' \rightsquigarrow \mathbf{G}}$

Ensure the matrix is **strictly positive**

Why Does Teleporting Solve Our Problem?

$$\mathbf{G} = d\mathbf{M}' + \frac{1-d}{N} \underbrace{\begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & 1 & \dots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \dots & 1 \end{bmatrix}}_{\mathbf{1}_{N \times N}}$$

The matrix **G** so modified is (column) **stochastic** and **strictly positive**

Why Does Teleporting Solve Our Problem?

$$\mathbf{G} = d\mathbf{M}' + \frac{1-d}{N}\underbrace{\begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & 1 & \dots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \dots & 1 \end{bmatrix}}_{\mathbf{1}_{N\times N}}$$

The matrix **G** so modified is (column) **stochastic** and **strictly positive**

The Perron-Frobenius theorem now applies to **G** and guarantees the existence (convergence) and uniqueness of the steady-state eigenvector $\mathbf{r}^*$

$$\mathbf{r}(t) = \mathbf{G}^t\mathbf{r}(0)$$

$$\mathbf{r} \rightsquigarrow \mathbf{r}^* \text{ as } t \rightarrow \infty$$

How Do We Actually Compute PageRank?

$$\mathbf{r}(t+1) = \mathbf{G}\mathbf{r}(t)$$

How Do We Actually Compute PageRank?

$$\mathbf{r}(t+1) = \mathbf{G}\mathbf{r}(t)$$

Key step is matrix-vector multiplication

How Do We Actually Compute PageRank?

$$\mathbf{r}(t+1) = \mathbf{G}\mathbf{r}(t)$$

Key step is matrix-vector multiplication

Easy if we have enough memory to store **G**, **r**(t+1), and **r**(t)

How Do We Actually Compute PageRank?

$$\mathbf{r}(t+1) = \mathbf{G}\mathbf{r}(t)$$

Key step is matrix-vector multiplication

Easy if we have enough memory to store **G**, **r**(t+1), and **r**(t)

Problem:

G represents a **fully-connected** graph with a huge number of nodes (web pages)

G is a dense matrix

How Do We Actually Compute PageRank?

Assuming the number of web pages in the graph is $N=10^9$

G will have N^2 entries = 10^{18}

Say each entry is stored using a 32-bit integer (i.e., 4 bytes per entry)

How Do We Actually Compute PageRank?

Assuming the number of web pages in the graph is $N=10^9$

G will have N^2 entries = 10^{18}

Say each entry is stored using a 32-bit integer (i.e., 4 bytes per entry)

We will need 4×10^{18} bytes = **4 EB** just to store **G**

How Do We Actually Compute PageRank?

Assuming the number of web pages in the graph is $N=10^9$

G will have N^2 entries = 10^{18}

Say each entry is stored using a 32-bit integer (i.e., 4 bytes per entry)

We will need 4×10^{18} bytes = **4 EB** just to store **G**

Plus, we will need 4×10^9 bytes = 4 GB to store $\mathbf{r}(t+1)$ and $\mathbf{r}(t)$ each = **8 GB**

How Do We Actually Compute PageRank?

Assuming the number of web pages in the graph is $N=10^9$

G will have N^2 entries = 10^{18}

Say each entry is stored using a 32-bit integer (i.e., 4 bytes per entry)

We will need 4×10^{18} bytes = **4 EB** just to store **G**

Plus, we will need 4×10^9 bytes = 4 GB to store $\mathbf{r}(t+1)$ and $\mathbf{r}(t)$ each = **8 GB**

Note: The Web contains far more than $N=10^9$ pages!

Re-Arrange the Equation

$$\mathbf{r} = \mathbf{G}\mathbf{r}$$

$$\mathbf{G}_{v,w} = d\mathbf{M}'_{v,w} + \frac{1-d}{N}$$

Re-Arrange the Equation

$$\mathbf{r} = \mathbf{G}\mathbf{r}$$

$$\mathbf{G}_{v,w} = d\mathbf{M}'_{v,w} + \frac{1-d}{N}$$

$$r_v = \sum_{w=1}^{N} \mathbf{G}_{v,w} \times r_w$$

Re-Arrange the Equation

$$\mathbf{r} = \mathbf{G}\mathbf{r}$$

$$\mathbf{G}_{v,w} = d\mathbf{M}'_{v,w} + \frac{1-d}{N}$$

$$r_v = \sum_{w=1}^{N} \mathbf{G}_{v,w} \times r_w$$

$$r_v = \sum_{w=1}^{N} \underbrace{\left(d\mathbf{M}'_{v,w} + \frac{1-d}{N}\right)}_{\mathbf{G}_{v,w}} \times r_w$$

Re-Arrange the Equation

$$\mathbf{r} = \mathbf{G}\mathbf{r} \qquad \mathbf{G}_{v,w} = d\mathbf{M}'_{v,w} + \frac{1-d}{N}$$

$$r_v = \sum_{w=1}^{N} \mathbf{G}_{v,w} \times r_w \qquad r_v = \sum_{w=1}^{N} \underbrace{\left(d\mathbf{M}'_{v,w} + \frac{1-d}{N} \right)}_{\mathbf{G}_{v,w}} \times r_w$$

$$r_v = \sum_{w=1}^{N} d\mathbf{M}'_{v,w} \times r_w + \sum_{w=1}^{N} \frac{1-d}{N} \times r_w$$

Re-Arrange the Equation

$$r_v = \sum_{w=1}^{N} d\mathbf{M}'_{v,w} \times r_w + \sum_{w=1}^{N} \frac{1-d}{N} \times r_w$$

Re-Arrange the Equation

$$r_v = \sum_{w=1}^{N} d\mathbf{M}'_{v,w} \times r_w + \sum_{w=1}^{N} \frac{1-d}{N} \times r_w$$

$$r_v = \sum_{w=1}^{N} d\mathbf{M}'_{v,w} \times r_w + \frac{1-d}{N} \underbrace{\sum_{w=1}^{N} r_w}_{=1}$$

Re-Arrange the Equation

$$r_v = \sum_{w=1}^{N} d\mathbf{M}'_{v,w} \times r_w + \sum_{w=1}^{N} \frac{1-d}{N} \times r_w \qquad r_v = \sum_{w=1}^{N} d\mathbf{M}'_{v,w} \times r_w + \frac{1-d}{N} \underbrace{\sum_{w=1}^{N} r_w}_{=1}$$

$$r_v = \sum_{w=1}^{N} d\mathbf{M}'_{v,w} \times r_w + \frac{1-d}{N}$$

Re-Arrange the Equation

$$r_v = \sum_{w=1}^{N} d\mathbf{M}'_{v,w} \times r_w + \sum_{w=1}^{N} \frac{1-d}{N} \times r_w \qquad r_v = \sum_{w=1}^{N} d\mathbf{M}'_{v,w} \times r_w + \frac{1-d}{N} \underbrace{\sum_{w=1}^{N} r_w}_{=1}$$

$$r_v = \sum_{w=1}^{N} d\mathbf{M}'_{v,w} \times r_w + \frac{1-d}{N}$$

$$\boxed{\mathbf{r} = d\mathbf{M}'\mathbf{r} + \left[\frac{1-d}{N}\right]_{N\times 1}} \qquad \begin{bmatrix} \frac{1-d}{N} \\ \frac{1-d}{N} \\ \vdots \\ \frac{1-d}{N} \end{bmatrix}$$

Re-Arrange the Equation

$$\boxed{\mathbf{r} = d\mathbf{M}'\mathbf{r} + \left[\frac{1-d}{N}\right]_{N\times 1}}$$

M' is a sparse matrix (with no dangling nodes!)

Re-Arrange the Equation

$$\boxed{\mathbf{r} = d\mathbf{M}'\mathbf{r} + \left[\frac{1-d}{N}\right]_{N\times 1}}$$

M' is a sparse matrix (with no dangling nodes!)

Approximately 10 links per web page reduces the amount of memory required to store **M'** by a factor of 8 w.r.t. **G** (10^{10} vs. 10^{18} entries)

Re-Arrange the Equation

$$\boxed{\mathbf{r} = d\mathbf{M}'\mathbf{r} + \left[\frac{1-d}{N}\right]_{N\times 1}}$$

M' is a sparse matrix (with no dangling nodes!)

Approximately 10 links per web page reduces the amount of memory required to store **M'** by a factor of 8 w.r.t. **G** (10^{10} vs. 10^{18} entries)

We can work with **M'** rather than **G**

Re-Arrange the Equation

$$\boxed{\mathbf{r} = d\mathbf{M}'\mathbf{r} + \left[\frac{1-d}{N}\right]_{N\times 1}}$$

At each iteration we can compute PageRank vector as follows:

1. $\mathbf{r}(t+1) = d\mathbf{M}'\mathbf{r}(t)$
2. $\mathbf{r}(t+1) = \mathbf{r}(t+1) + \left[\frac{1-d}{N}\right]_{N\times 1}$

Add the constant (1-d)/N to each component of **r**(t+1)

PageRank: Pseudocode

Algorithm: PageRank

Input : A directed Web graph $G = (V, E)$, where $|V| = N$ and its associated matrix $\mathbf{M}_{N \times N}$ defined as follows: $\mathbf{M}_{v,w} = \frac{1}{o_w}$ if w points to v, 0 otherwise ($o_w = |O_w|$ where $O_w = \{x \in V : (w, x) \in E\}$);
A *damping factor* $d \in (0, 1)$;
A *tolerance* $\epsilon > 0$.

Output: The PageRank vector $\mathbf{r}^*_{N \times 1}$

Init : $t \leftarrow 0; \mathbf{r}(t) \leftarrow \left(\frac{1}{N}, \ldots, \frac{1}{N}\right)$;

repeat
    $t \leftarrow t + 1$;
    /* Compute the temporary PageRank score of every page v */
    for $i \leftarrow 1$ **to** N **do**
        $r_v^{\text{tmp}}(t) \leftarrow \sum_{w \in I_v} \frac{r_w(t-1)}{o_w}$; /* $r_v^{\text{tmp}}(t) = 0$ if v has no in-links */
    end
    /* Adjust the PageRank score of each page v with *teleporting* */
    for $i \leftarrow 1$ **to** N **do**
        $r_v(t) \leftarrow d \times r_v^{\text{tmp}}(t) + \frac{1-d}{N}$;
    end
until $|\mathbf{r}(t) - \mathbf{r}(t-1)| < \epsilon$
return $\mathbf{r}^* = \mathbf{r}(t)$;

Take-Home Message of Today

- We present an example of **link analysis** algorithm: **PageRank**

Take-Home Message of Today

- We present an example of **link analysis** algorithm: **PageRank**
- **Goal:** Find an **importance score** associated with each web page

Take-Home Message of Today

- We present an example of **link analysis** algorithm: **PageRank**
- **Goal:** Find an **importance score** associated with each web page
- Represent the Web graph as a matrix **M** where a link between page *v* and *w* is a **vote** from *v* to *w*

Take-Home Message of Today

- We present an example of **link analysis** algorithm: **PageRank**
- **Goal:** Find an **importance score** associated with each web page
- Represent the Web graph as a matrix **M** where a link between page *v* and *w* is a **vote** from *v* to *w*
- **2** different yet equivalent approaches:
  - **Linear Algebra** → Matrix eigenvector
  - **Probabilistic** → Stationary distribution of Markov chain (**random walk**)

Take-Home Message of Today

- The **existence** (convergence) and **uniqueness** of PageRank is guaranteed only for certain matrices **M** (Perron-Frobenius theorem)

Take-Home Message of Today

- The **existence** (convergence) and **uniqueness** of PageRank is guaranteed only for certain matrices **M** (Perron-Frobenius theorem)
- The Web graph is **disconnected** and may contain **no-exit loops**

Take-Home Message of Today

- The **existence** (convergence) and **uniqueness** of PageRank is guaranteed only for certain matrices **M** (Perron-Frobenius theorem)
- The Web graph is **disconnected** and may contain **no-exit loops**
- Google solution: **probabilistic teleport links**

Take-Home Message of Today

- The **existence** (convergence) and **uniqueness** of PageRank is guaranteed only for certain matrices **M** (Perron-Frobenius theorem)
- The Web graph is **disconnected** and may contain **no-exit loops**
- Google solution: **probabilistic teleport links**
- Still efficiently computable from the original, sparse matrix **M**