Markov Chains, LAMP Models and Reverse-Engineering

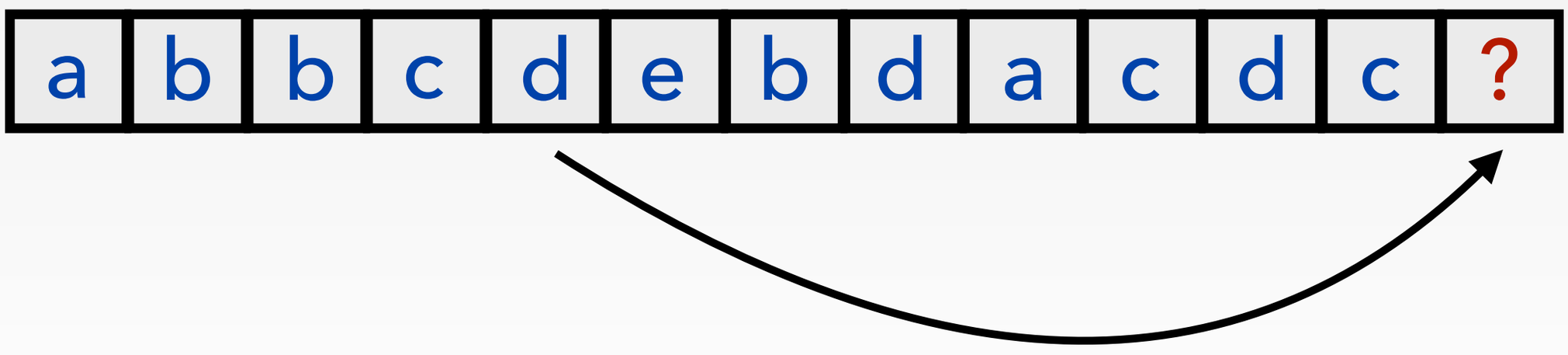
LAMP Models

Ravi Kumar, Maithra Raghu,
Tamas Sarlos and Andrew Tomkins
[Ref: WWW 2017]

Problem setting

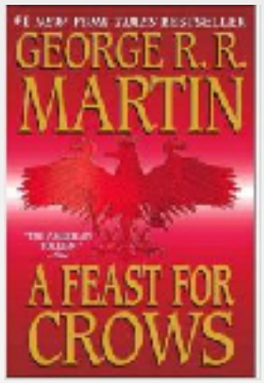
We consider models of sequences of outputs

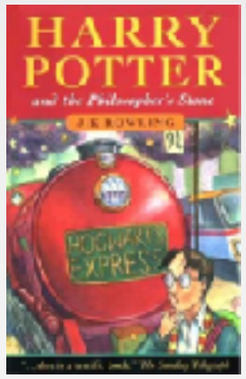
- Output ‘d’ can depend on earlier ‘d’ anywhere in history
- Dependence on history can be learned



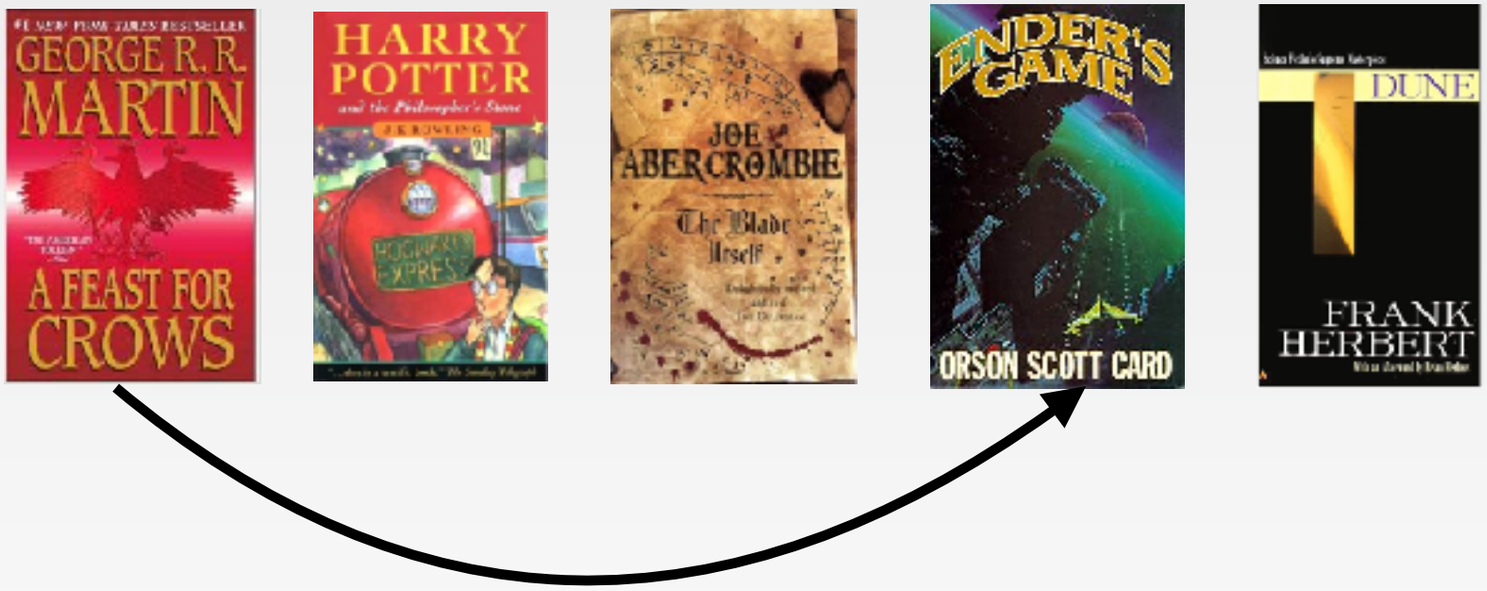
What if output ‘c’ is often (eventually) followed by output ‘d’?

Example: Science Fiction Novels

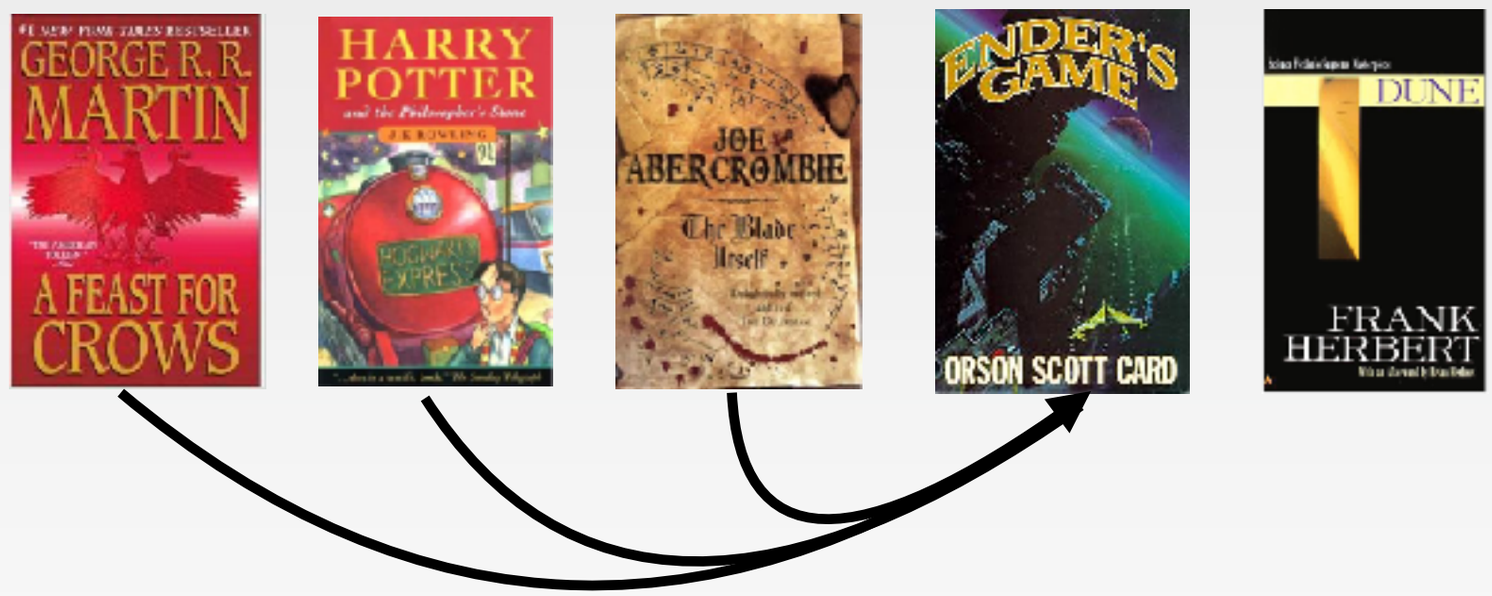




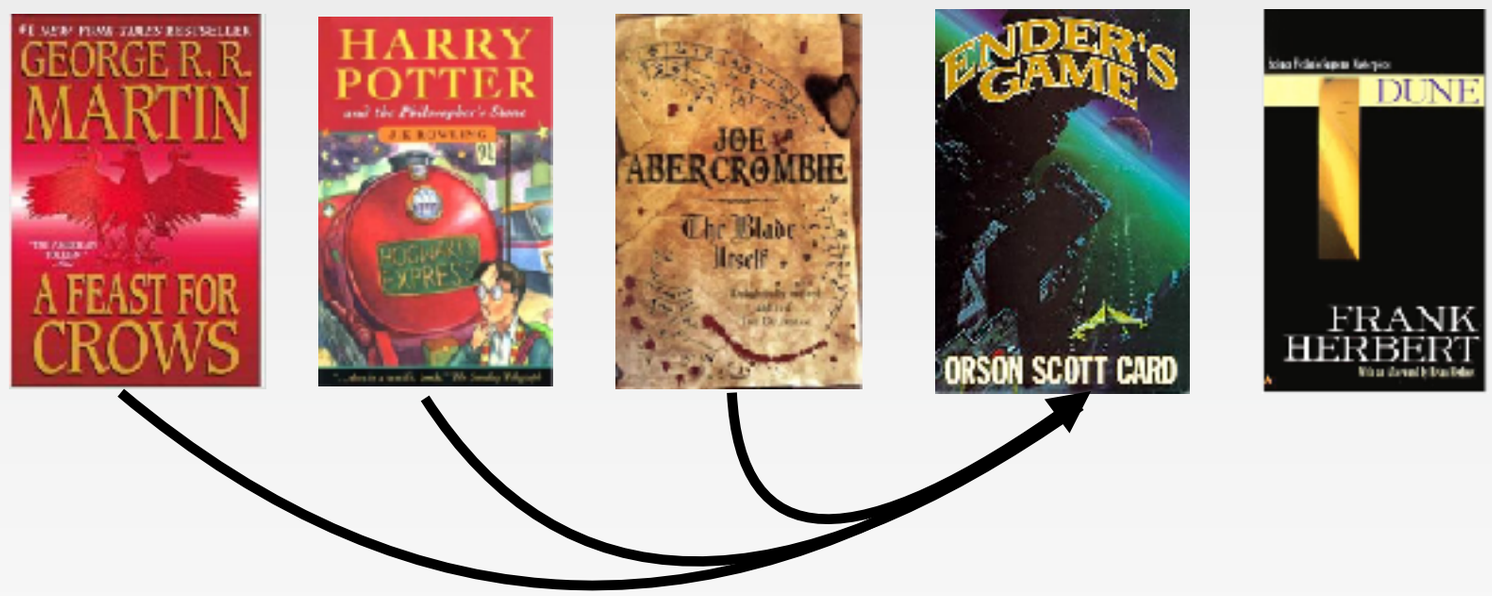
Example: Science Fiction Novels



Example: Science Fiction Novels



Example: Science Fiction Novels



Many other examples:

Spotify

REUTERS

Simplest approach: consider most recent element

a	b	b	c	d	e	b	d	a	c	d	c	?

Most recent letter most predictive. Following c:
{ a:100, b:200, c:1273, d:11 }

Can write Pr[next letter | current letter] as matrix:

$$W = \begin{pmatrix} 0.5 & 0.1 & 0.1 & 0.3 \\ 0 & 0.8 & 0.15 & 0.05 \\ .06 & 0.13 & .8 & .007 \\ 0.1 & 0.1 & 0.1 & 0.7 \end{pmatrix}$$

First-order Markov Model $MM_1(W)$: $x_{\text{new}} = W^T x_{\text{old}}$

But is this enough?

Generally, looking at more history should provide better models

Approaches to long-range dependencies:

- High-order or variable-order Markov models
- Deep network sequence models
- Point processes
- Many others

Higher Order Markov Models

- Next state only depends on **k previous** states
- But dependence is arbitrary

$$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow \cdots \rightarrow x_{t-k} \rightarrow \cdots \rightarrow x_{t-1} \rightarrow ?$$

- n possible states
- n^{k+1} parameters

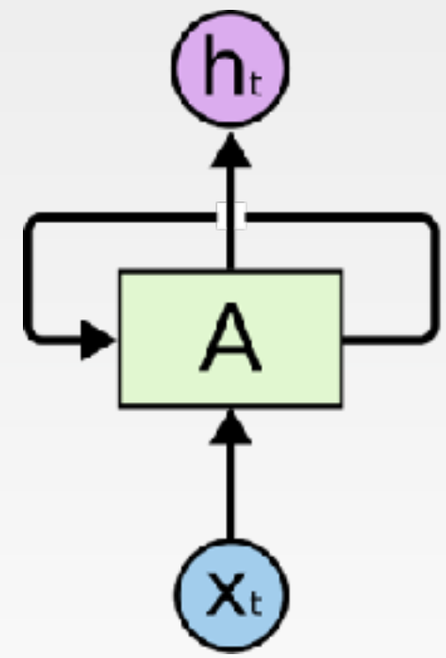
$$W = \begin{pmatrix} w(1,1) & \cdots & w(1,n) \\ w(w,1) & \cdots & w(2,n) \\ \vdots & \ddots & \vdots \\ w(n^k,1) & \cdots & p(n^k,n) \end{pmatrix}$$

Even Variable-Order models require exponential space for order-d dependencies

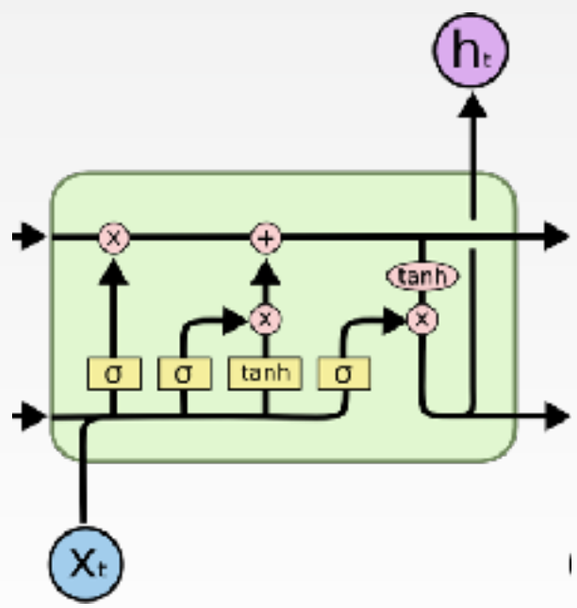
Deep Neural Network Models

Recurrent neural networks

- (Generating Sequences with RNNs, Graves, 2014)



- LSTMs (Long-Short Term Memory)
- Complex non-linear relations between previous states



Concerns

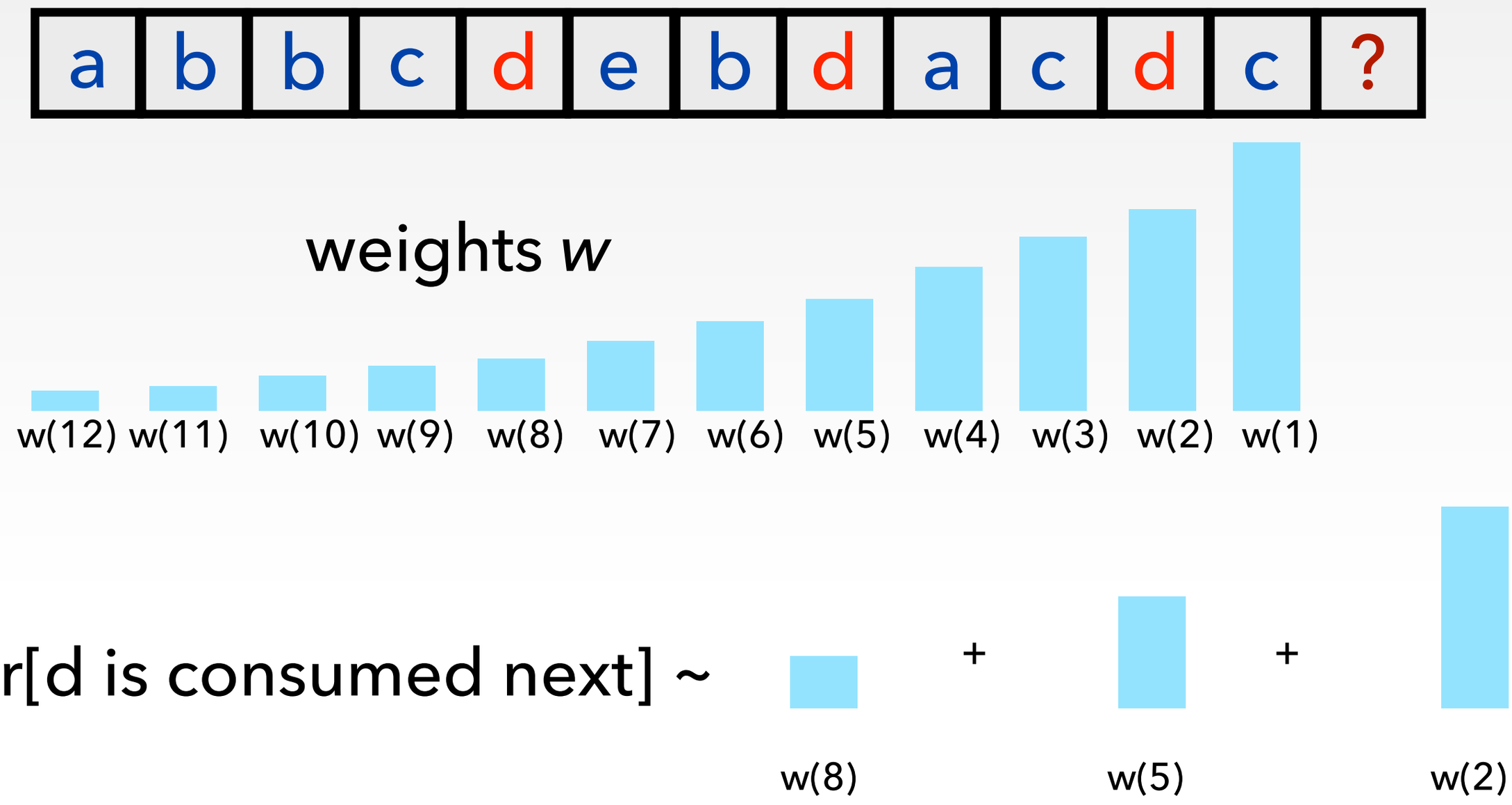
- Slow to train
- Requires lots of data

Introduction to recency weighting

Significant body of work on models of re-consumption, based on extensions of Simon's copying model [Simon'55]:

Introduction to recency weighting

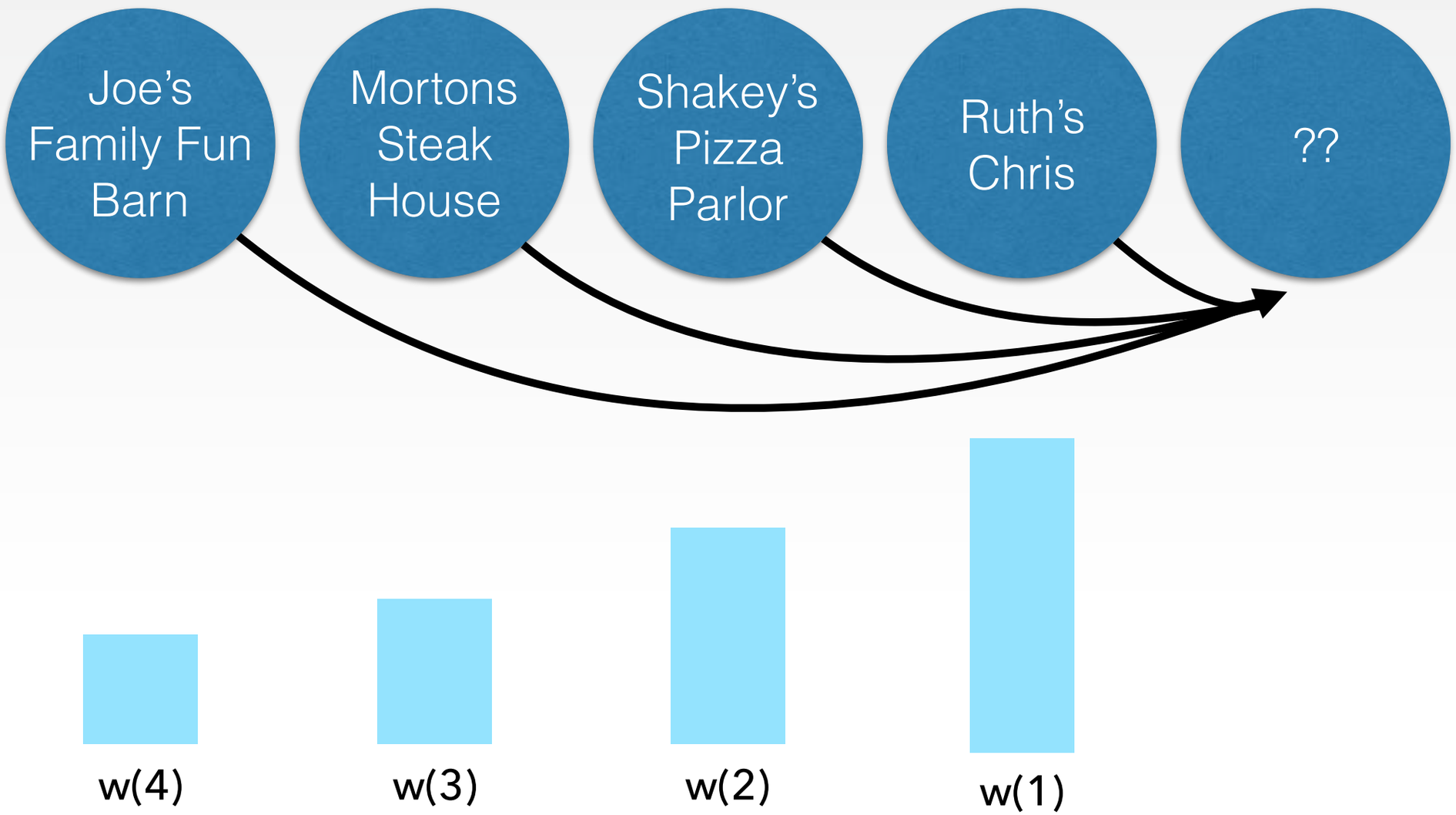
Significant body of work on models of re-consumption, based on extensions of Simon's copying model [Simon'55]:



Combining Recency-Weighting with Markov

Extending the same idea to Markov models:

Next state is a mixture



Linear Additive Markov Process (LAMP)

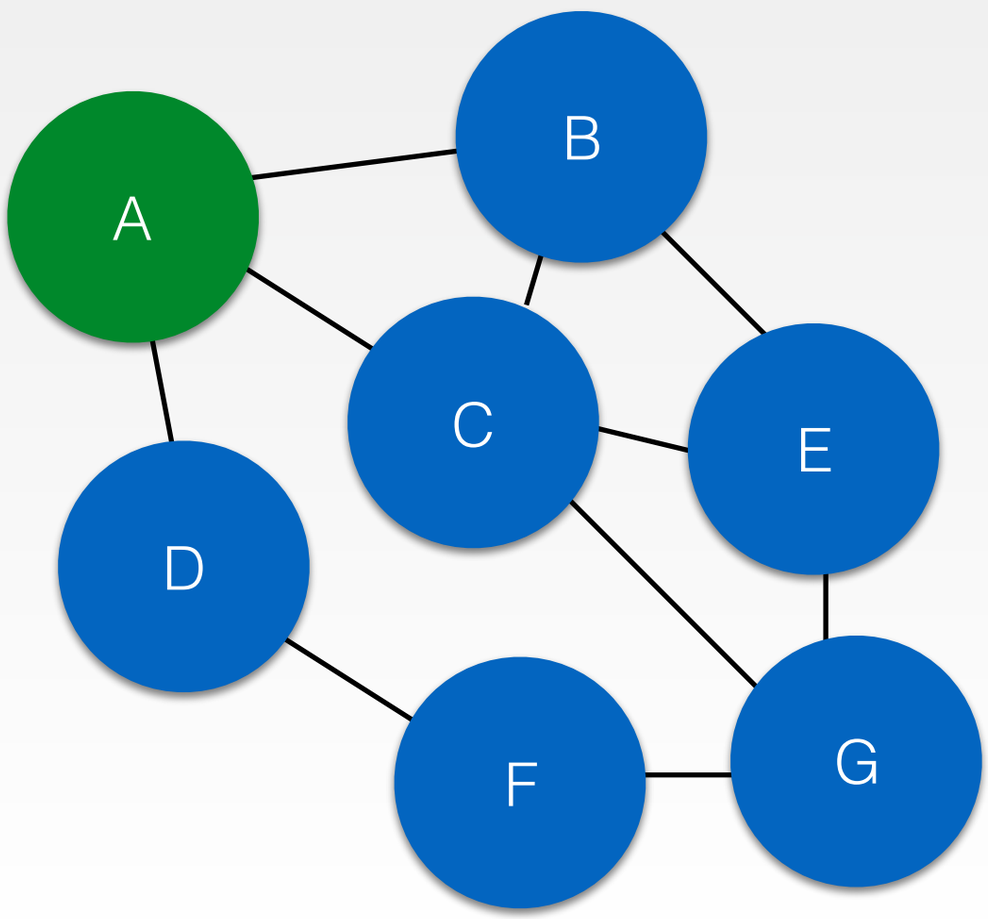
Definition of $\text{LAMP}_k(w, W)$

- **W** stochastic (transition) matrix
- Vector **w** with k weights

$$\Pr[X_t = x_t | x_0, \ldots, x_{t-1}] = \sum_{i=1}^{k} w_i W^T \vec{\mathbf{1}}_{x_{t-i}}$$

Total parameter complexity: NNZ(W) + k
Must learn both matrix W and history distribution w
We use alternating minimization — details in paper

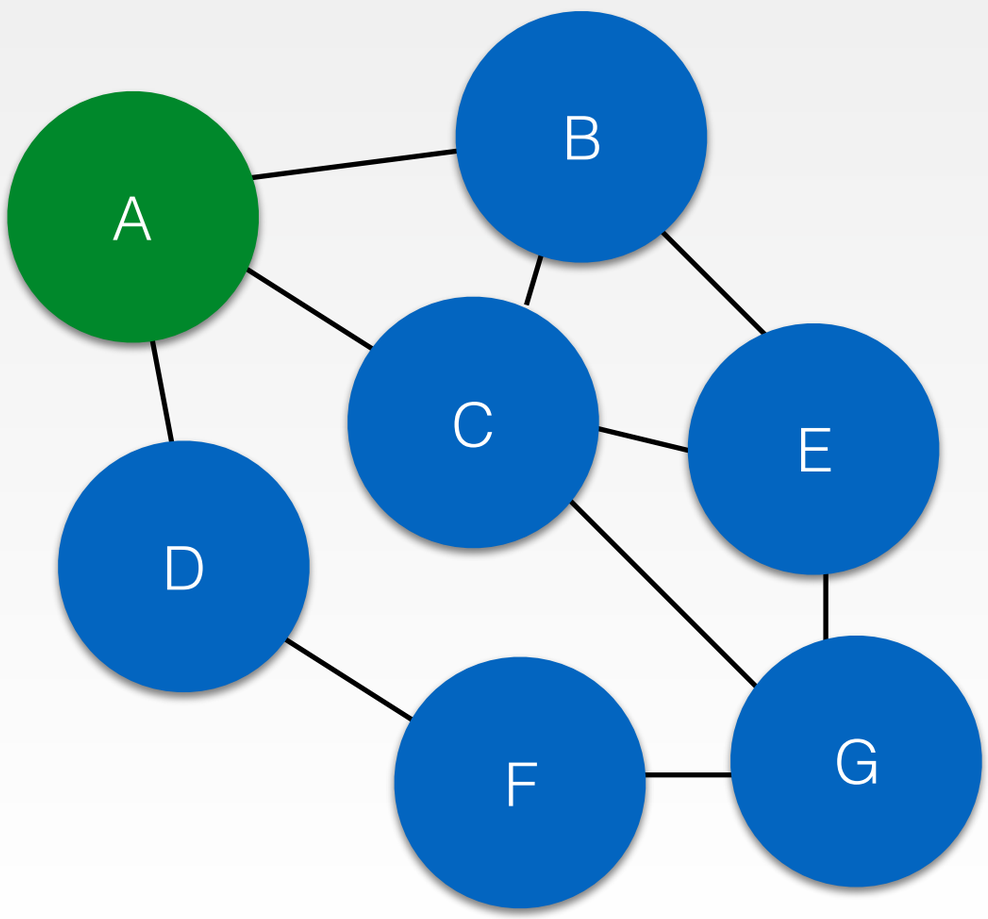
Example LAMP Walk



Current path:

A

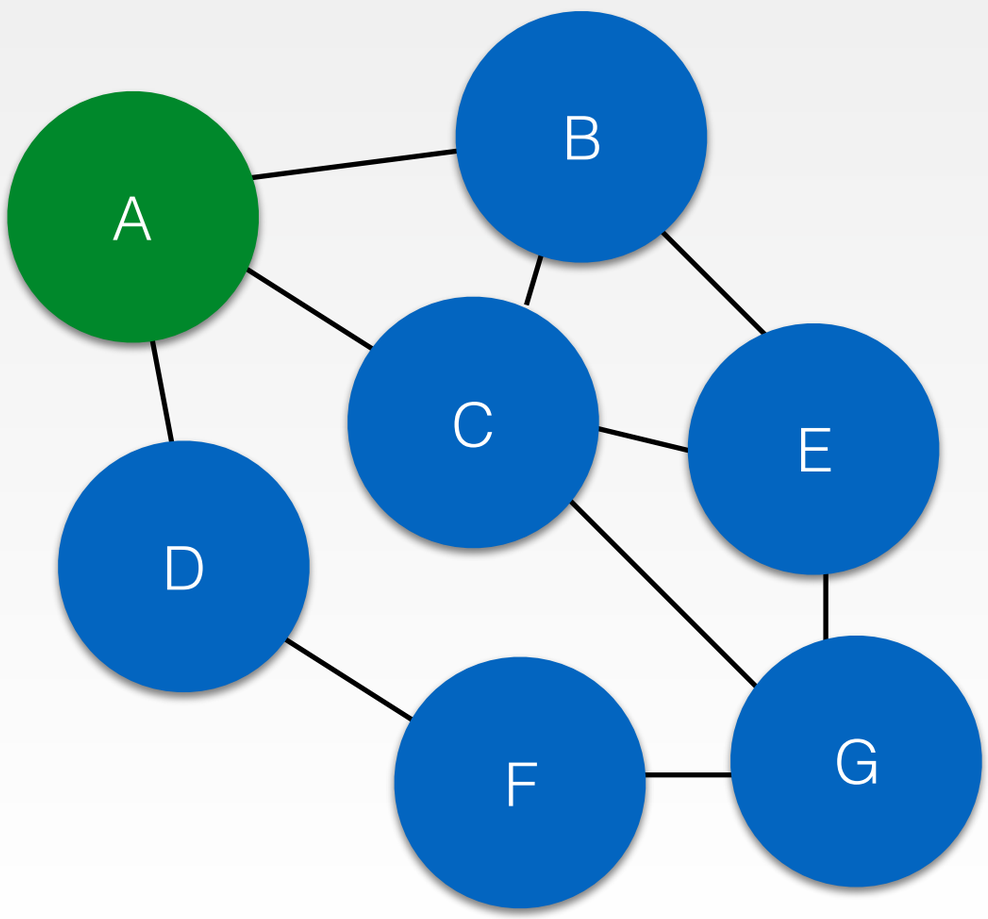
Example LAMP Walk



Current path:

A ← Move from

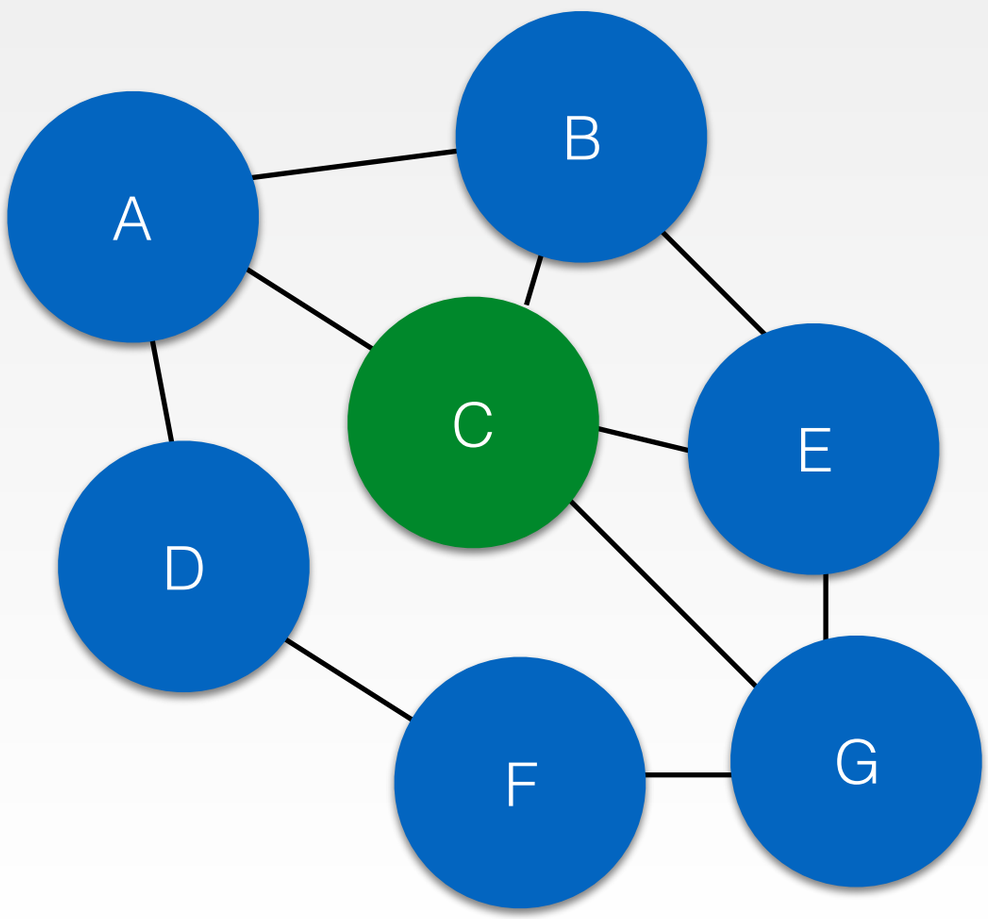
Example LAMP Walk



Current path:

A ← Move from

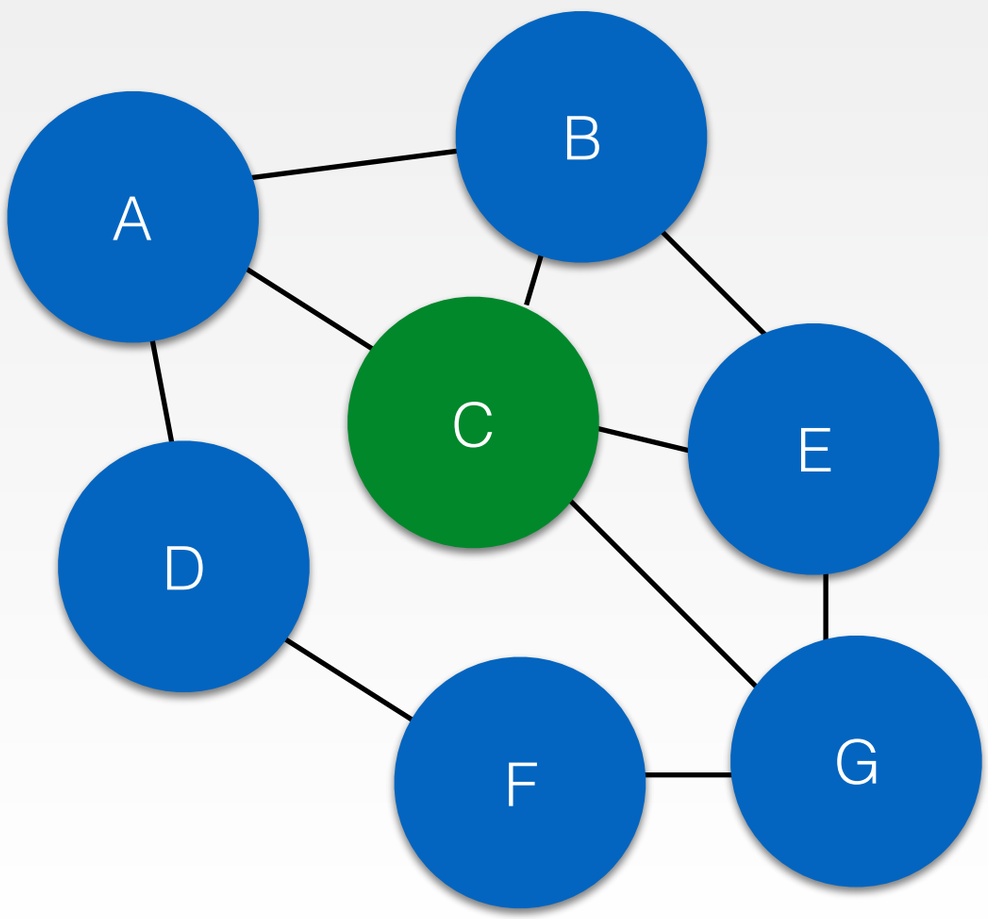
Example LAMP Walk



Current path:

A
C

Example LAMP Walk

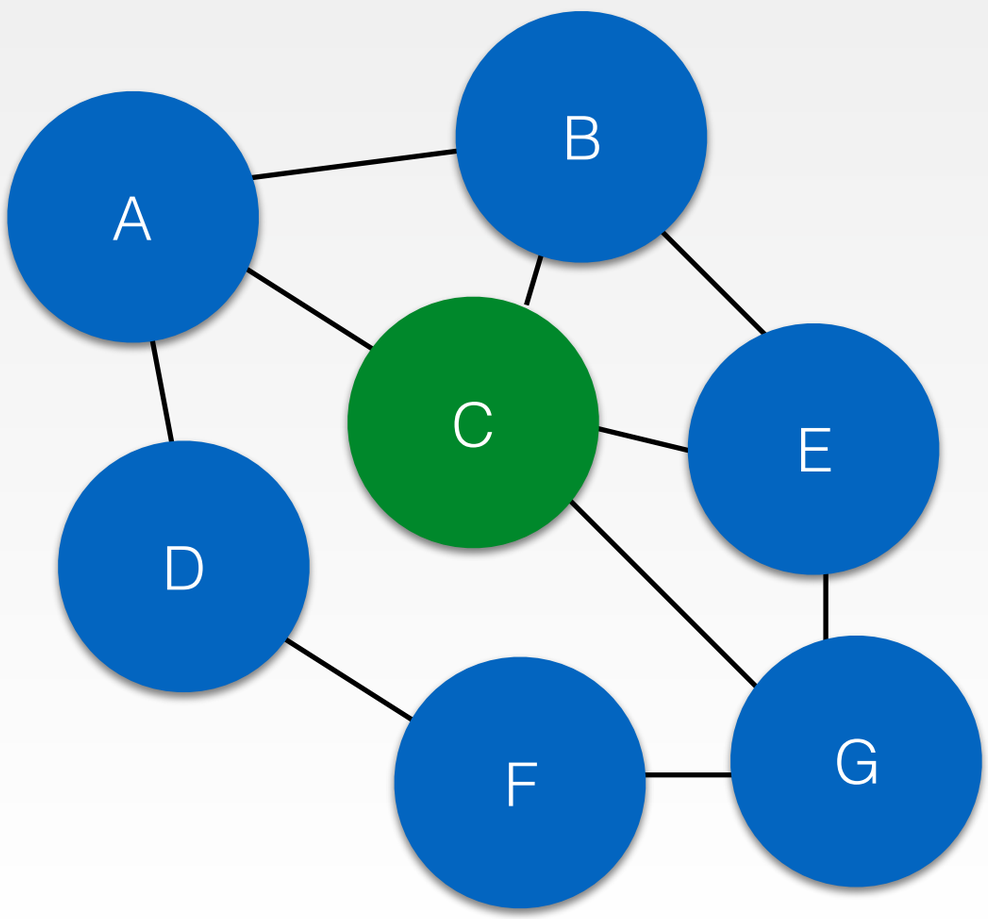


Current path:

A ← Move from

C

Example LAMP Walk

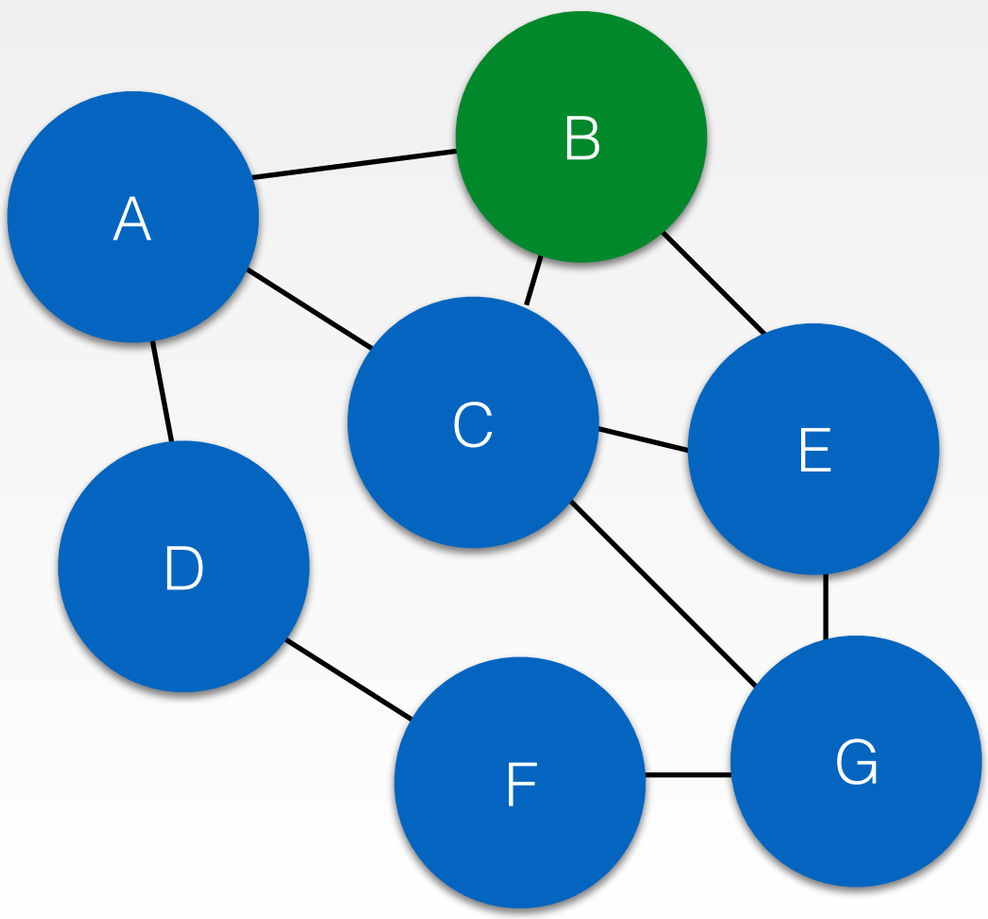


Current path:

A ← Move from

C

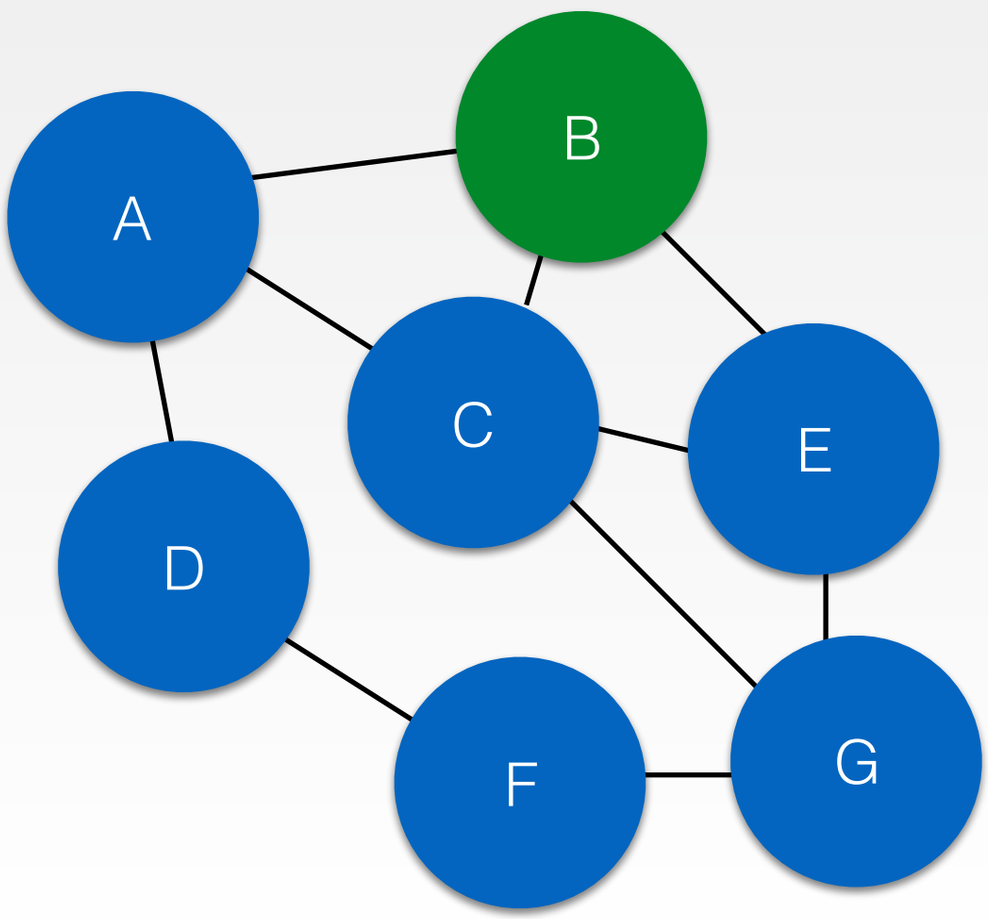
Example LAMP Walk



Current path:

A
C
B

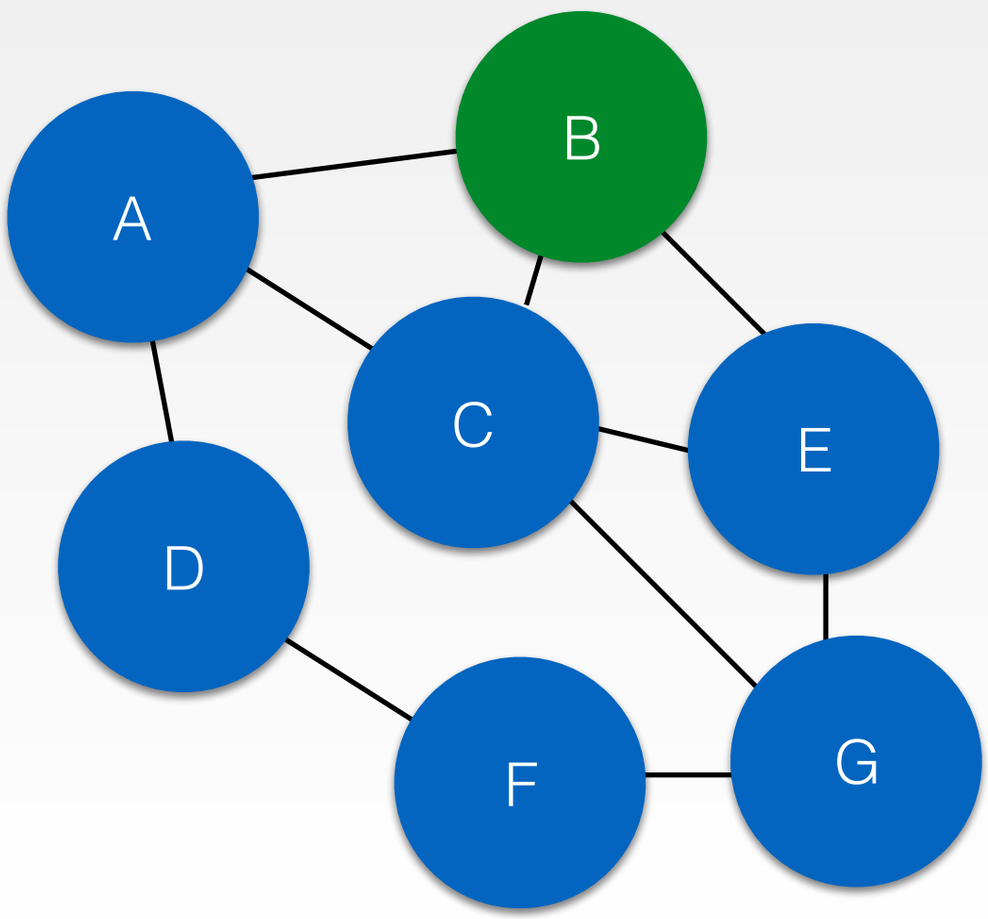
Example LAMP Walk



Current path:

A
C
B ← Move from

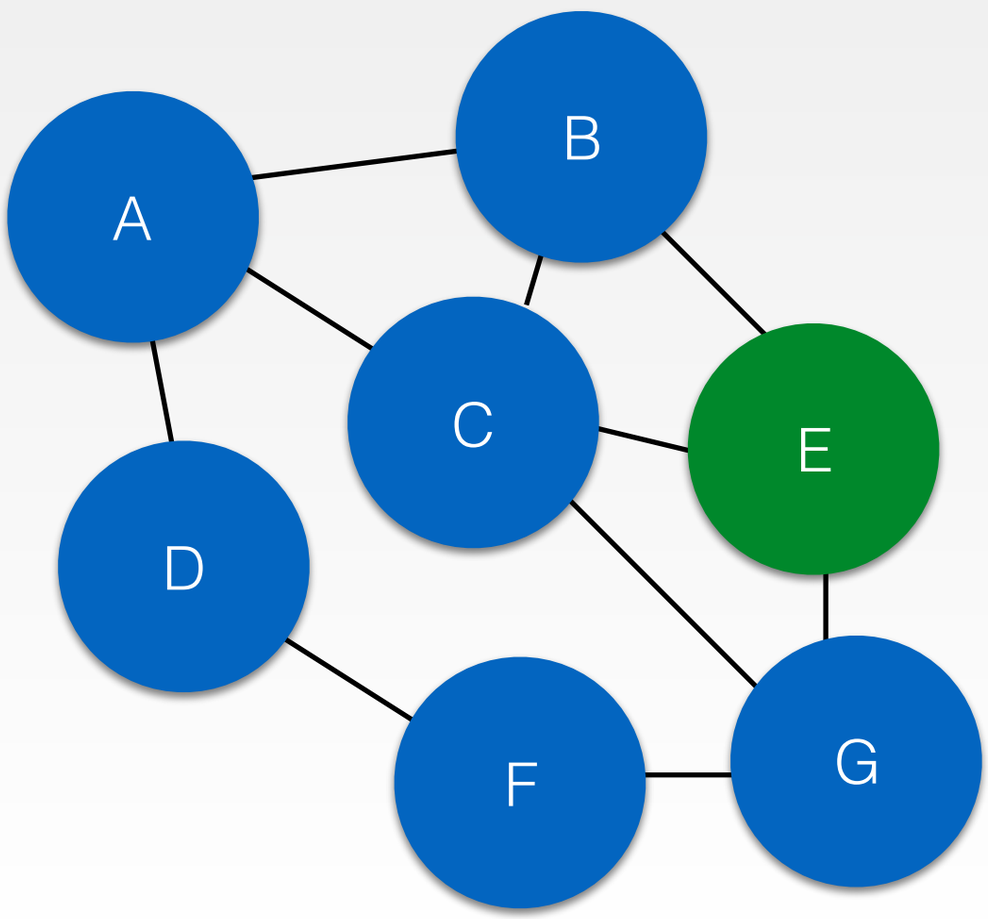
Example LAMP Walk



Current path:

A
C
B ← Move from

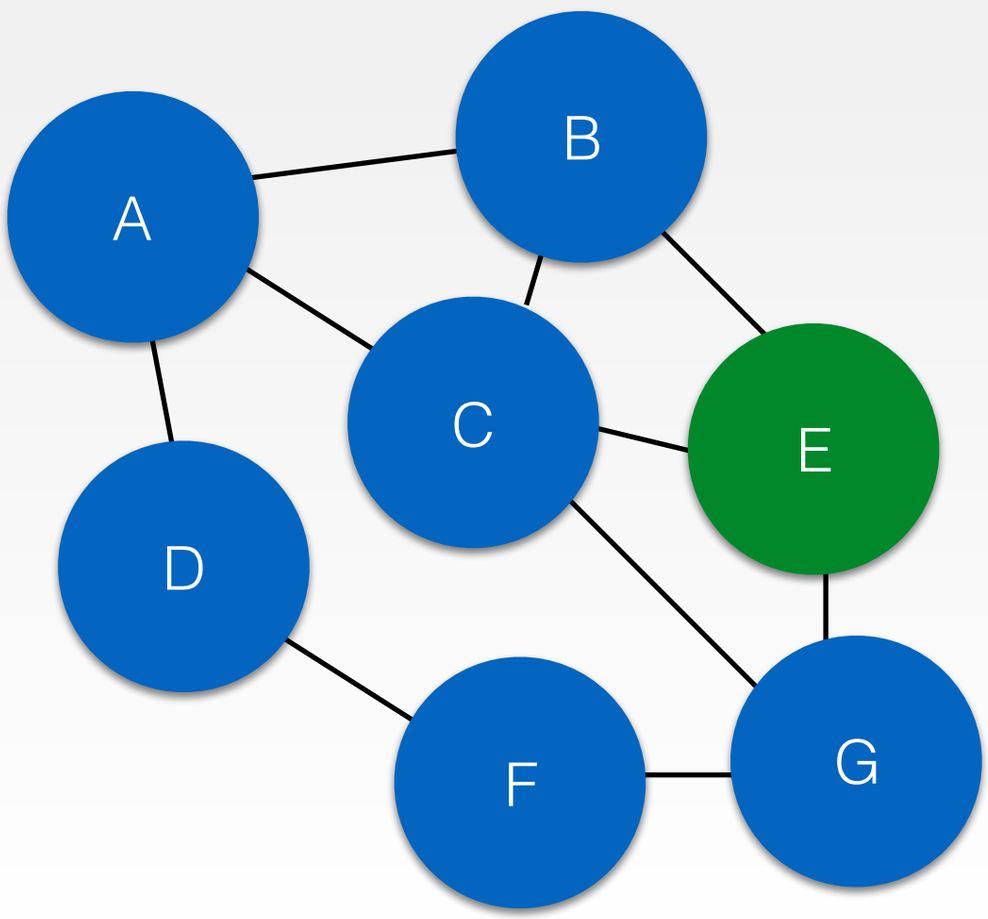
Example LAMP Walk



Current path:

A
C
B
E

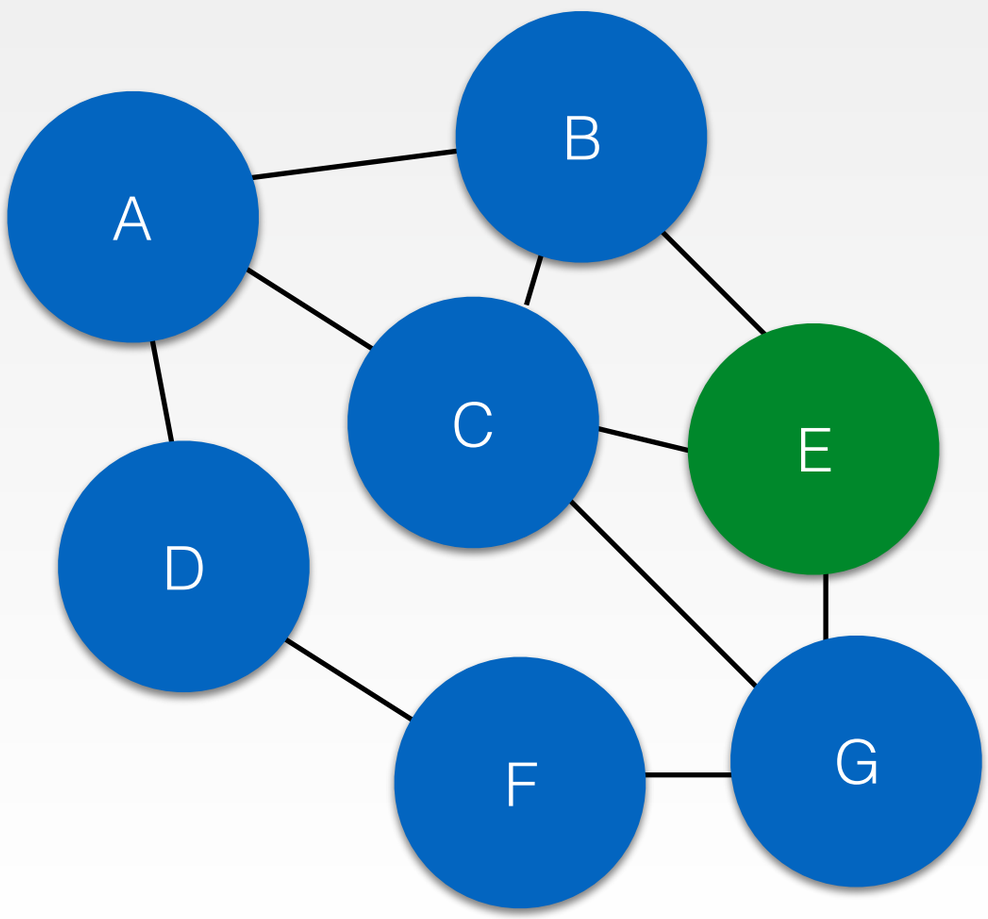
Example LAMP Walk



Current path:

A
C ← Move from
B
E

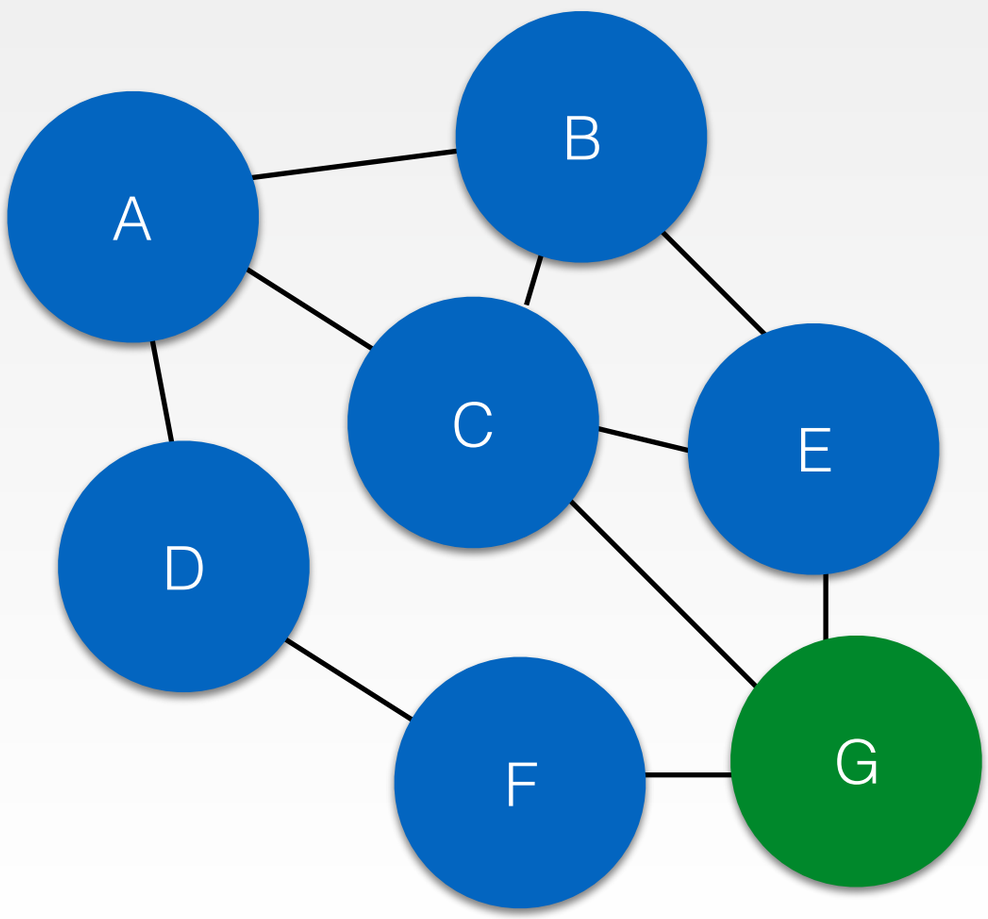
Example LAMP Walk



Current path:

A
C ← Move from
B
E

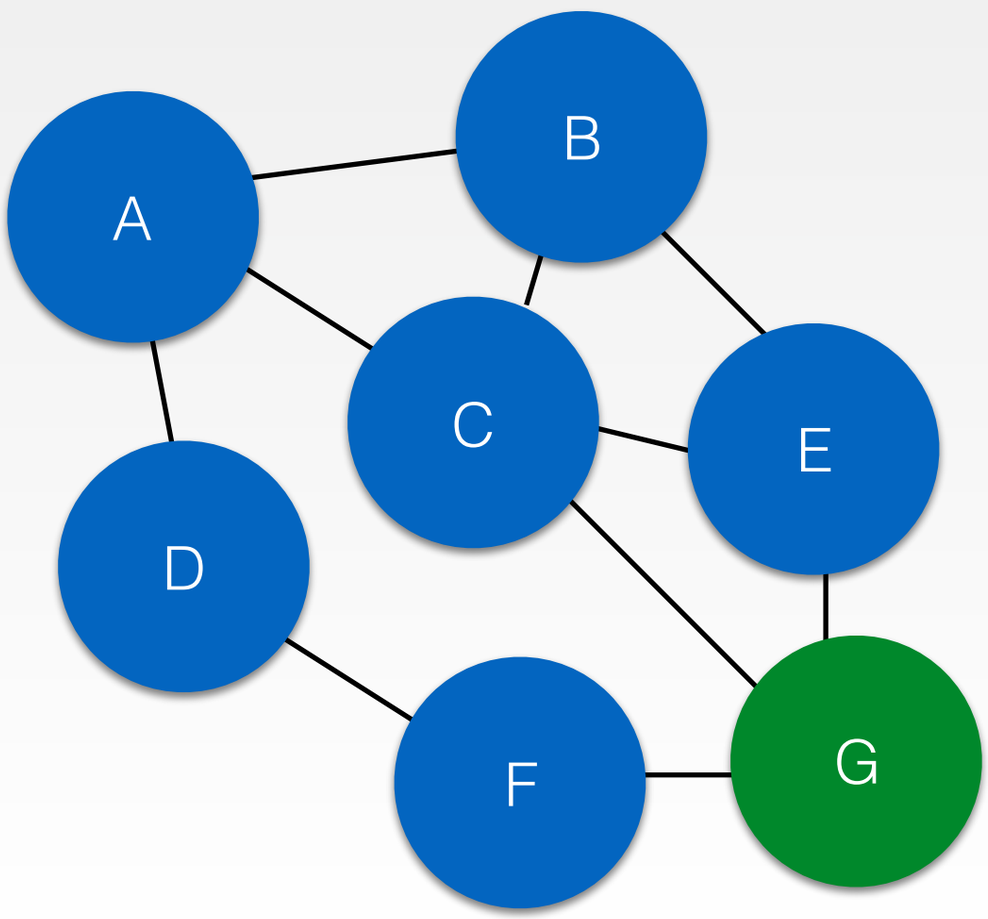
Example LAMP Walk



Current path:

A
C
B
E
G

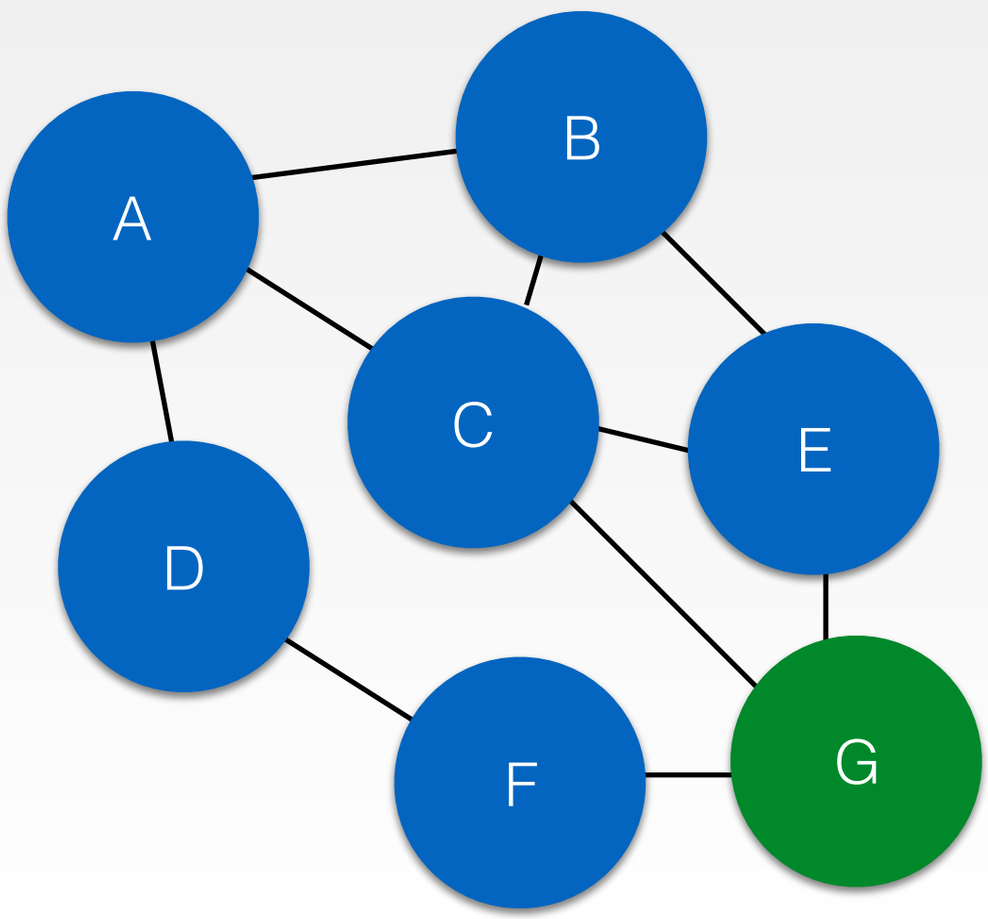
Example LAMP Walk



Current path:

A ← Move from
C
B
E
G

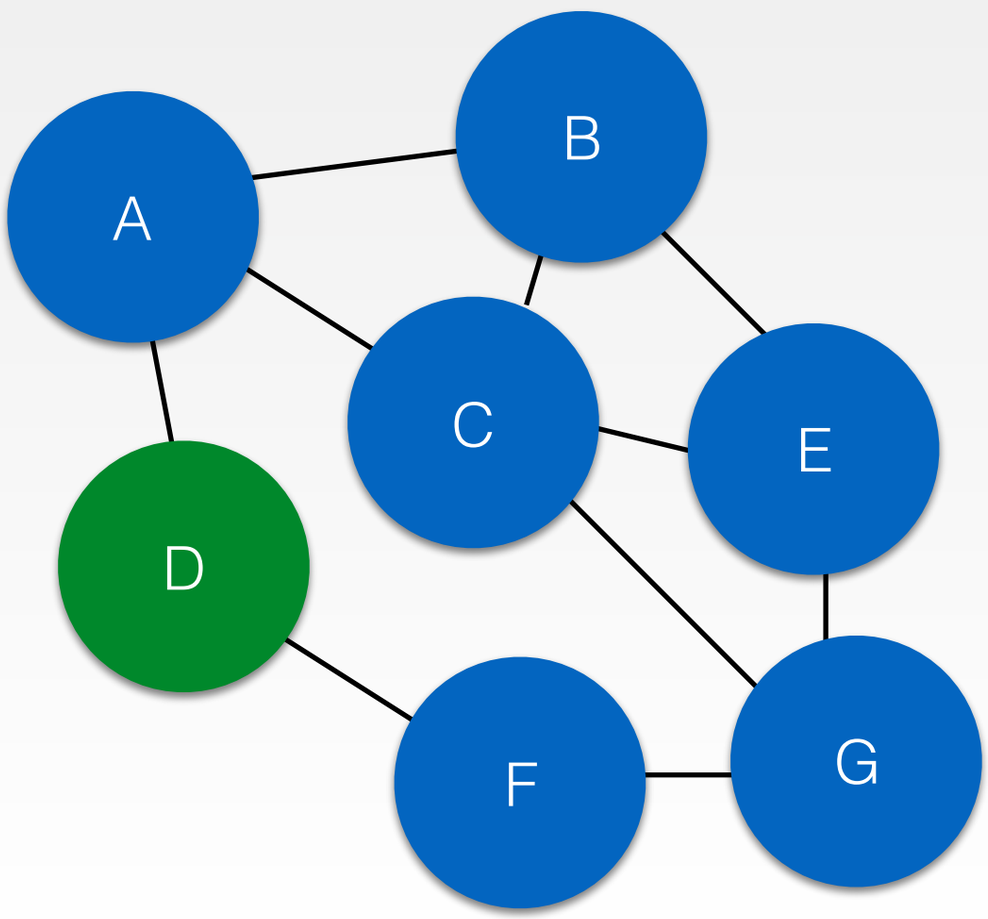
Example LAMP Walk



Current path:

A ← Move from
C
B
E
G

Example LAMP Walk



Current path:

A
C
B
E
G
D

Expressivity and Evolution of LAMP

1. $LAMP_k(w,W)$ cannot be approximated by MM_{k-1}
2. $LAMP_k(w,W)$ is a subset of MM_k

Expressivity and Evolution of LAMP

1. $\mathrm{LAMP}_k(w,W)$ cannot be approximated by MM_{k-1}
2. $\mathrm{LAMP}_k(w,W)$ is a subset of MM_k

State distribution of LAMP at different timesteps:

Time 0

π_0

Expressivity and Evolution of LAMP

1. $\mathrm{LAMP}_k(w,W)$ cannot be approximated by MM_{k-1}
2. $\mathrm{LAMP}_k(w,W)$ is a subset of MM_k

State distribution of LAMP at different timesteps:

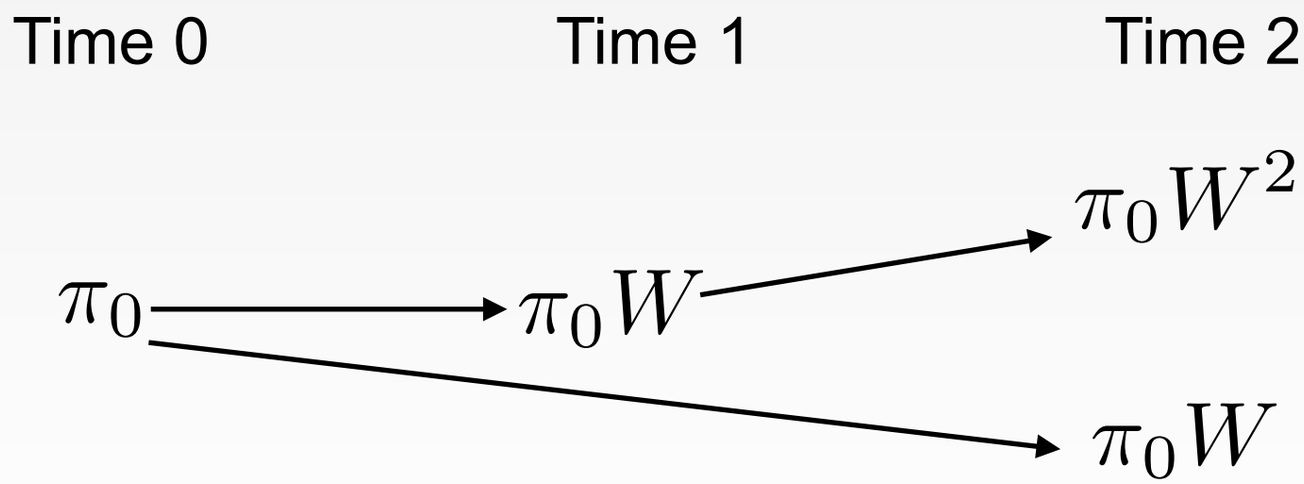
Time 0 Time 1

$$\pi_0 \longrightarrow \pi_0 W$$

Expressivity and Evolution of LAMP

1. $\mathrm{LAMP}_k(w,W)$ cannot be approximated by MM_{k-1}
2. $\mathrm{LAMP}_k(w,W)$ is a subset of MM_k

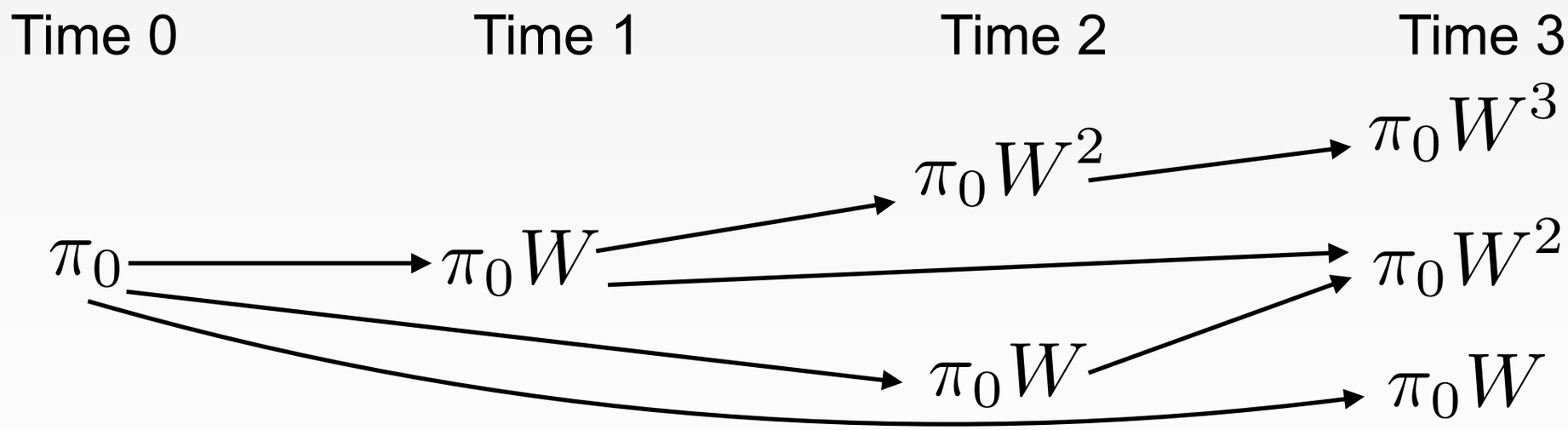
State distribution of LAMP at different timesteps:



Expressivity and Evolution of LAMP

1. $\mathrm{LAMP}_k(w,W)$ cannot be approximated by MM_{k-1}
2. $\mathrm{LAMP}_k(w,W)$ is a subset of MM_k

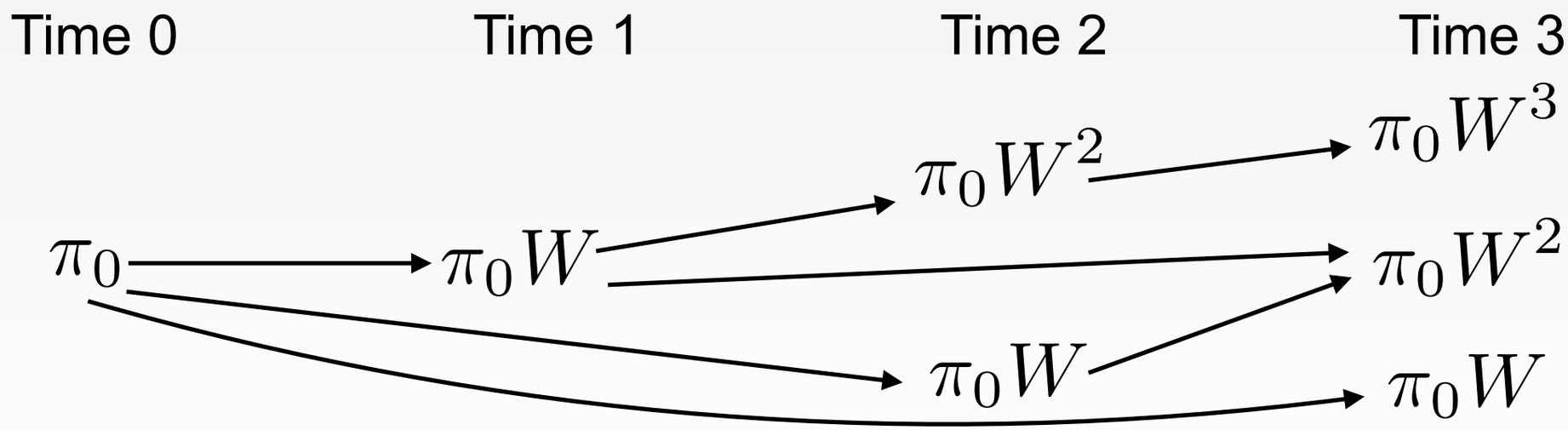
State distribution of LAMP at different timesteps:



Expressivity and Evolution of LAMP

1. $\mathrm{LAMP}_k(w,W)$ cannot be approximated by MM_{k-1}
2. $\mathrm{LAMP}_k(w,W)$ is a subset of MM_k

State distribution of LAMP at different timesteps:



Correct random variable: exponent at time t = e_t
Evolution: pick exponent from previous *k* (according to *w*), add 1 to it.

[See also Wu and Gleich (arXiv)]

Steady State of LAMP

Let $m = \min\{e_{t-k+1}, \ldots, e_t\}$

Note: $e_{t+1} \geq 1 + m$

By induction: $\min\{e_{t+1}, \ldots, e_{t+k}\} \geq 1 + m$

Therefore:

$$e_t \geq \left\lfloor \frac{t}{k} \right\rfloor$$

Conclusion: $LAMP_k(w,W)$ has same steady state as $MM_1(W)$

LAMP has same steady state but different dynamics

Exponent Processes

Look back from exponent at time t

Time:	$t - \sum_{I=1}^{H(t)} W_i$	$\ldots$	$t - W_1 - W_2$	$t - W_1$	t
State:	π_0	$\ldots$	$\pi_0 W^{e_t - 2}$	$\pi_0 W^{e_t - 1}$	$\pi_0 W^{e_t}$

$H(t)$ is a stopping time, when this sum first crosses t

But this is just a renewal process!

Theorem: By Strong Law of Large Numbers for Renewal Processes:

$$\lim_{t \to \infty} H(t) = \frac{t}{E(w)}$$

LAMP Mixing

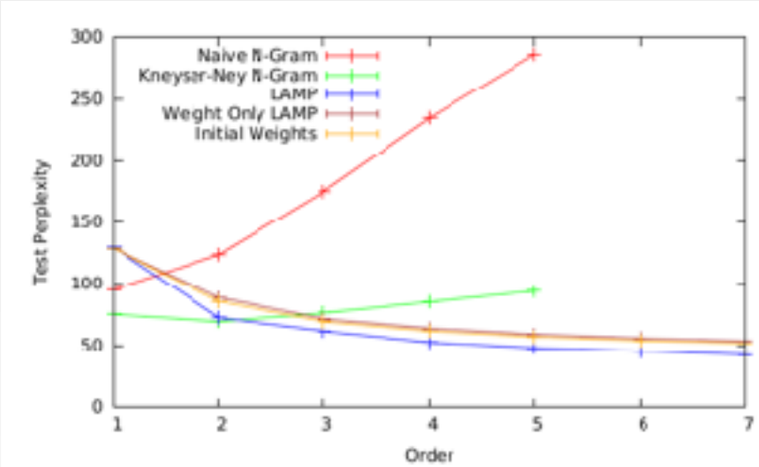
- Can derive concentration bounds
- Gives strong statements on mixing time of LAMP, based on mixing of underlying first-order MM

Data for Evaluation

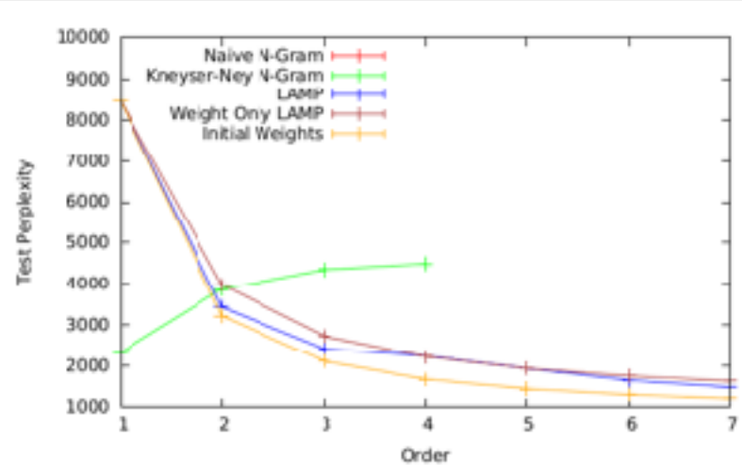
Wikispeedia

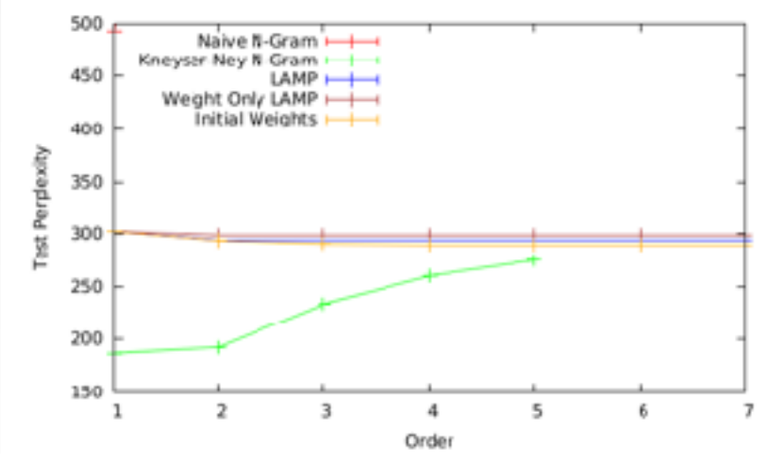
Experiments: Total Perplexity

BrightKite

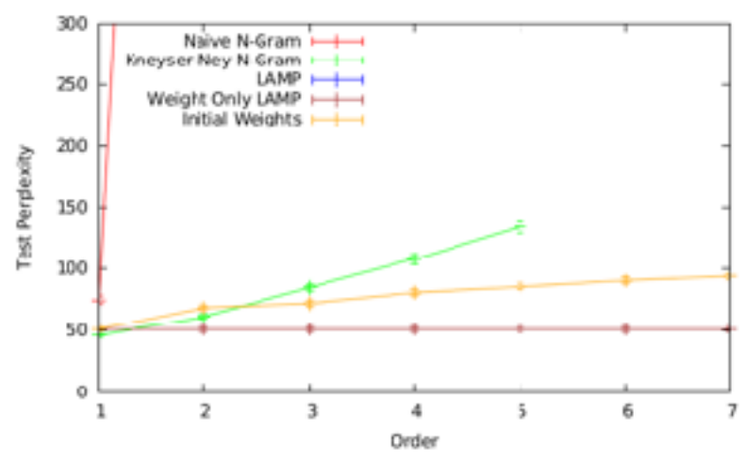


LastFM





Reuters



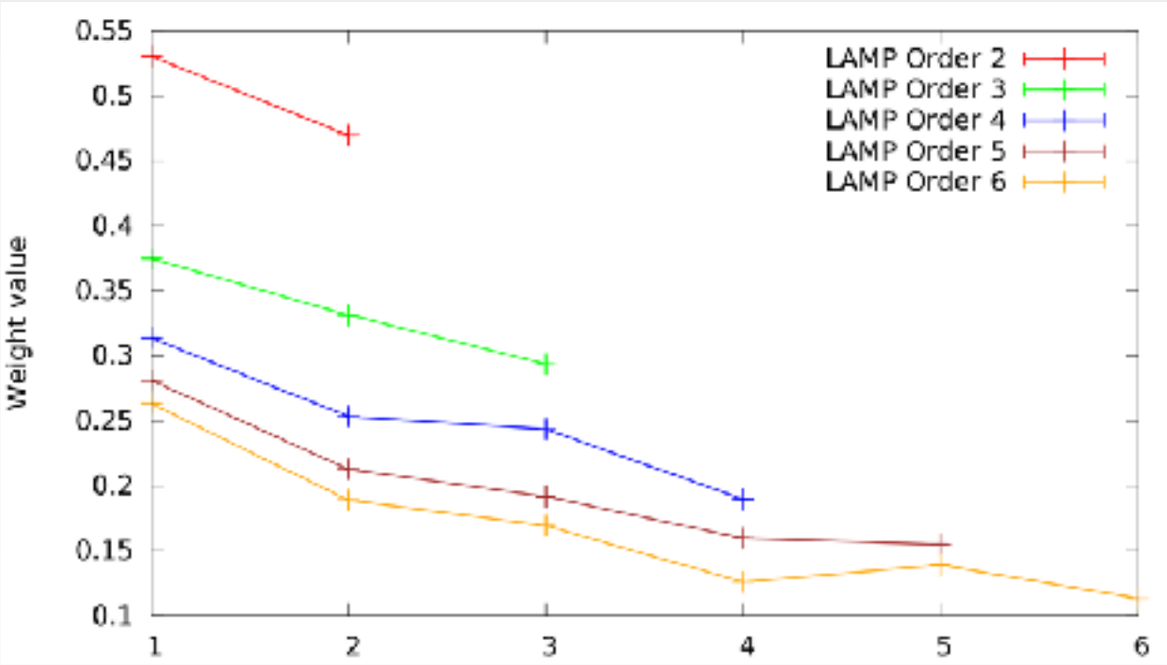
Wikispeedia

Observations

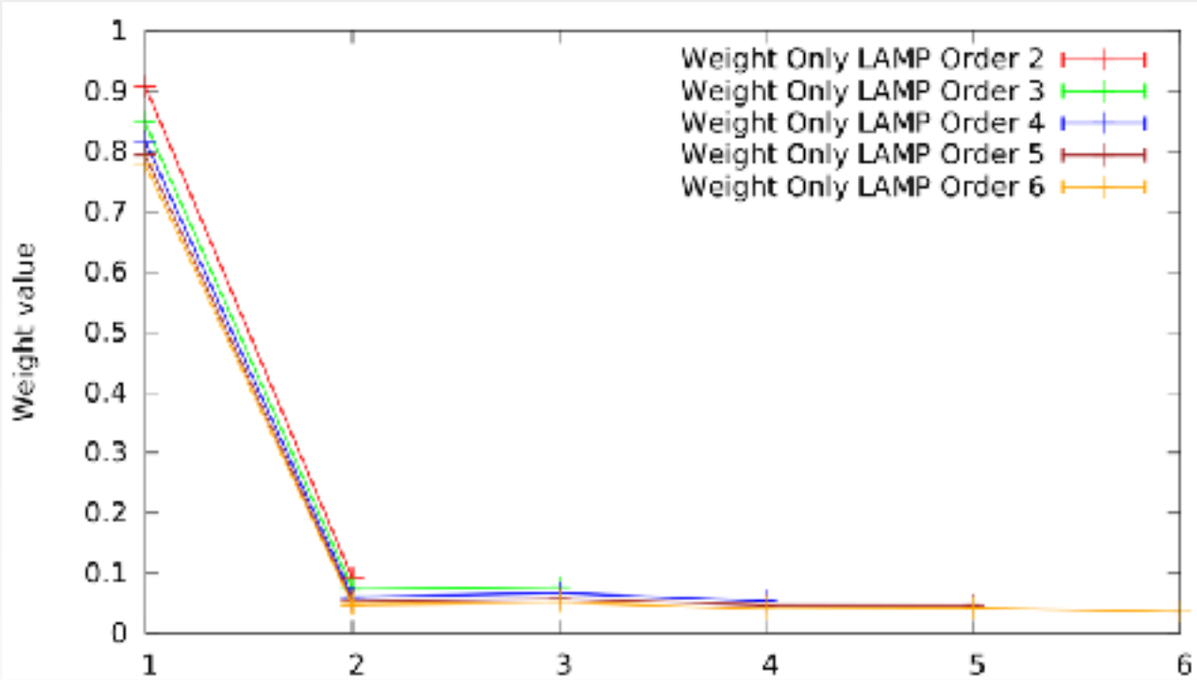
- In general, N-grams and Kneyser-Ney N-grams struggle to use higher order information without overfitting
- Exception is Reuters (text data) which these models have been designed to do better on

Experiments: learned weight distribution

BrightKite



Wikispeedia



- LAMP learns weight decay where useful (BrightKite)
- If history isn't useful (Wikispeedia), then turns into First Order Markov Chain

Experiments

Comparison with LSTMs

Algorithm	BRIGHTKITE	LASTFM	REUTERS
LAMP order 6, 1.5 iter	38.4	1054.6	296.8
LSTM, short training time	85.8	1359.1	105.4
LSTM, long training time	51.0	525.7	60.4

- LAMP does better than LSTM on some datasets (e.g. BrightKite)

- Better or equal performance on other datasets (e.g. LastFM) with similar amounts of training time
  - With 20x training time, LSTM does better
- LSTM does better on text data (better at using text statistics, similar to N-grams)

Reverse Engineering a Markov Chain

Ravi Kumar, Andrew Tomkins, Sergei Vassilvitskii and Erik Vee

[Ref: WSDM 2015]

Random Walks & Markov Chains

Markov Chains in Data Analysis:

- Simple, yet capture a lot of interactions
- Typically: compute & use the stationary distribution
- Beautiful theory with great applications

Examples:

- PageRank: Random surfer stationary distribution
- Translation: Use language models to build phrases
- ...

A Recommendation Chain

Google

markov chain

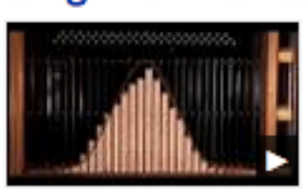
Web Videos Books Images Shopping More Search tools

About 2,250,000 results (0.30 seconds)

Markov chain - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/**Markov_chain** Wikipedia
A **Markov chain** (discrete-time **Markov chain** or DTMC), named after Andrey Markov, is a mathematical system that undergoes transitions from one state to ...
Examples of Markov chains - Andrey Markov - State space - Stochastic matrix

[PDF] Chapter 11, Markov Chains
www.dartmouth.edu/~chance/.../Chapter11.pdf Dartmouth College
Chapter 11. **Markov Chains**. 11.1 Introduction. Most of our study of probability has dealt with independent trials processes. These processes are the basis of ...

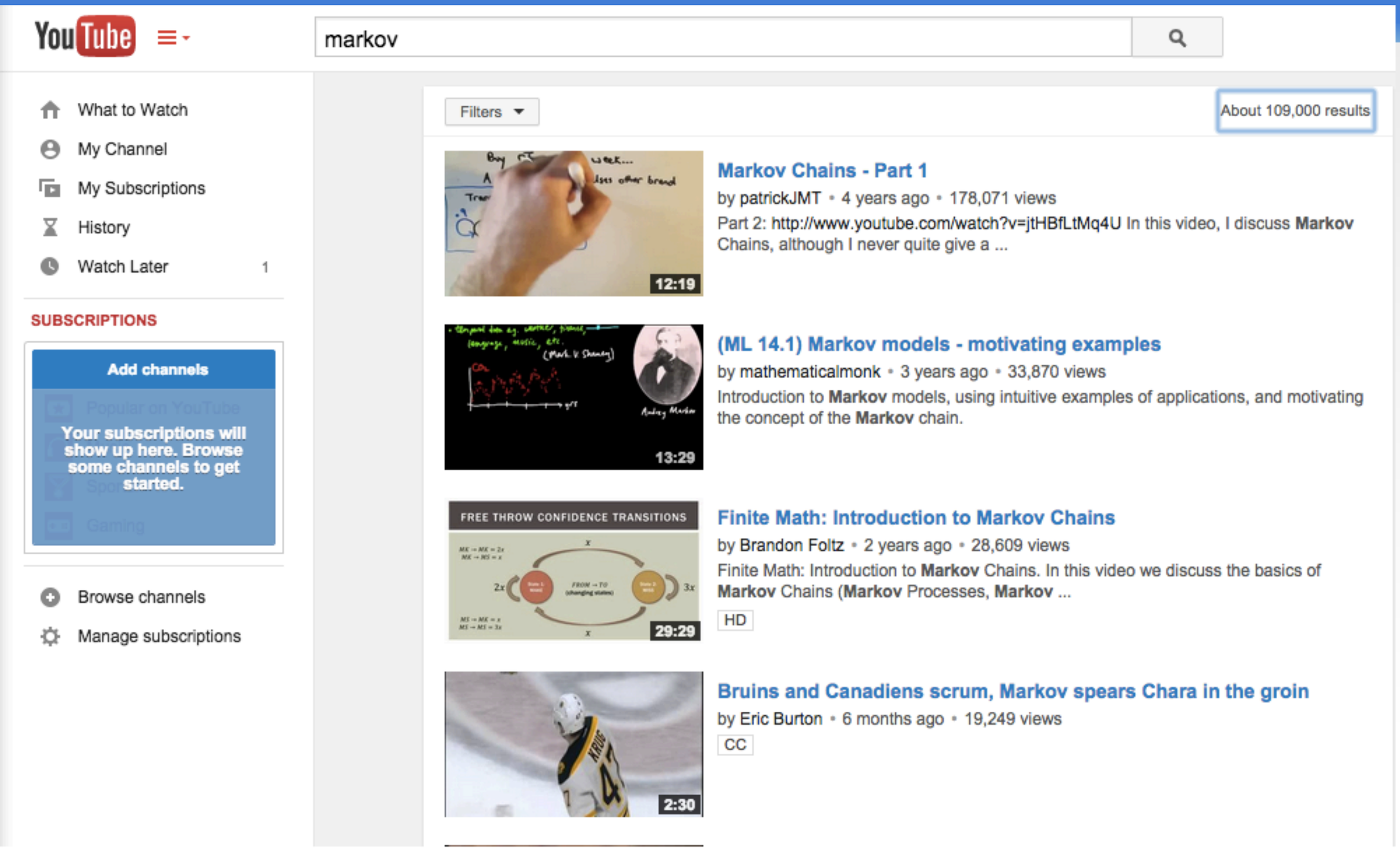
Origin of Markov chains - Khan Academy



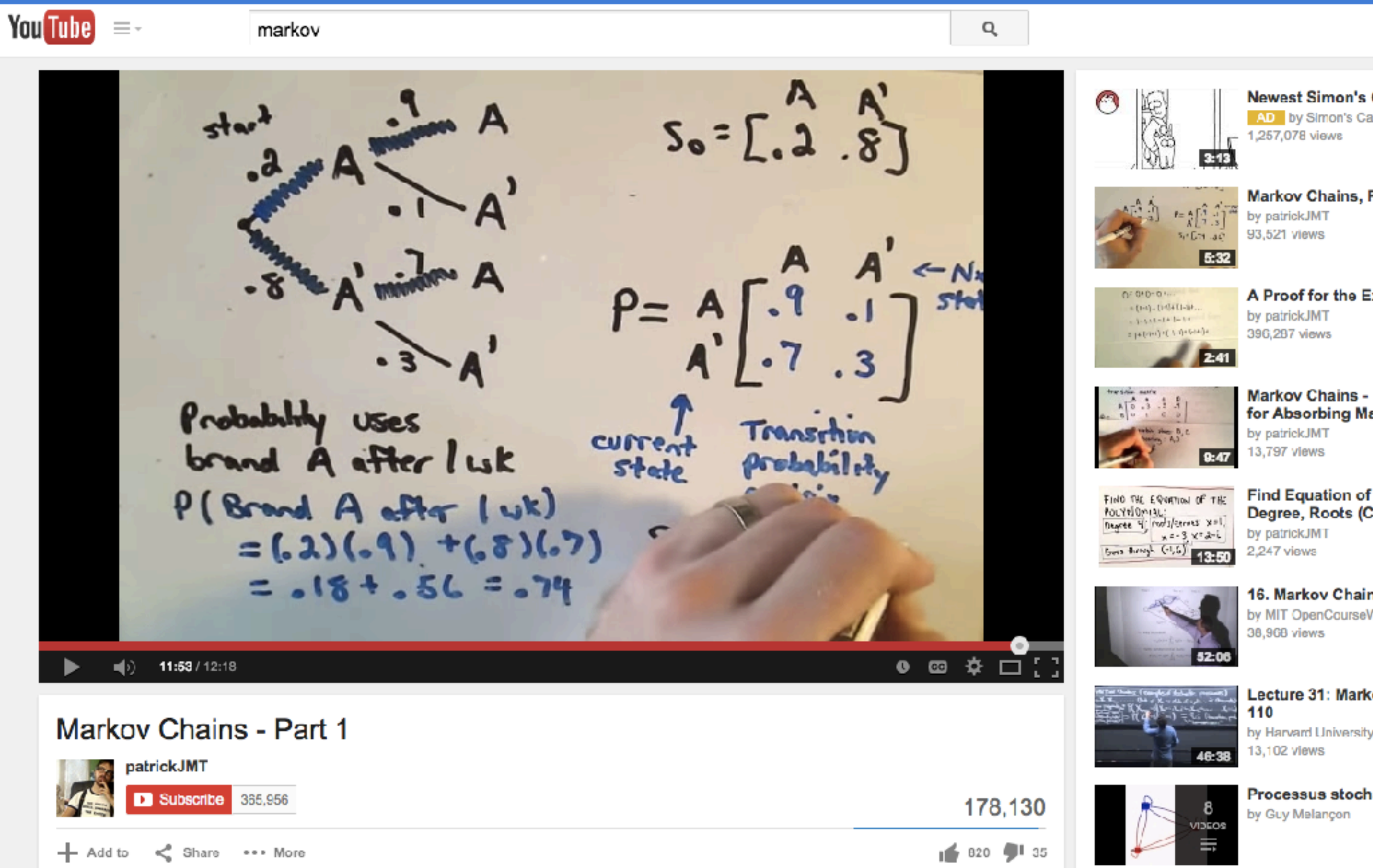
www.khanacademy.org/.../**markov_cha**... Khan Academy
Could **Markov chains** be considered a basis of some (random) cellular automaton? I mean, each **Markov** ...

Markov Chains
setosa.io/blog/2014/07/26/**markov-chains**/
Jul 26, 2014 - **Markov chains**, named after Andrey Markov, are mathematical systems that hop from one "state" (a situation or set of values) to another.

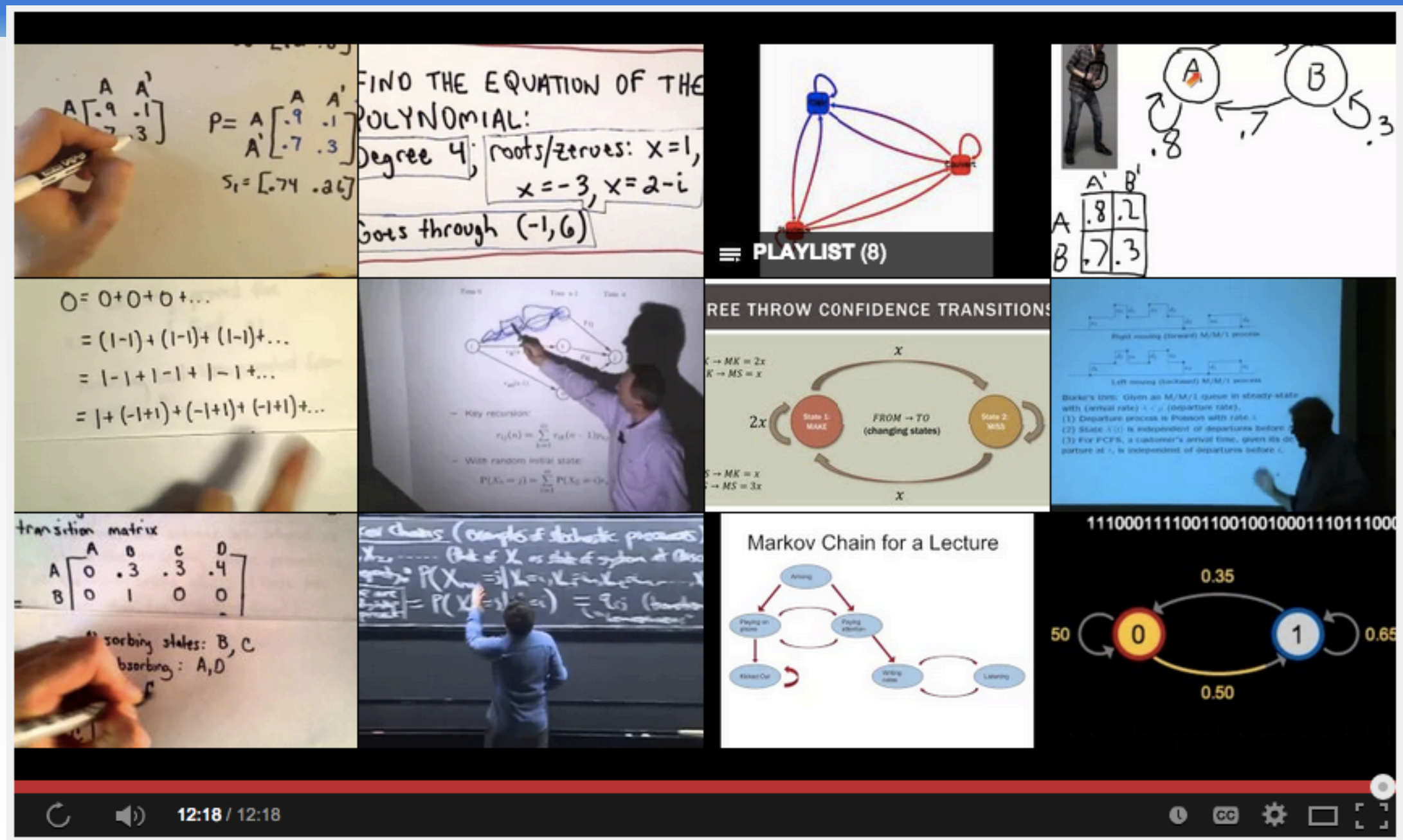
A Recommendation Chain



A Recommendation Chain



A Recommendation Chain



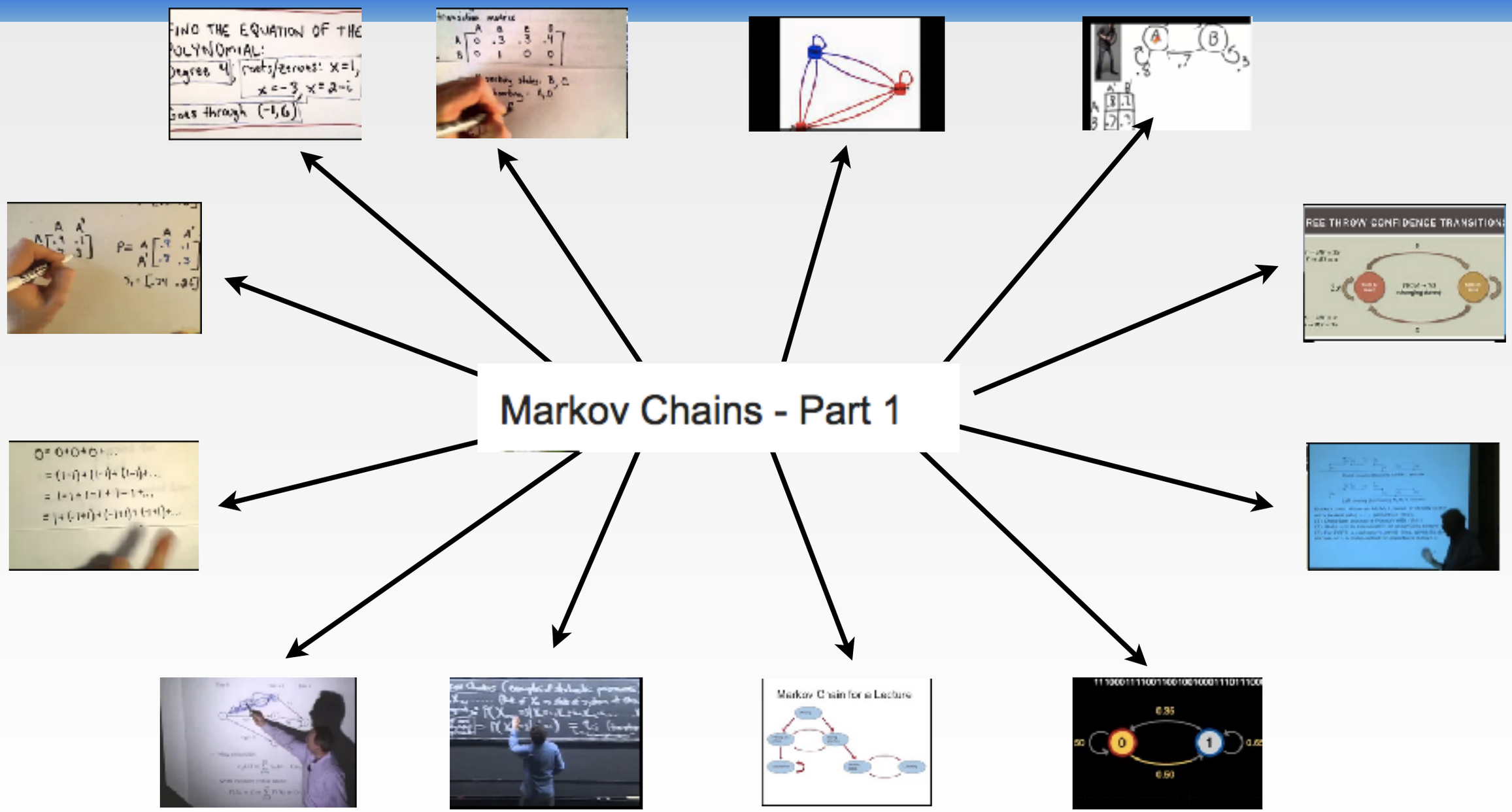
Markov Chains - Part 1

patrickJMT

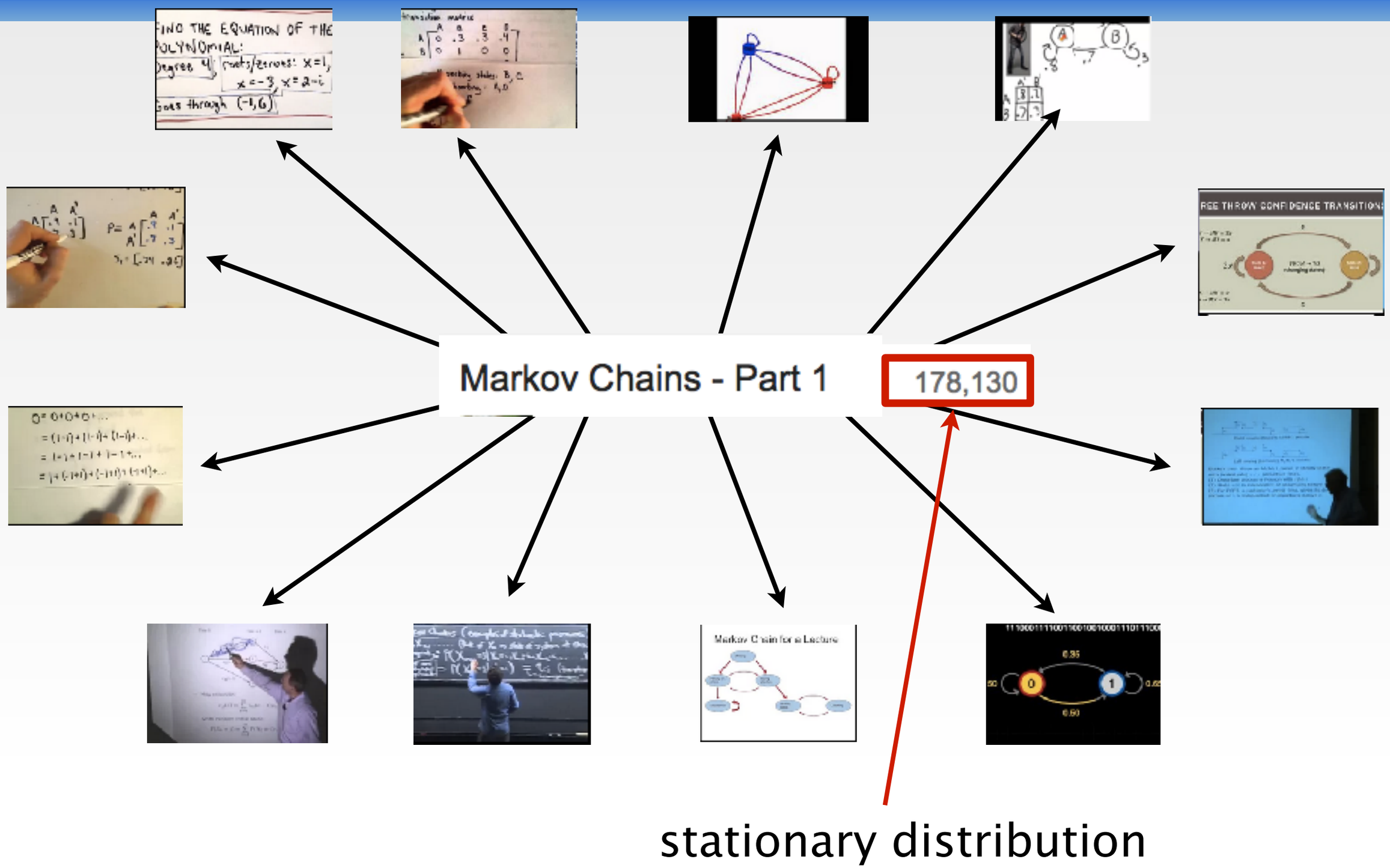
Subscribe 365,956

178,130

A Recommendation Chain



A Recommendation Chain



A Recommendation Chain

Example:

- Items: videos
- Stationary Distribution: view counts

Why are some videos more popular:

- Better (higher quality) videos
- More frequently recommended

Today:

- Disentangle these two reasons

Inverting a Markov Chain

Problem:

- Given a stationary distribution, find the Markov Chain that generated it.

Given:

- Graph G
- Distribution π

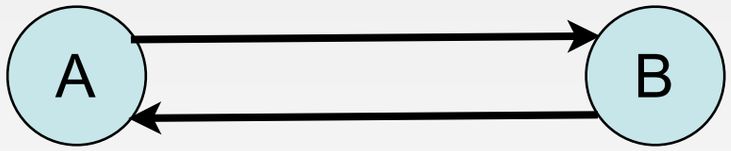
Output:

- Transition Matrix M that generated it

Feasibility

Feasibility:

- Not always feasible



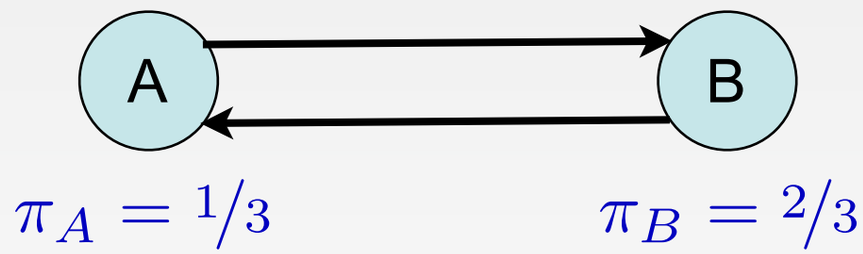
$\pi_A = 1/3$ $\pi_B = 2/3$

π

Feasibility

Feasibility:

- Not always feasible



Definition:

- A directed graph is consistent if there is a flow that preserves the steady state.
- Any strongly connected graph with self loops is consistent

Theorem:

- For any consistent graph, there exists a Markov chain with π as its stationary distribution.

Constraints

The problem is under-constrained:

- n constraints
- $m - n \gg n$ variables

Constraints

The problem is under-constrained:

- n constraints
- $m - n \gg n$ variables

Approaches

- [Tomlin `03]: MaxEnt objective on variables (regularization)

Constraints

The problem is under-constrained:

- n constraints
- $m - n \gg n$ variables

Approaches

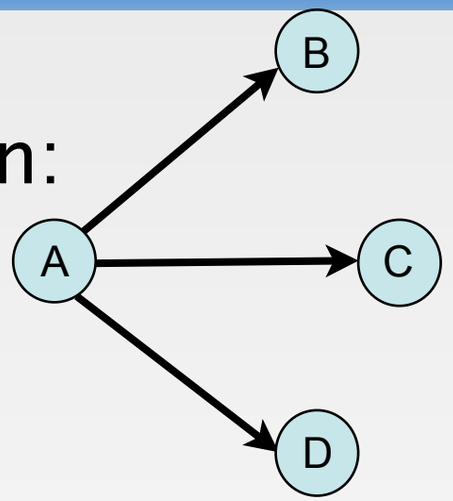
- [Tomlin `03]: MaxEnt objective on variables (regularization)
- [Today] Limit the degrees of freedom

- For each vertex v_i let s_i be its score. The Markov Chain is the function of the scores
- Scores express “quality” or “attractiveness”

From Scores to Transitions

Transition probability $M_{A \to C}$ depends on:

- Score of the destination s_c
- Parameter of the edge w_{AC}

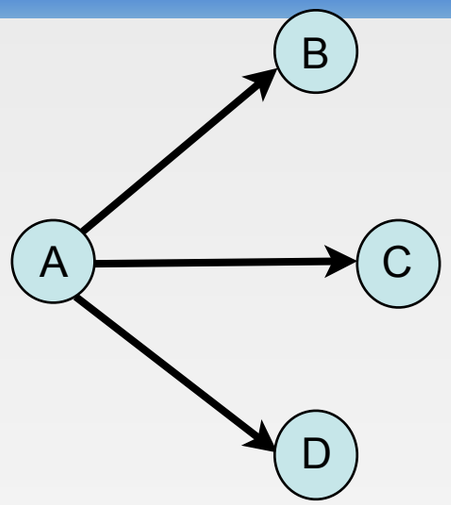


Simplest Example

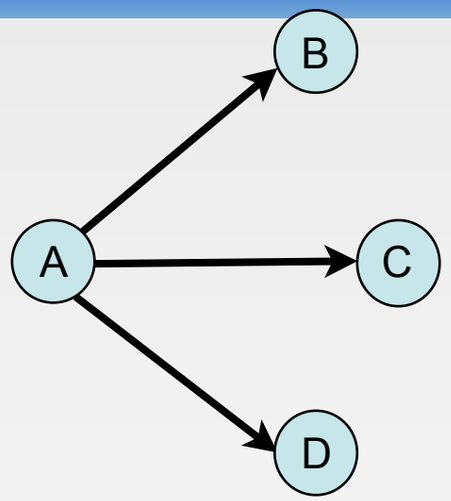
Weighted Random Walk:

- All of the edge weights are set to 1
- Transition probability proportional to the score

$$M_{A \to C} = \frac{s_C}{s_B + s_C + s_D}$$



Simplest Example

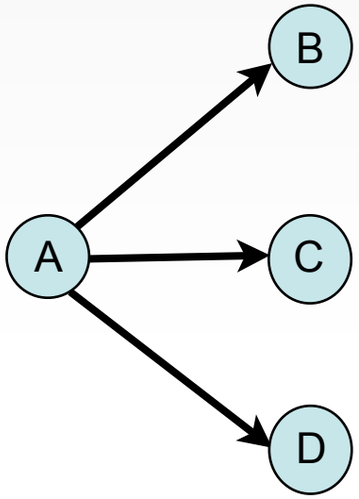


Weighted Random Walk:

- All of the edge weights are set to 1
- Transition probability proportional to the score

$$M_{A \to C} = \frac{s_C}{s_B + s_C + s_D}$$

- Transition probabilities are context dependent:



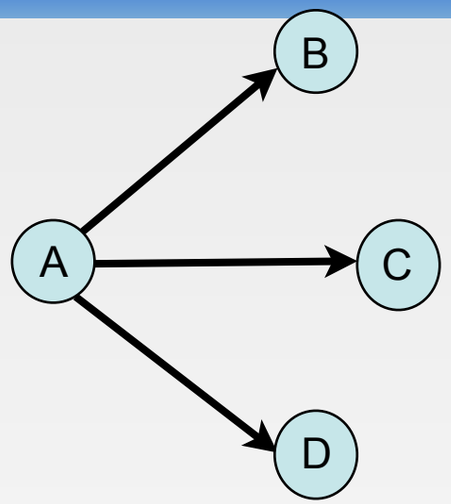
$s_B = 100$

$s_C = 10$

$s_D = 1$

$M_{A \to C} = 0.09$

Simplest Example

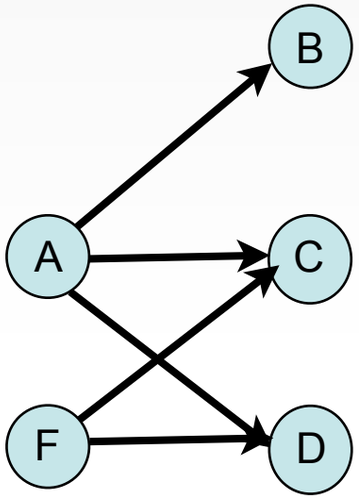


Weighted Random Walk:

- All of the edge weights are set to 1
- Transition probability proportional to the score

$$M_{A \to C} = \frac{s_C}{s_B + s_C + s_D}$$

- Transition probabilities are context dependent:



$s_B = 100$

$s_C = 10$

$s_D = 1$

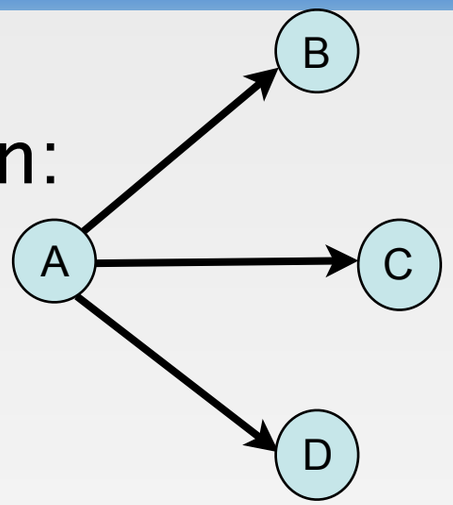
$M_{A \to C} = 0.09$

$M_{F \to C} = 0.91$

From Scores to Transitions

Transition probability $M_{A \to C}$ depends on:

- Score of the destination s_c
- Parameter of the edge w_{AC}
- Call this function f



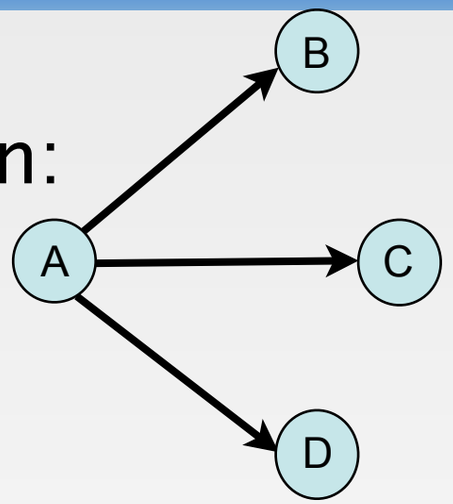
Formally: $M_{A \to C} \propto f(s_C, w_{AC})$

$$M_{A \to C} = \frac{f(s_C, w_{AC})}{f(s_C, w_{AC}) + f(s_B, w_{AB}) + f(s_D, w_{AD})}$$

From Scores to Transitions

Transition probability $M_{A \to C}$ depends on:

- Score of the destination s_c
- Parameter of the edge w_{AC}
- Call this function f



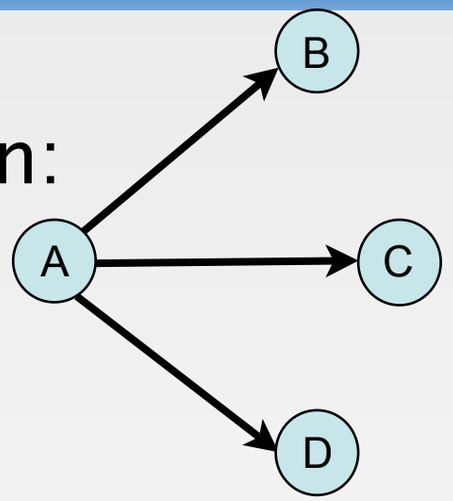
Formally: $M_{A \to C} \propto f(s_C, w_{AC})$

Sanity Check on f:

- Continuous in s
- Monotone in s

From Scores to Transitions

Transition probability $M_{A \to C}$ depends on:



- Score of the destination s_c
- Parameter of the edge w_{AC}
- Call this function f

Formally: $M_{A \to C} \propto f(s_C, w_{AC})$

Sanity Check on f:

- Continuous in s
- Monotone in s
- Unbounded in s : $\lim_{s \to \infty} f(s, w) \to \infty$

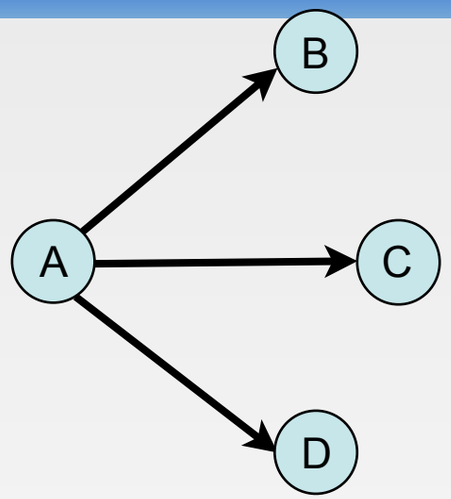
$$\lim_{s_c \to \infty} M_{A \to C} = 1$$

Simplest Example

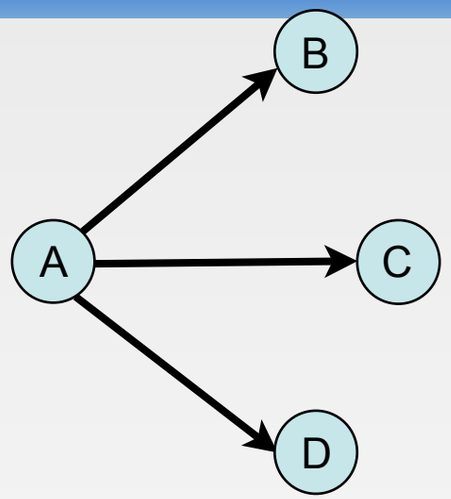
Weighted Random Walk:

- All of the edge weights are set to 1
- Transition probability proportional to the score

$$M_{A \to C} = \frac{s_C}{s_B + s_C + s_D}$$



More Examples



Weighted Random Walk:

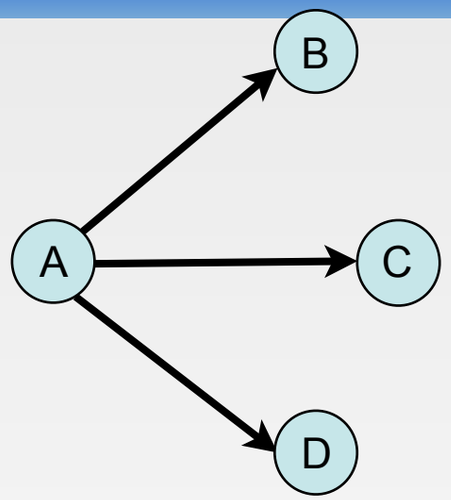
- All of the edge weights are set to 1
- Transition probability proportional to the score

$$M_{A \to C} = \frac{s_C}{s_B + s_C + s_D}$$

Seeking Similar Content:

- Edge weight: similarity between two nodes
- $M_{A \to C} \propto w_{AC} \cdot s_C$

More Examples



Weighted Random Walk:

- All of the edge weights are set to 1
- Transition probability proportional to the score

$$M_{A \to C} = \frac{s_C}{s_B + s_C + s_D}$$

Seeking Similar Content:

- Edge weight: similarity between two nodes
- $M_{A \to C} \propto w_{AC} \cdot s_C$

Overall:

- Decide whether items are popular due to high scores (attract all of the incoming traffic) or due to location (attract a little bit from many locations)

Main Theorem

Given:

- A consistent input G, π
- Monotone, continuous and unbounded function f

There exists:

- A unique set of scores $s_1, \ldots, s_n$
- So that π is the stationary distribution induced by f
- Moreover, the scores can be found in polynomial time

Main Theorem

Given:

- A consistent input G, π
- Monotone, continuous and unbounded function f

There exists:

- A unique set of scores $s_1, \ldots, s_n$
- So that π is the stationary distribution induced by f
- Moreover, the scores can be found in polynomial time

up to scaling

Main Theorem

Given:

- A consistent input G, π
- Monotone, continuous and unbounded function f

There exists:

- A unique set of scores $s_1, \ldots, s_n$
- So that π is the stationary distribution induced by f
- Moreover, the scores can be found in polynomial time

up to scaling

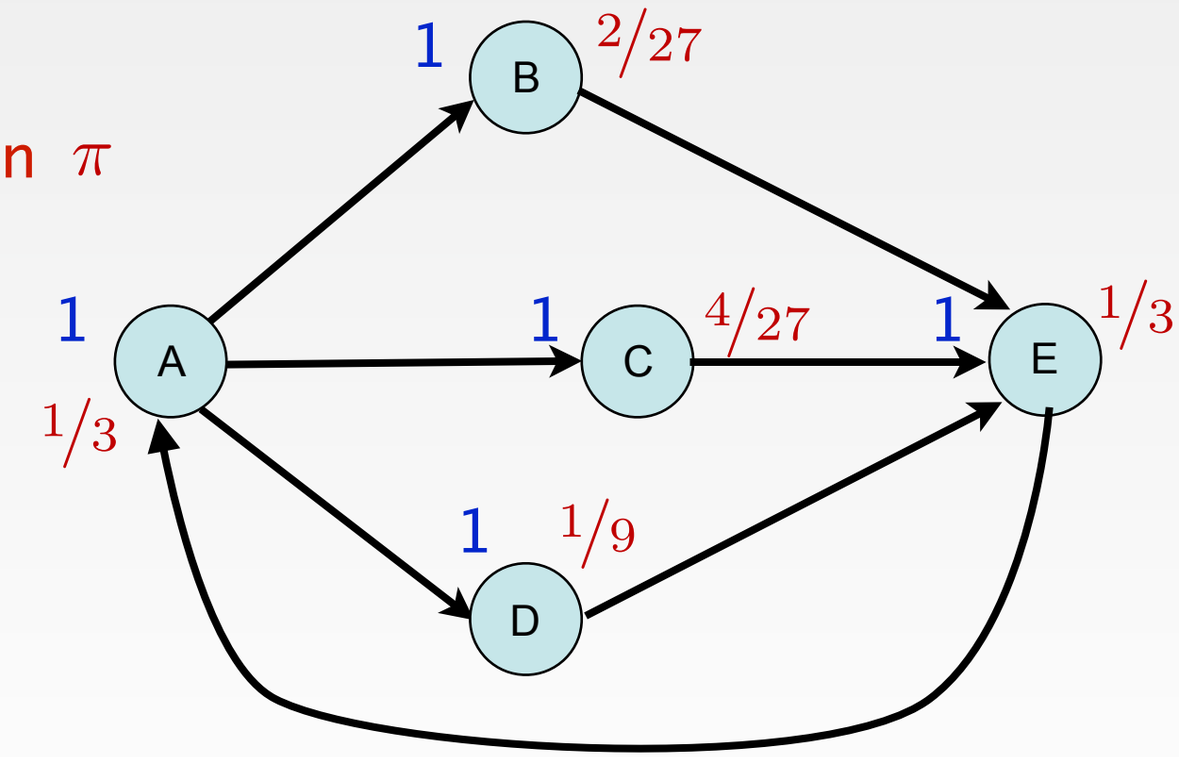
up to $(1 \pm \epsilon)$

Definitions

- Fix a set of scores s and permutation π

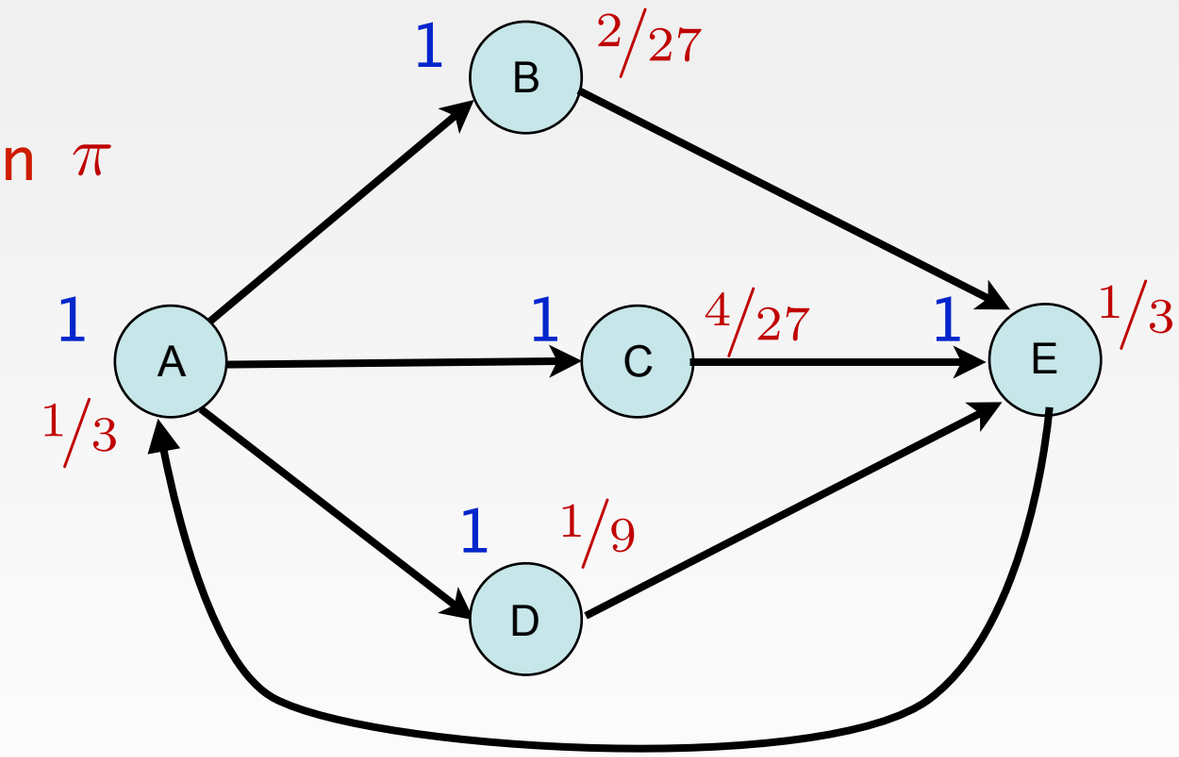
Definitions

- Fix a set of scores s and permutation π



Definitions

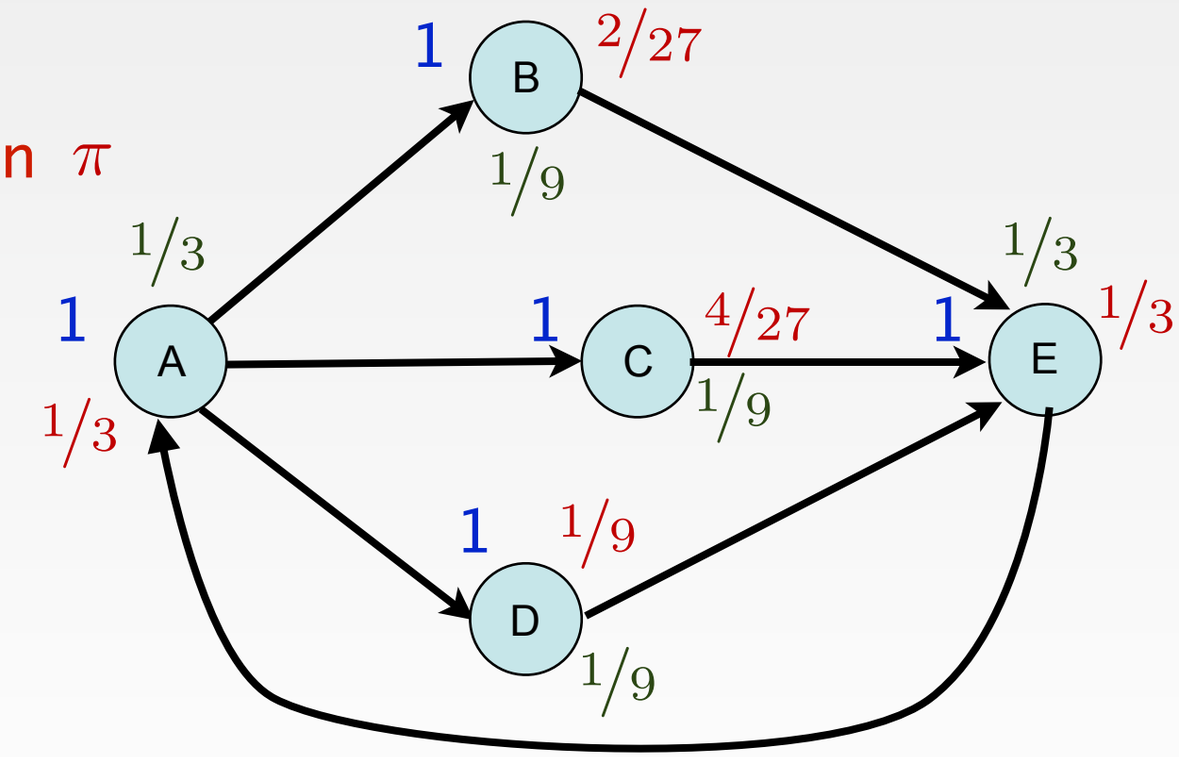
- Fix a set of scores s and permutation π
- Let $q_i(s)$ be the expected mass at v_i starting with π using s



-

Definitions

- Fix a set of scores s and permutation π
- Let $q_i(s)$ be the expected mass at v_i starting with π using s

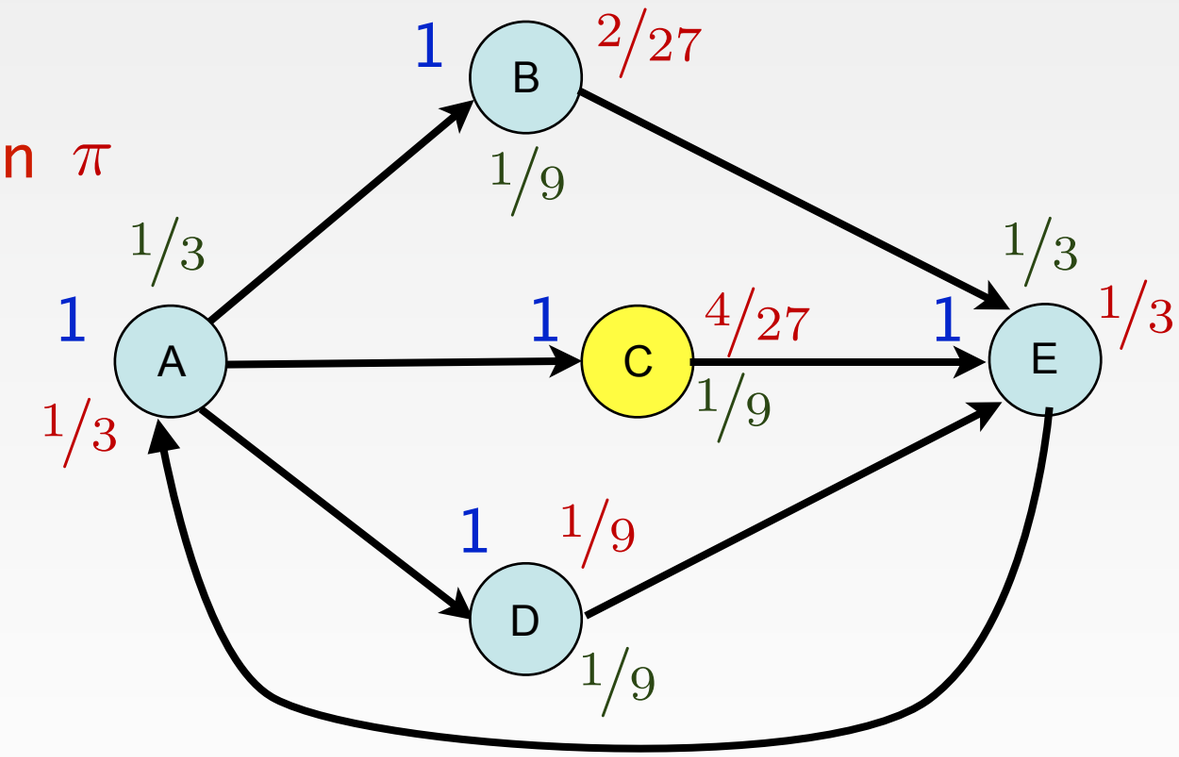


-

Definitions

- Fix a set of scores s and permutation π
- Let $q_i(s)$ be the expected mass at v_i starting with π using s
- Call a node underweight if

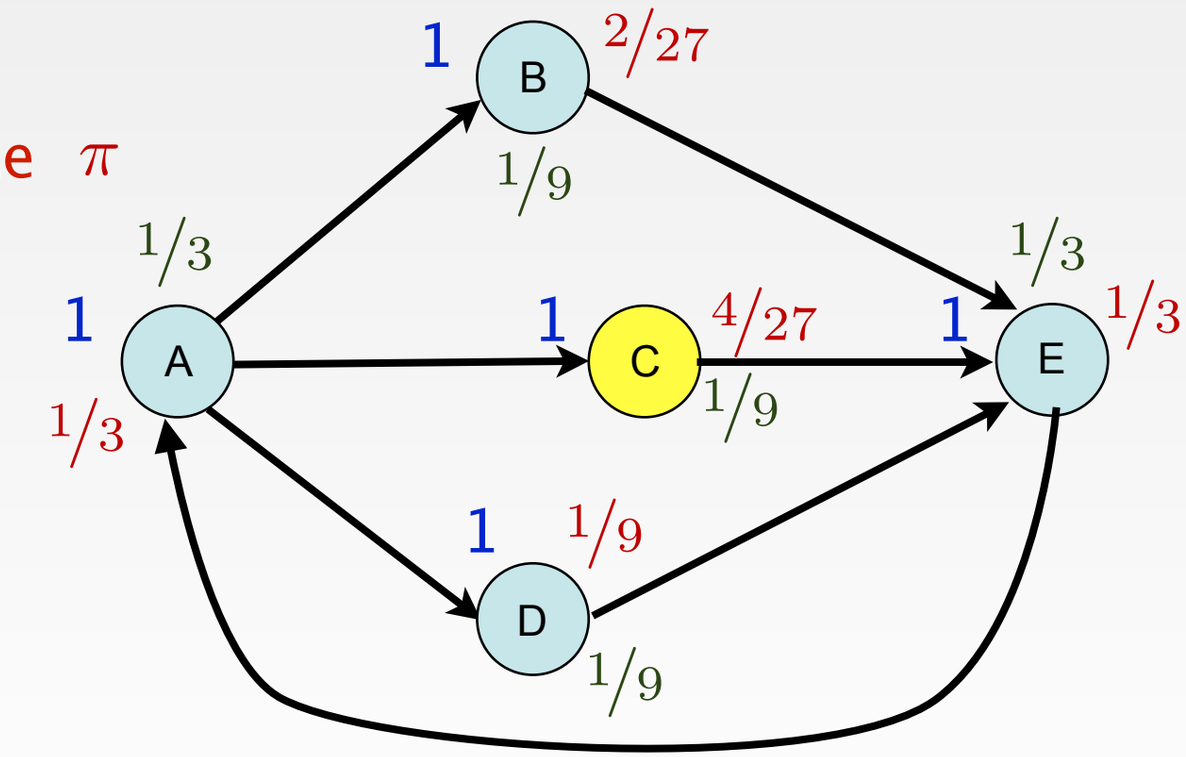
$$q_i(s) < (1-\epsilon)\pi_i$$



Definitions

- Fix a set of scores s and steady state π
- Let $q_i(s)$ be the expected mass at v_i starting with π using s
- Call a node underweight if

$$q_i(s) < (1-\epsilon)\pi_i$$

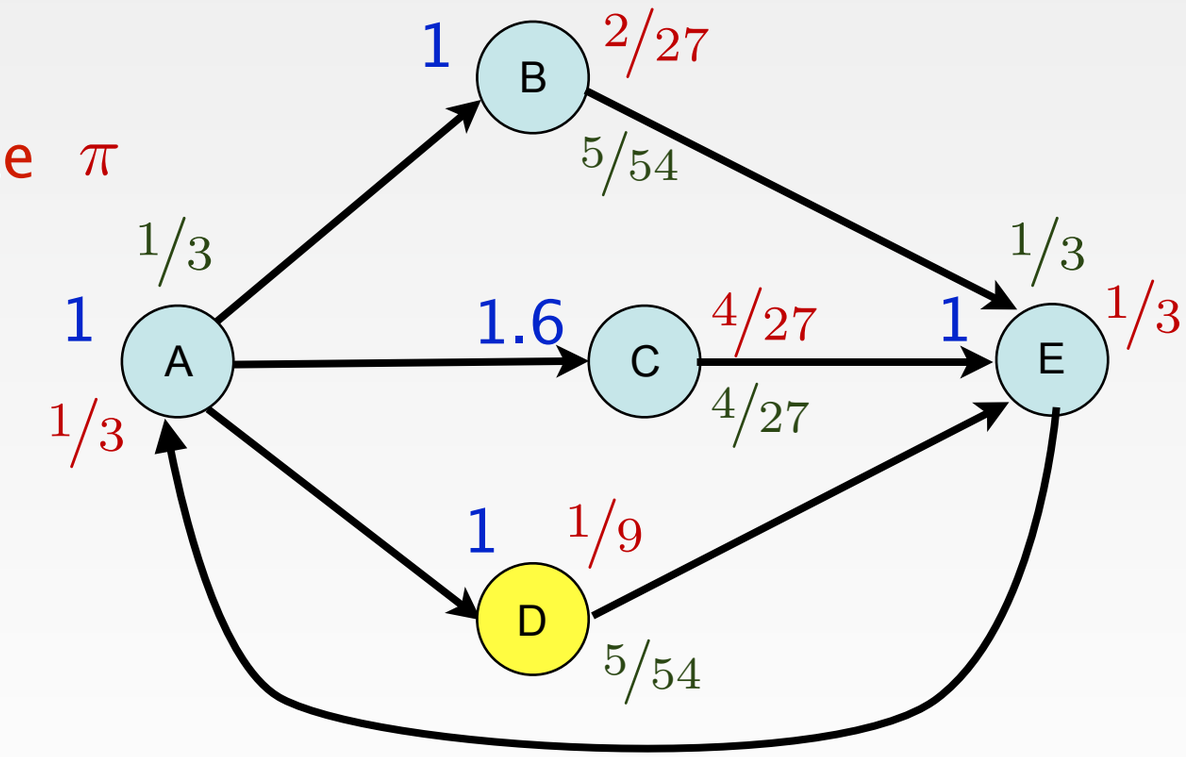


- Algorithm:
  - Repeatedly increase scores of underweight nodes

Definitions

- Fix a set of scores s and steady state π
- Let $q_i(s)$ be the expected mass at v_i starting with π using s
- Call a node underweight if

$$q_i(s) < (1-\epsilon)\pi_i$$



- Algorithm:
  - Repeatedly increase scores of underweight nodes

Definitions

- Fix a set of scores s and steady state π
- Let $q_i(s)$ be the expected mass at v_i starting with π using s
- Call a node underweight if

$$q_i(s) < (1-\epsilon)\pi_i$$

Algorithm:

- Start with $s_i^0 = 1/n$
- For $t = 1, \ldots$
  - For each $v_i \in V$:
  - If v_i underweight:
    Set $s_i^t : q_i(s_{-i}^{t-1}, s_i^t) = (1-\epsilon/2)\pi_i$
  - else:
    Set $s_i^t = s_i^{t-1}$

Definitions

- Fix a set of scores s and steady state π
- Let $q_i(s)$ be the expected mass at v_i starting with π using s
- Call a node underweight if

$$q_i(s) < (1-\epsilon)\pi_i$$

Algorithm:

- Start with $s_i^0 = 1/n$
- For $t = 1, \ldots$
  - For each $v_i \in V$:
  - If v_i underweight:
    Set $s_i^t : q_i(s_{-i}^{t-1}, s_i^t) = (1-\epsilon/2)\pi_i$
  - else:
    Set $s_i^t = s_i^{t-1}$

Guaranteed to exist because f is monotone, continuous, unbounded & G is consistent

Definitions

- Fix a set of scores s and steady state π
- Let $q_i(s)$ be the expected mass at v_i starting with π using s
- Call a node underweight if

$$q_i(s) < (1-\epsilon)\pi_i$$

Note: scores never decrease

Algorithm:

- Start with $s_i^0 = 1/n$
- For $t = 1, \ldots$
  - For each $v_i \in V$:
  - If v_i underweight:
    Set $s_i^t : q_i(s_{-i}^{t-1}, s_i^t) = (1-\epsilon/2)\pi_i$
  - else:
    Set $s_i^t = s_i^{t-1}$

Guaranteed to exist because f is monotone, continuous, unbounded & G is consistent

Definitions

- Fix a set of scores s and steady state π
- Let $q_i(s)$ be the expected mass at v_i starting with π using s
- Call a node underweight if

$$q_i(s) < (1-\epsilon)\pi_i$$

Note: scores never decrease

If q is ever below π, it will always stay below

Algorithm:

- Start with $s_i^0 = 1/n$
- For $t = 1, \ldots$
  - For each $v_i \in V$:
  - If v_i underweight:
    Set $s_i^t : q_i(s_{-i}^{t-1}, s_i^t) = (1-\epsilon/2)\pi_i$
  - else:
    Set $s_i^t = s_i^{t-1}$

Guaranteed to exist because f is monotone, continuous, unbounded & G is consistent

Proof of Convergence

Key Lemma:

- There is an explicit bound M such that $s_i^t \leq M$ for all i, t .

Proof of Convergence

Key Lemma:

- There is an explicit bound M such that $s_i^t \leq M$ for all i, t .

Proof Sketch:

- Consider a set of scores that grows without bound
- These scores all must be underweight (these are the only scores that increase)
- Not all scores can be underweight (sum of underweight scores below 1)
- The scores growing without bound are taking all of the probability mass from those bounded
- By consistency, this demand must be met, a contradiction.

Proof of Convergence

Key Lemma:

- There is an explicit bound M such that $s_i^t \leq M$ for all i, t .

Finishing the Proof:

- Scores increase multiplicatively by factor of $(1 + \epsilon/2)$
- M is bounded by $\left(\frac{n^2 W}{\epsilon p_{\min}}\right)^n$
- Overall: $O\left(\frac{n^2}{\epsilon} \log \frac{nW}{\epsilon p_{\min}}\right)$ iterations suffice.

But Does it Work...

Experimental Evaluation:

- Dataset: empirical transitions
- Input: Transition graph and the steady state distribution
- Output: Transition probabilities
- Metrics: LogLikelihood or RMSE

Datasets

Wiki:

- Navigation paths through wikipedia.
- About 200k transition pairs, 51k user traces over 4.6k nodes

Rest:

- Results of broad restaurant queries to Google.
- 100k transitions, 65k nodes

Entree:

- Chicago restaurant recommendation system from 90s
- 50k transitions, 27k nodes

Comedy:

- Given a pair of videos, predict which one is judged funnier
- 225k transitions, 75k nodes

Baselines

Popularity:

- Transition proportionally to the steady state distribution (score = pi)

Uniform:

- Uniform over out-edges

Pagerank:

- Transition proportionally to the node pagerank

Temperature:

- MaxEnt regularization approach

Inversion:

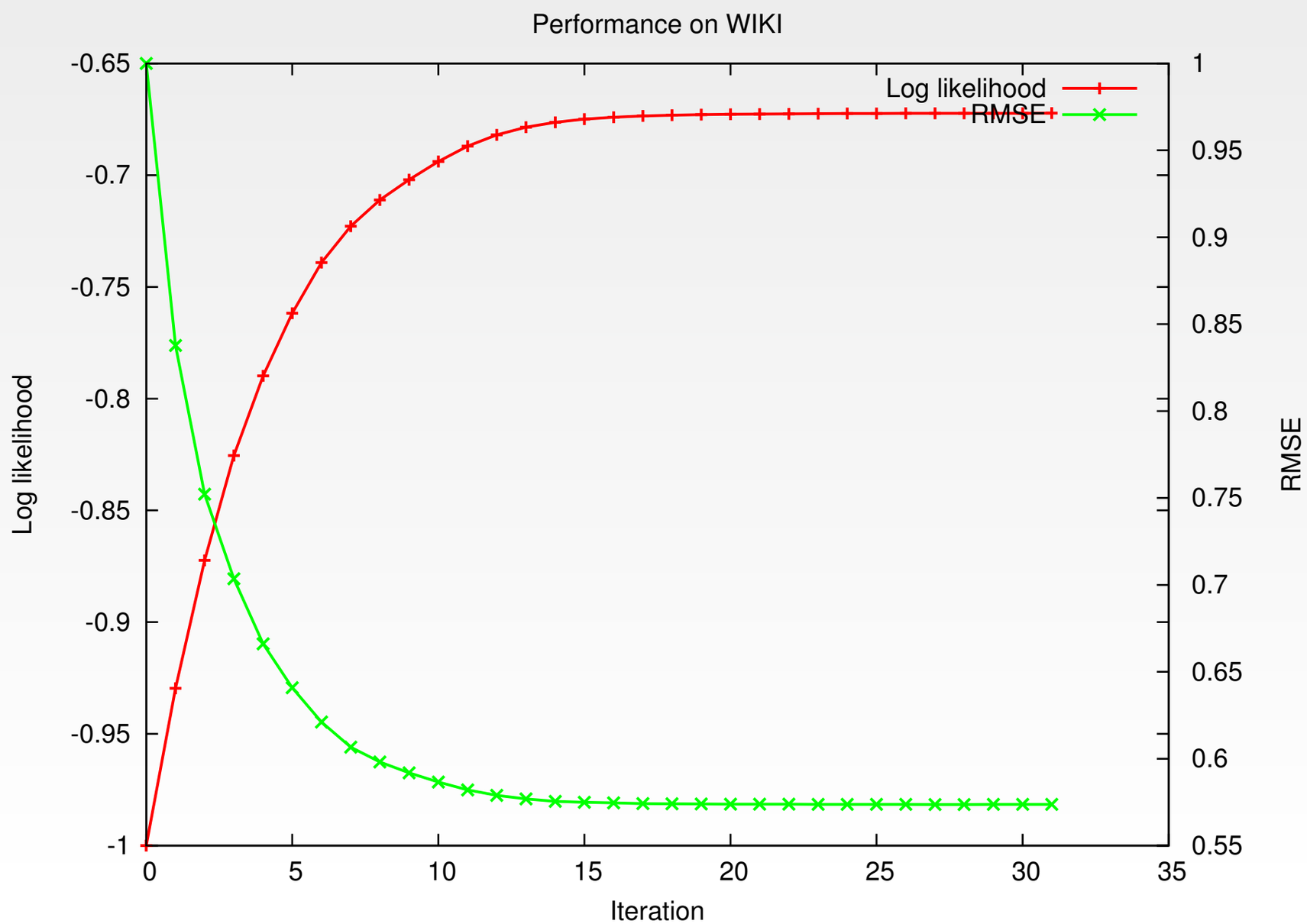
- Our algorithm

Results

RMSE Prediction:

	Popularity	Uniform	PageRank	Tempe-rature	Inversion
Wiki	1	0.65	0.83	0.65	**0.57**
Rest	1	1.17	1.39	1.21	**0.59**
Entree	1	0.69	1.01	0.56	**0.42**
Comedy	1	0.65	0.9	0.78	**0.36**

Convergence



The End