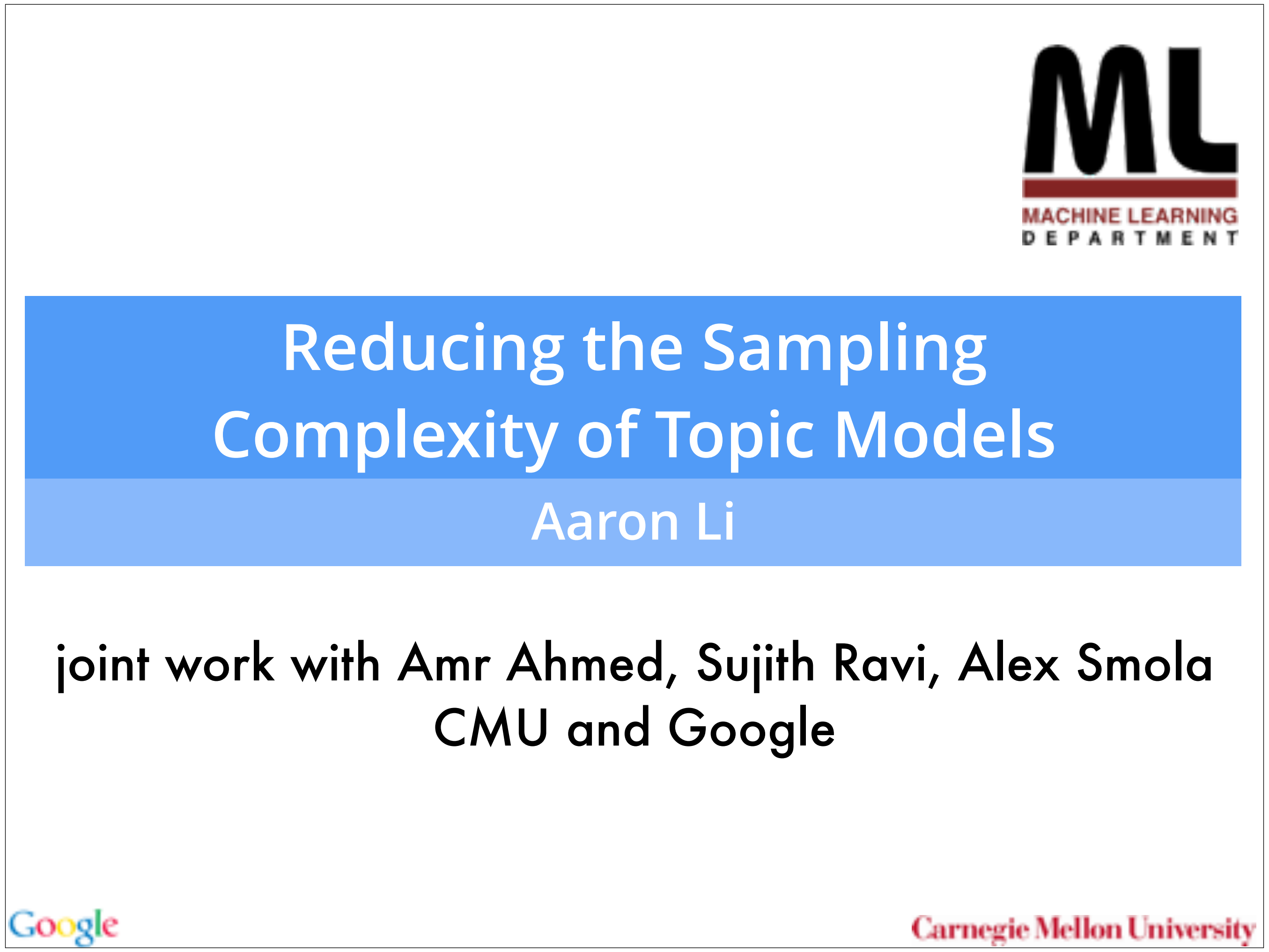


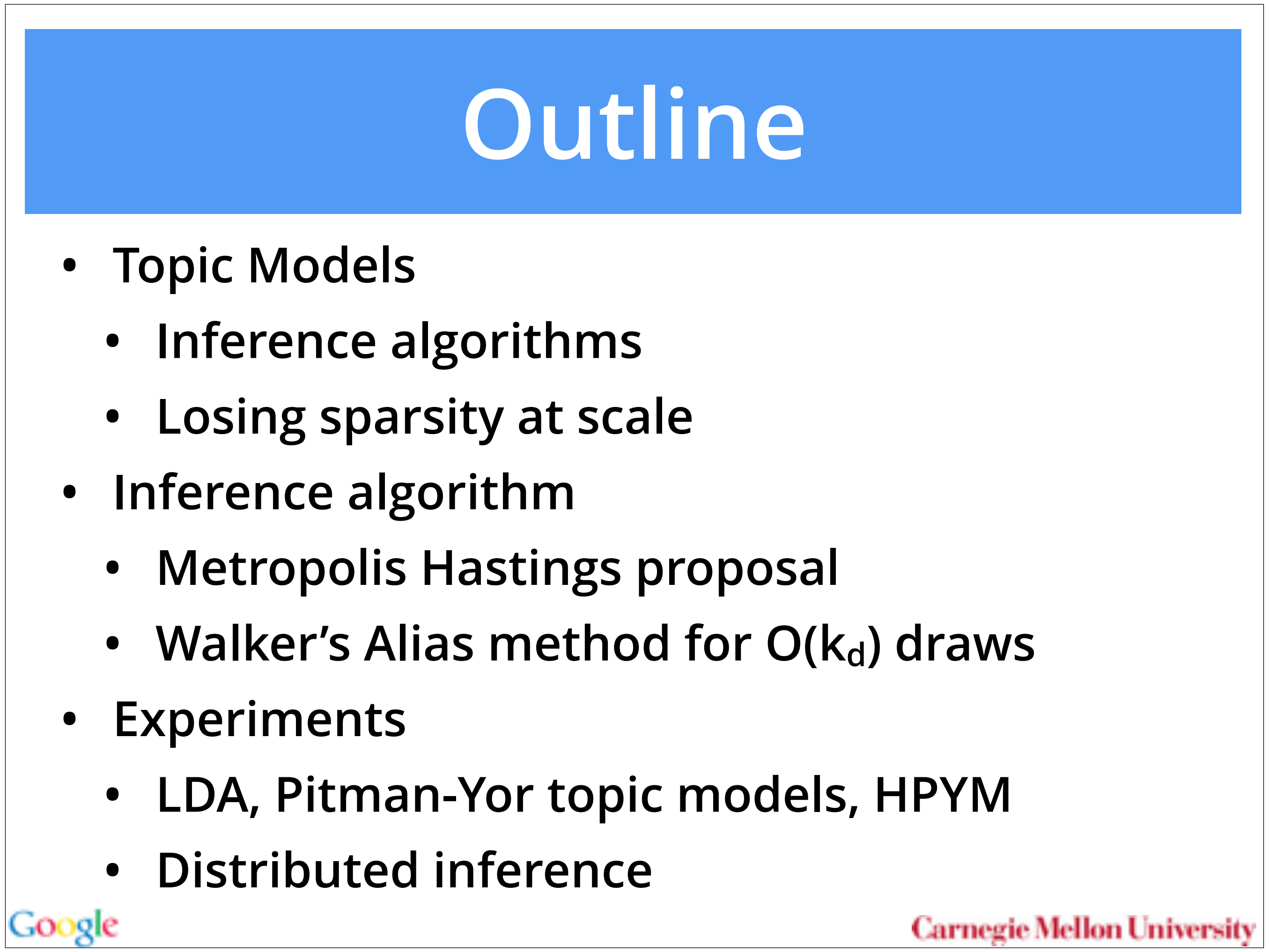
Reducing the Sampling Complexity of Topic Models

Aaron Li

joint work with Amr Ahmed, Sujith Ravi, Alex Smola
CMU and Google

Hello. I am Aaron Li from Carnegie Mellon. And it is my pleasure to present you the joint work with Amr, Sujith, and Alex.

Today I am going to talk about some fundamental techniques to make topic models run faster.



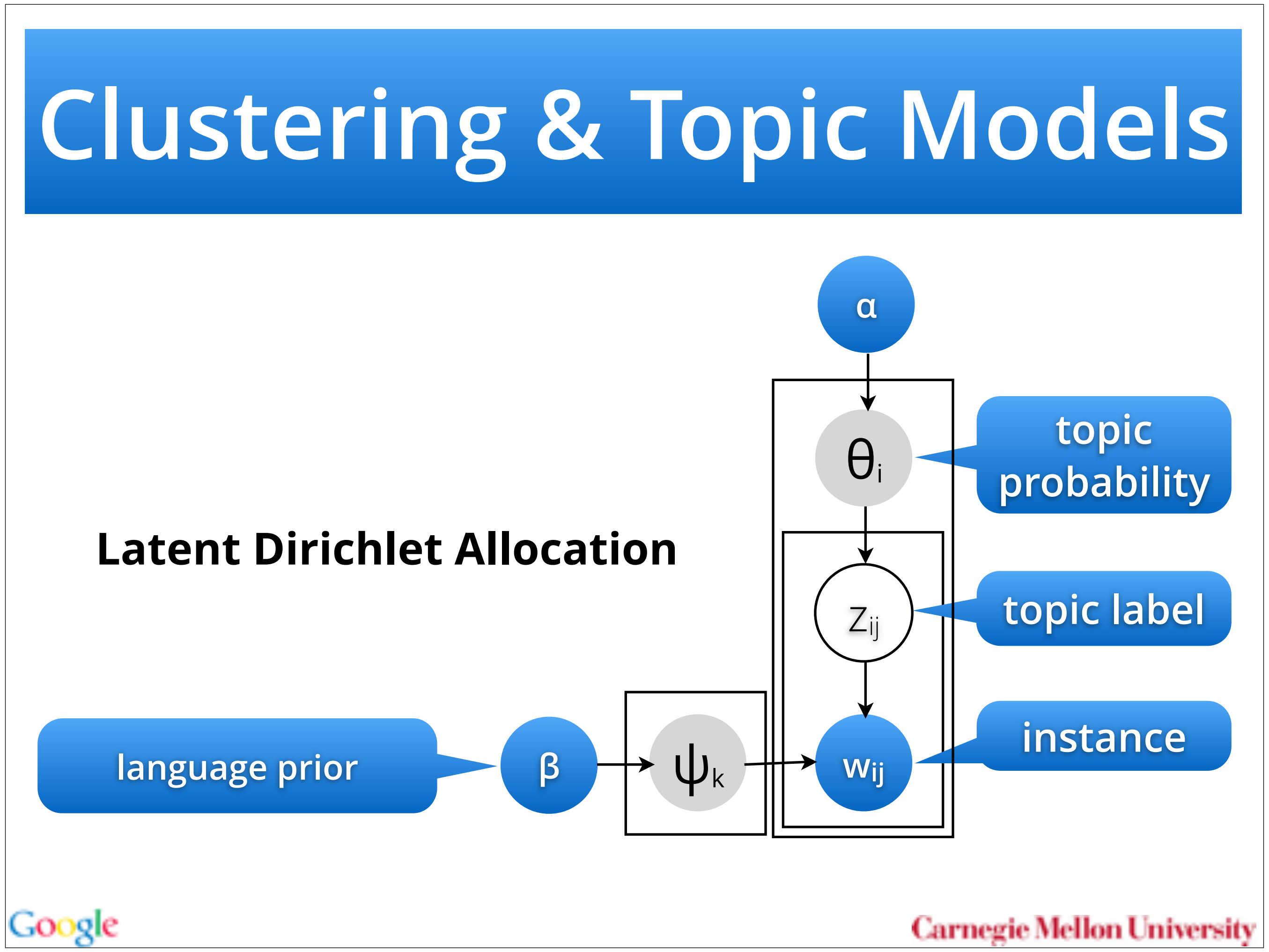
Outline

- Topic Models
  - Inference algorithms
  - Losing sparsity at scale
- Inference algorithm
  - Metropolis Hastings proposal
  - Walker's Alias method for O(k_d) draws
- Experiments
  - LDA, Pitman-Yor topic models, HPYM
  - Distributed inference

Obviously all of you sitting here are already experts of the subject, let me just quickly go through the basics, the current state-of-the-art, their shortcomings, then I will show your our alias method - it not only works for LDA, but also can be generalized, to work with more sophisticated models. For example, Pitman-Yor topic models, and Hierarchical Dirichlet Process.

Models

Okay, let’s get started...



Clustering & Topic Models

Latent Dirichlet Allocation

topic probability

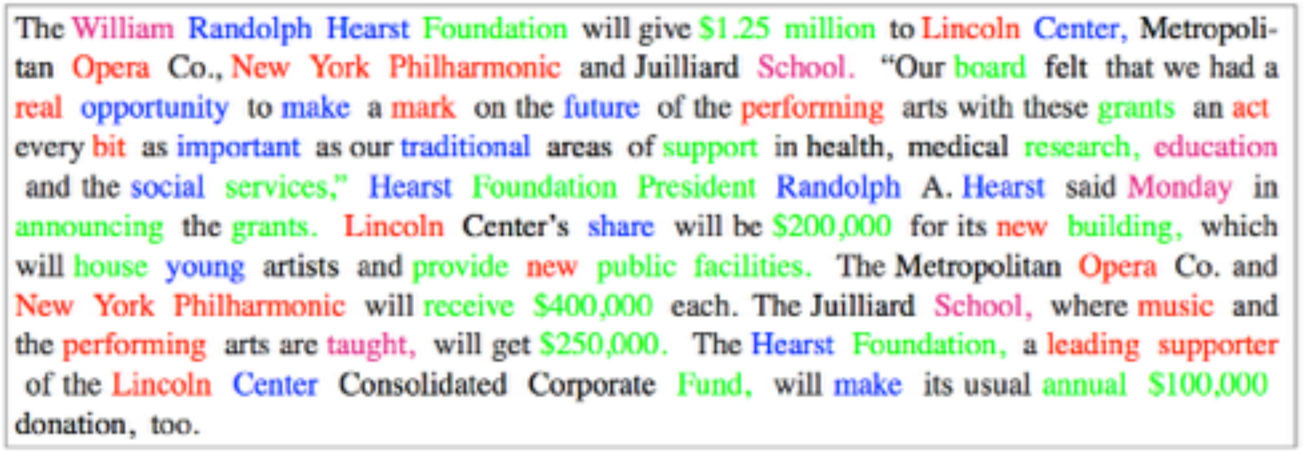
topic label

instance

language prior

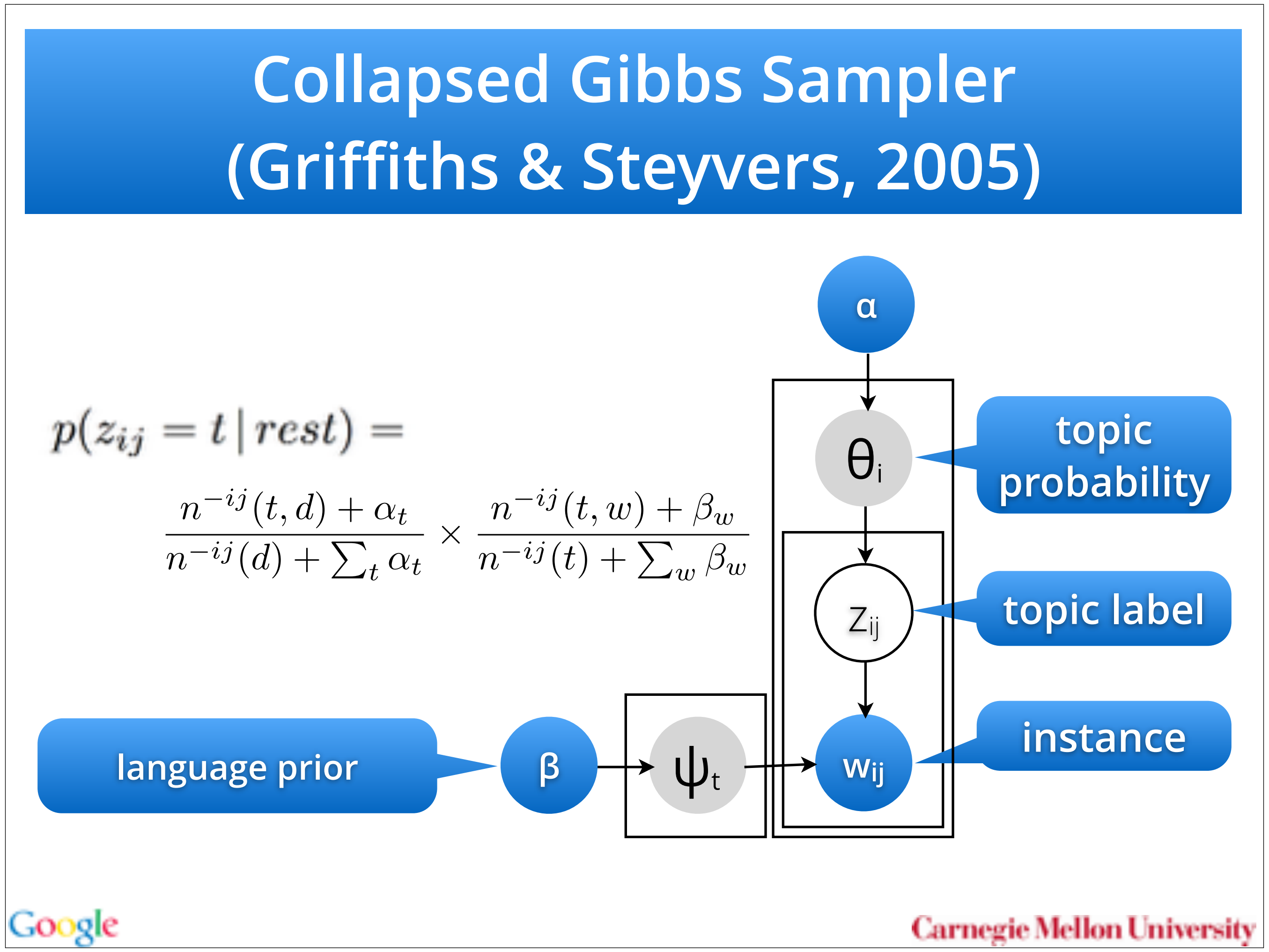
This is the good old LDA model that everyone loves.

Topics in text
(Blei, Ng, Jordan, 2003)



The William Randolph Hearst Foundation will give $1.25 million to Lincoln Center, Metropolitan Opera Co., New York Philharmonic and Juilliard School. "Our board felt that we had a real opportunity to make a mark on the future of the performing arts with these grants an act every bit as important as our traditional areas of support in health, medical research, education and the social services," Hearst Foundation President Randolph A. Hearst said Monday in announcing the grants. Lincoln Center's share will be $200,000 for its new building, which will house young artists and provide new public facilities. The Metropolitan Opera Co. and New York Philharmonic will receive $400,000 each. The Juilliard School, where music and the performing arts are taught, will get $250,000. The Hearst Foundation, a leading supporter of the Lincoln Center Consolidated Corporate Fund, will make its usual annual $100,000 donation, too.

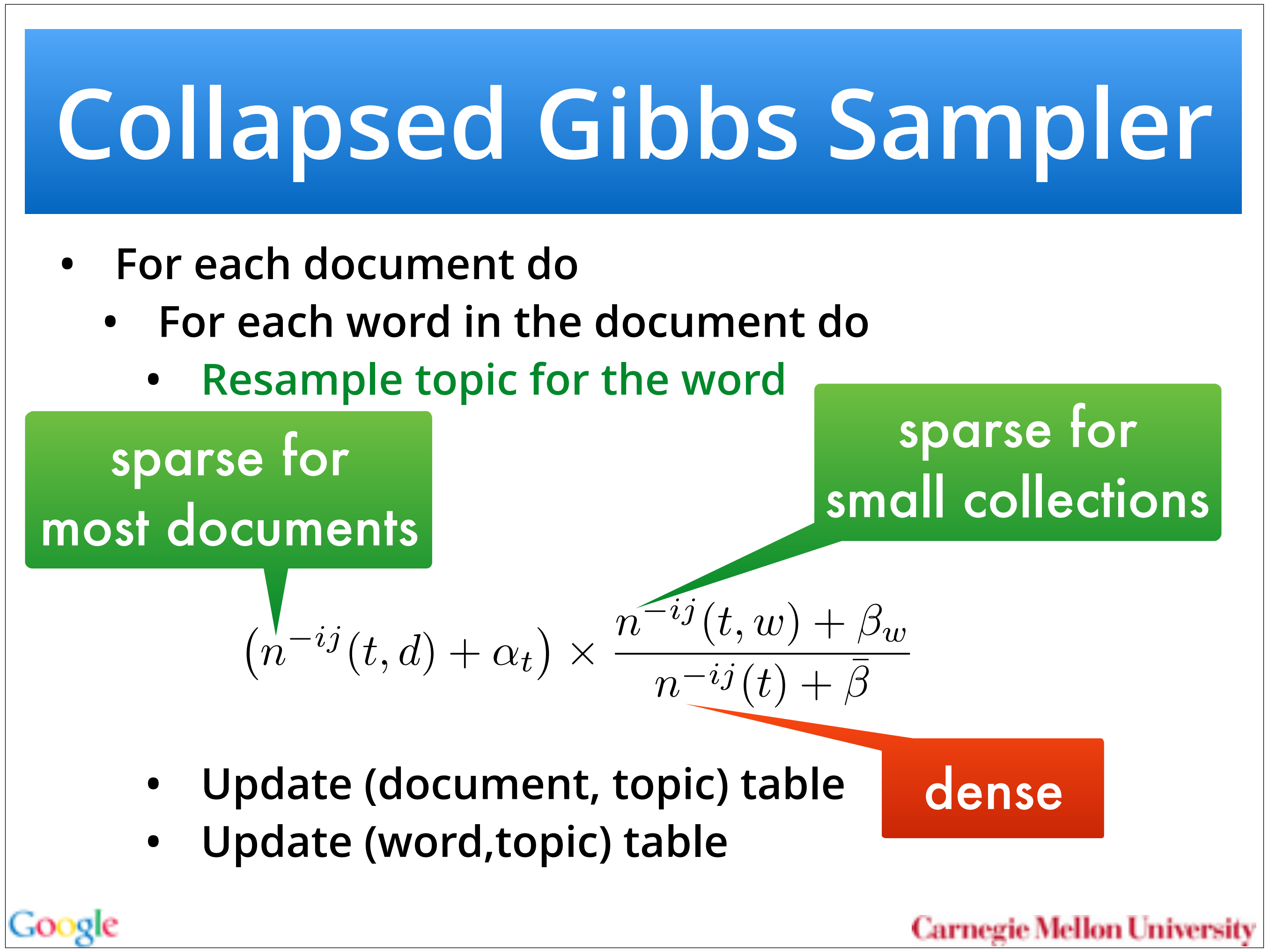
This is a standard application of LDA. It is used across all fields of data science to analyze billions of documents, images, videos, and user activities.



Collapsed Gibbs Sampler (Griffiths & Steyvers, 2005)

$$p(z_{ij} = t \,|\, rest) = \frac{n^{-ij}(t,d) + \alpha_t}{n^{-ij}(d) + \sum_t \alpha_t} \times \frac{n^{-ij}(t,w) + \beta_w}{n^{-ij}(t) + \sum_w \beta_w}$$

This is standard sampling equation for Collapsed Gibbs Sampler. Let me introduce the notations: t is topic, d is document, i is the document index, and j is the word index. Everything else is standard – n(t,d) is the document-topic count and n(t,w) is the topic-word count



Collapsed Gibbs Sampler

- For each document do
  - For each word in the document do
    - Resample topic for the word

sparse for most documents

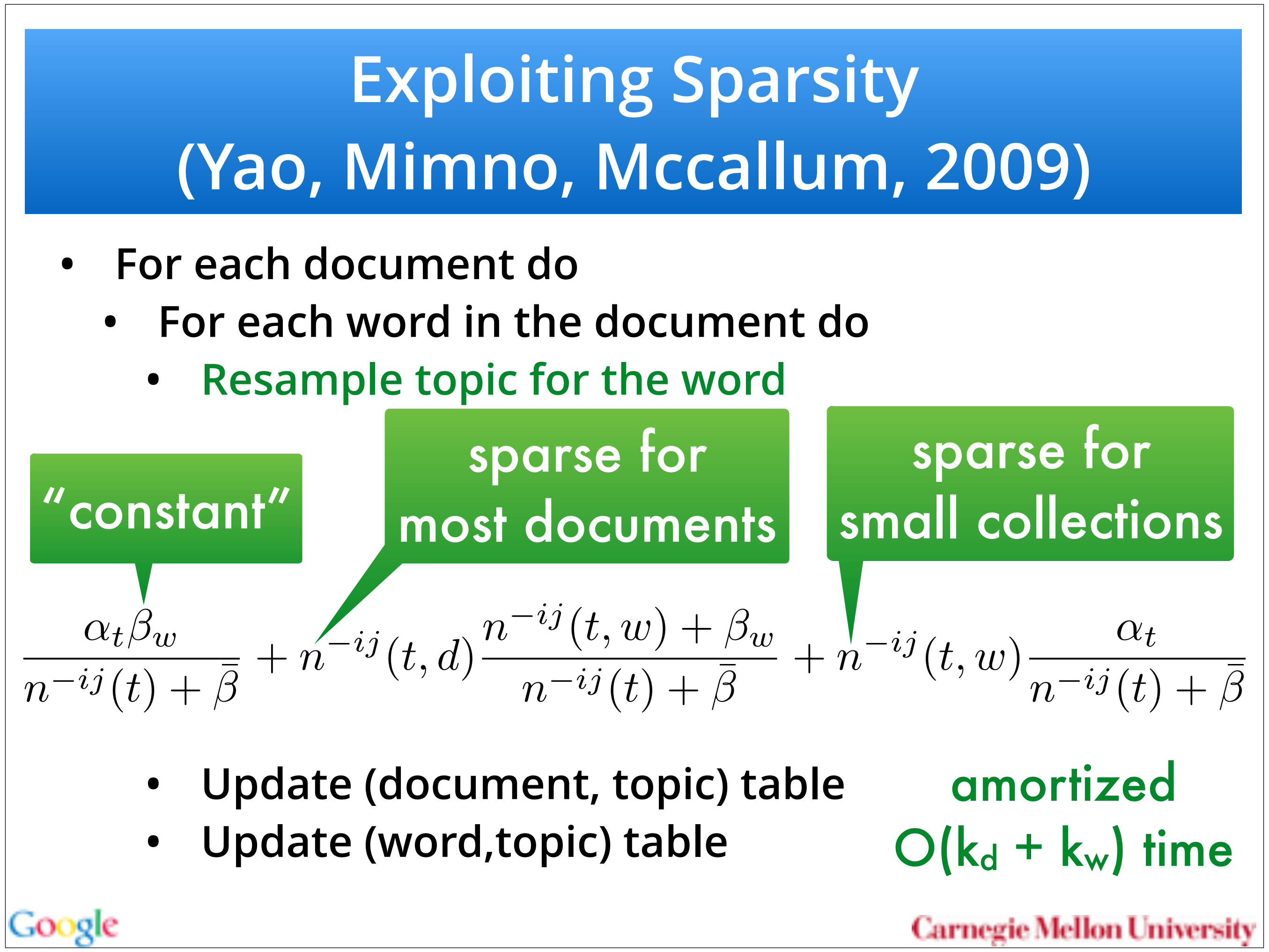
sparse for small collections

$$\left(n^{-ij}(t,d)+\alpha_t\right) \times \frac{n^{-ij}(t,w)+\beta_w}{n^{-ij}(t)+\bar{\beta}}$$

dense

    - Update (document, topic) table
    - Update (word,topic) table

The standard way to speed up the Collapsed Gibbs Sampler is to look at the sparsities of each term. For example the n(t,d) variables have only a few non-zero values for all t, because documents are short and each token in the document contributes no more than one topic. Similarly n(t,w) is also sparse if the average word frequency is low, which is generally true for small collections.



Exploiting Sparsity (Yao, Mimno, Mccallum, 2009)

- For each document do
  - For each word in the document do
    - Resample topic for the word

$$\overbrace{\frac{\alpha_t\beta_w}{n^{-ij}(t)+\bar{\beta}}}^{\text{“constant”}} + \overbrace{n^{-ij}(t,d)}^{\text{sparse for most documents}}\frac{n^{-ij}(t,w)+\beta_w}{n^{-ij}(t)+\bar{\beta}} + \overbrace{n^{-ij}(t,w)}^{\text{sparse for small collections}}\frac{\alpha_t}{n^{-ij}(t)+\bar{\beta}}$$

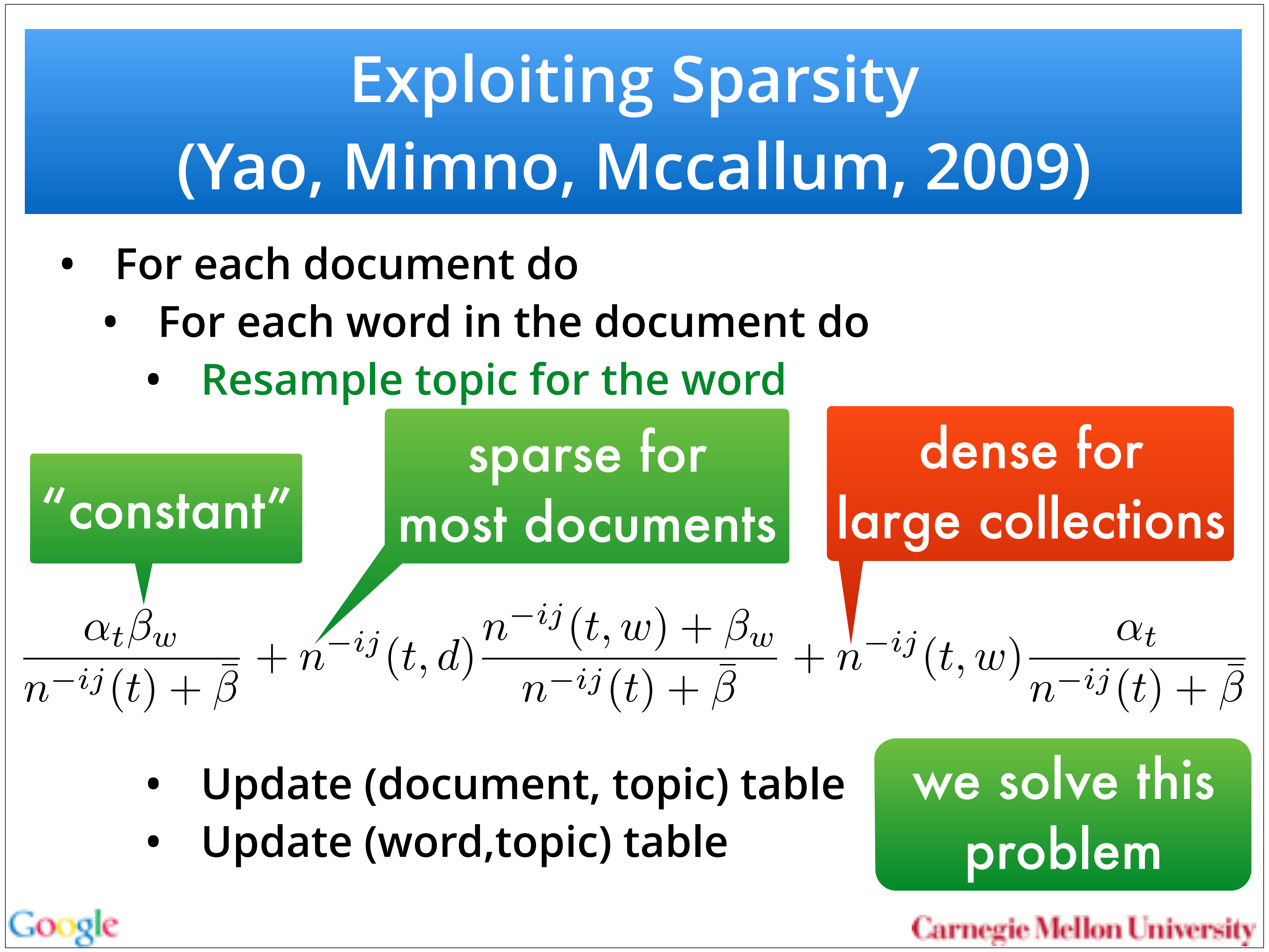
    - Update (document, topic) table
    - Update (word,topic) table

amortized $O(k_d + k_w)$ time

Google

Carnegie Mellon University

SparseLDA was created using this principle – the sparse terms are expanded as a multiplier for each term. It is very effective on small collections.



Exploiting Sparsity (Yao, Mimno, Mccallum, 2009)

- For each document do
  - For each word in the document do
    - Resample topic for the word

“constant”

sparse for most documents

dense for large collections

$$\frac{\alpha_t\beta_w}{n^{-ij}(t)+\bar{\beta}} + n^{-ij}(t,d)\frac{n^{-ij}(t,w)+\beta_w}{n^{-ij}(t)+\bar{\beta}} + n^{-ij}(t,w)\frac{\alpha_t}{n^{-ij}(t)+\bar{\beta}}$$

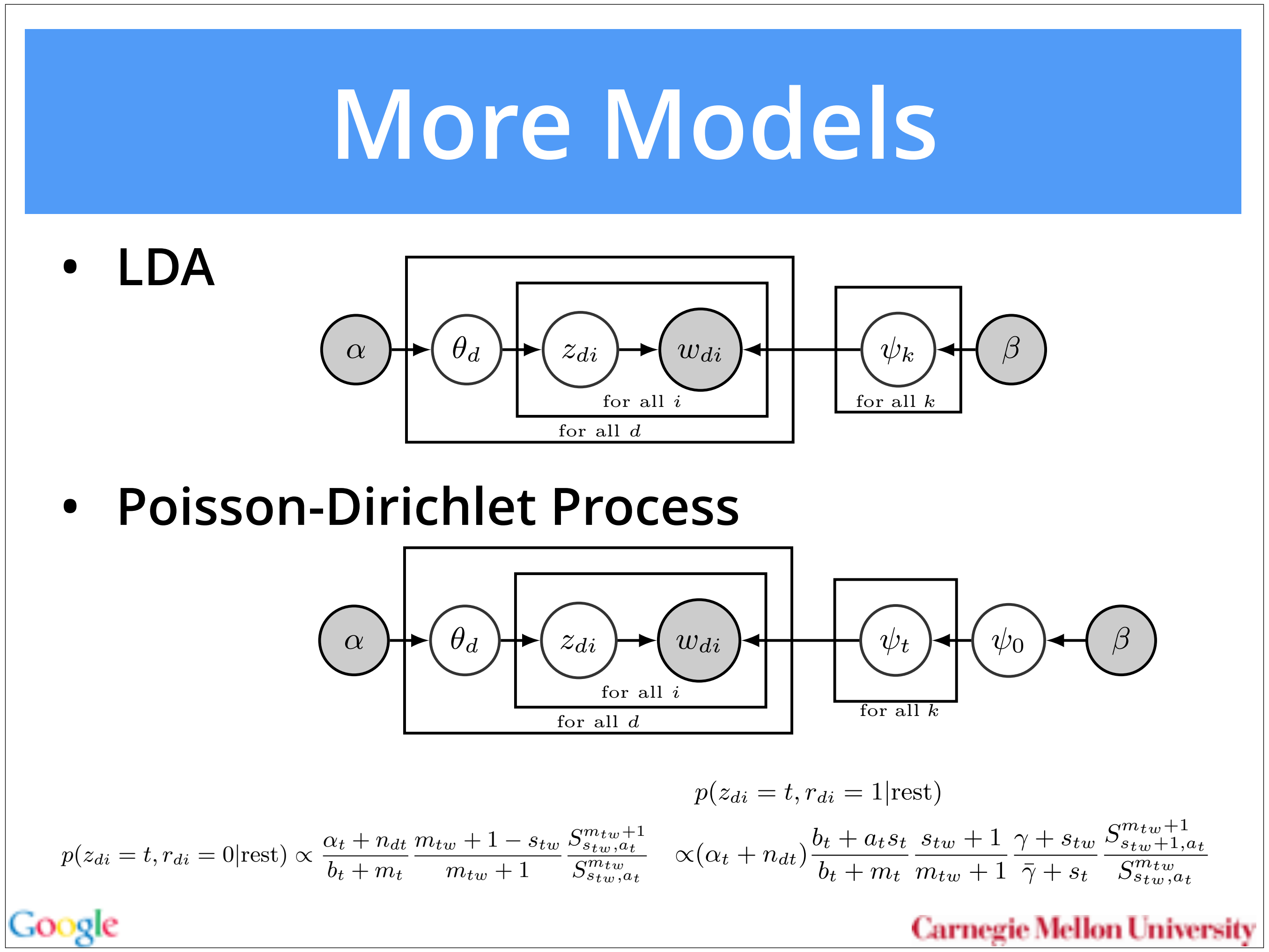
    - Update (document, topic) table
    - Update (word,topic) table

we solve this problem

Google

Carnegie Mellon University

But not on large collections. In large collections word frequency is a lot higher, and the n(t,w) variable becomes dense. The sampling performance falls back to the naive algorithm in the worse case.



More Models

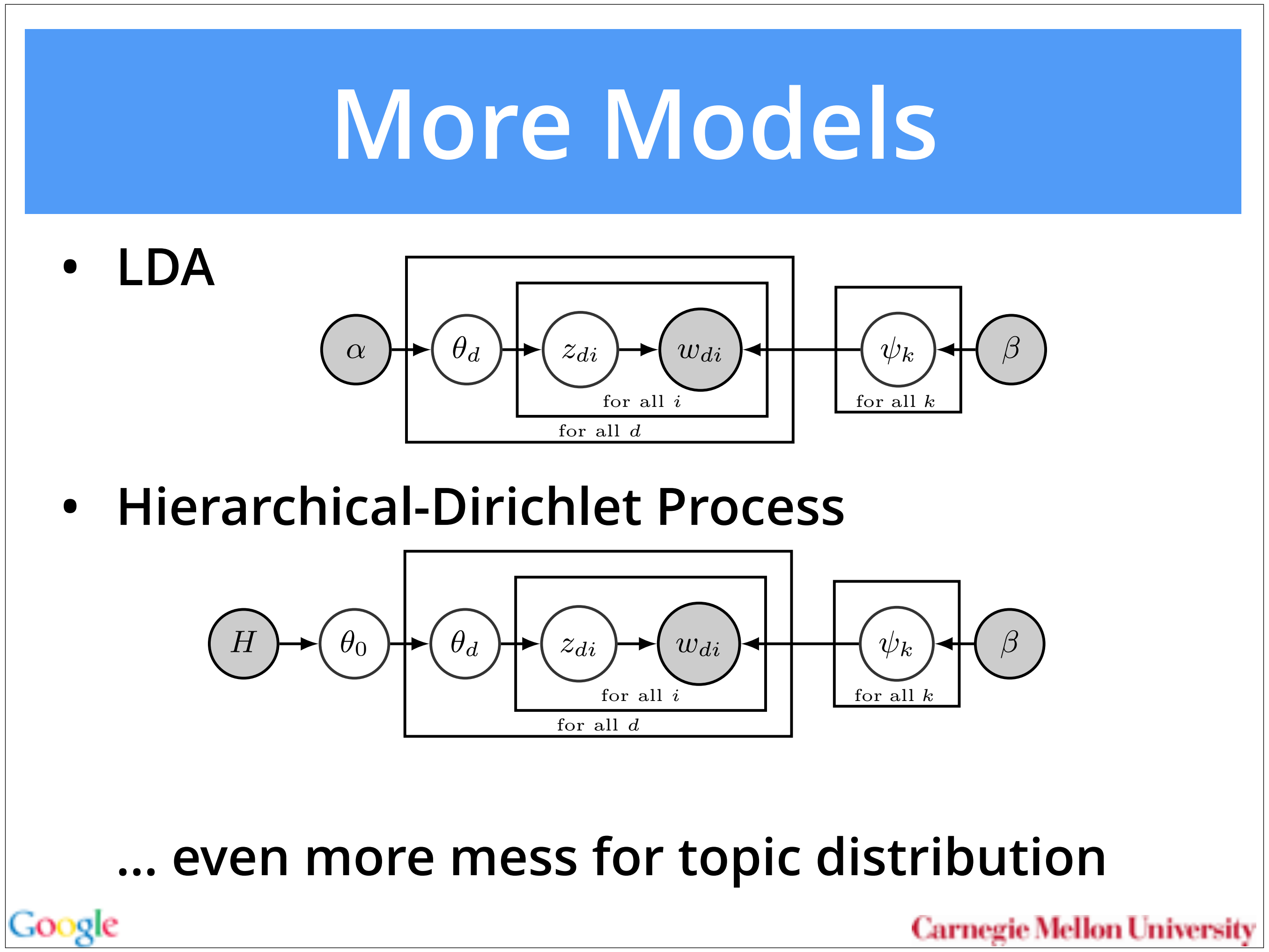
- LDA

- Poisson-Dirichlet Process

$$p(z_{di} = t, r_{di} = 0|\text{rest}) \propto \frac{\alpha_t + n_{dt}}{b_t + m_t} \frac{m_{tw} + 1 - s_{tw}}{m_{tw} + 1} \frac{S^{m_{tw}+1}_{s_{tw},a_t}}{S^{m_{tw}}_{s_{tw},a_t}}$$

$$p(z_{di} = t, r_{di} = 1|\text{rest}) \propto (\alpha_t + n_{dt}) \frac{b_t + a_t s_t}{b_t + m_t} \frac{s_{tw} + 1}{m_{tw} + 1} \frac{\gamma + s_{tw}}{\bar{\gamma} + s_t} \frac{S^{m_{tw}+1}_{s_{tw}+1,a_t}}{S^{m_{tw}}_{s_{tw},a_t}}$$

The same issues also appear in other topic models. Poisson–Dirichlet Process



More Models

- LDA

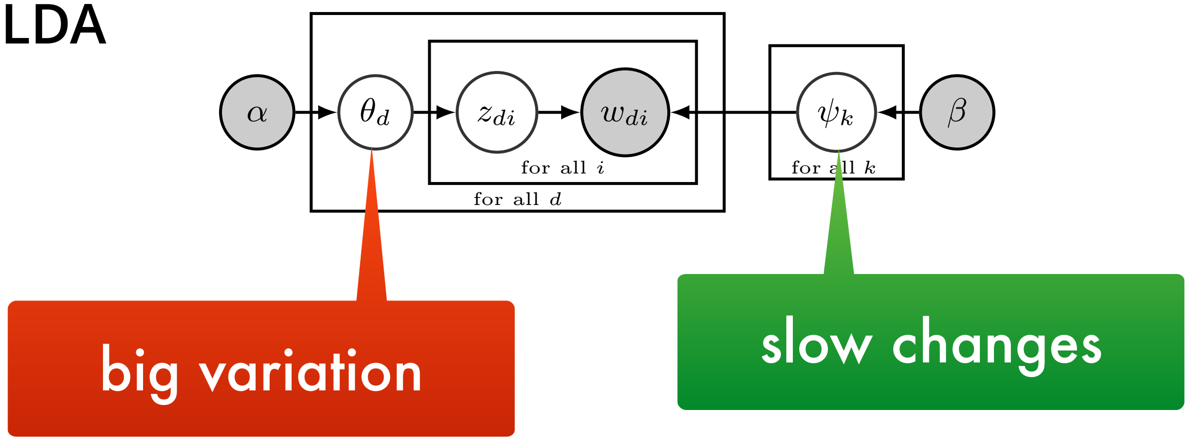
- Hierarchical-Dirichlet Process

… even more mess for topic distribution

and Hierarchical Dirichlet Process are two examples. Their sampling equations are very complex and they don’t even decompose into sparse terms even for small collections.

Key Idea of the Paper

- LDA

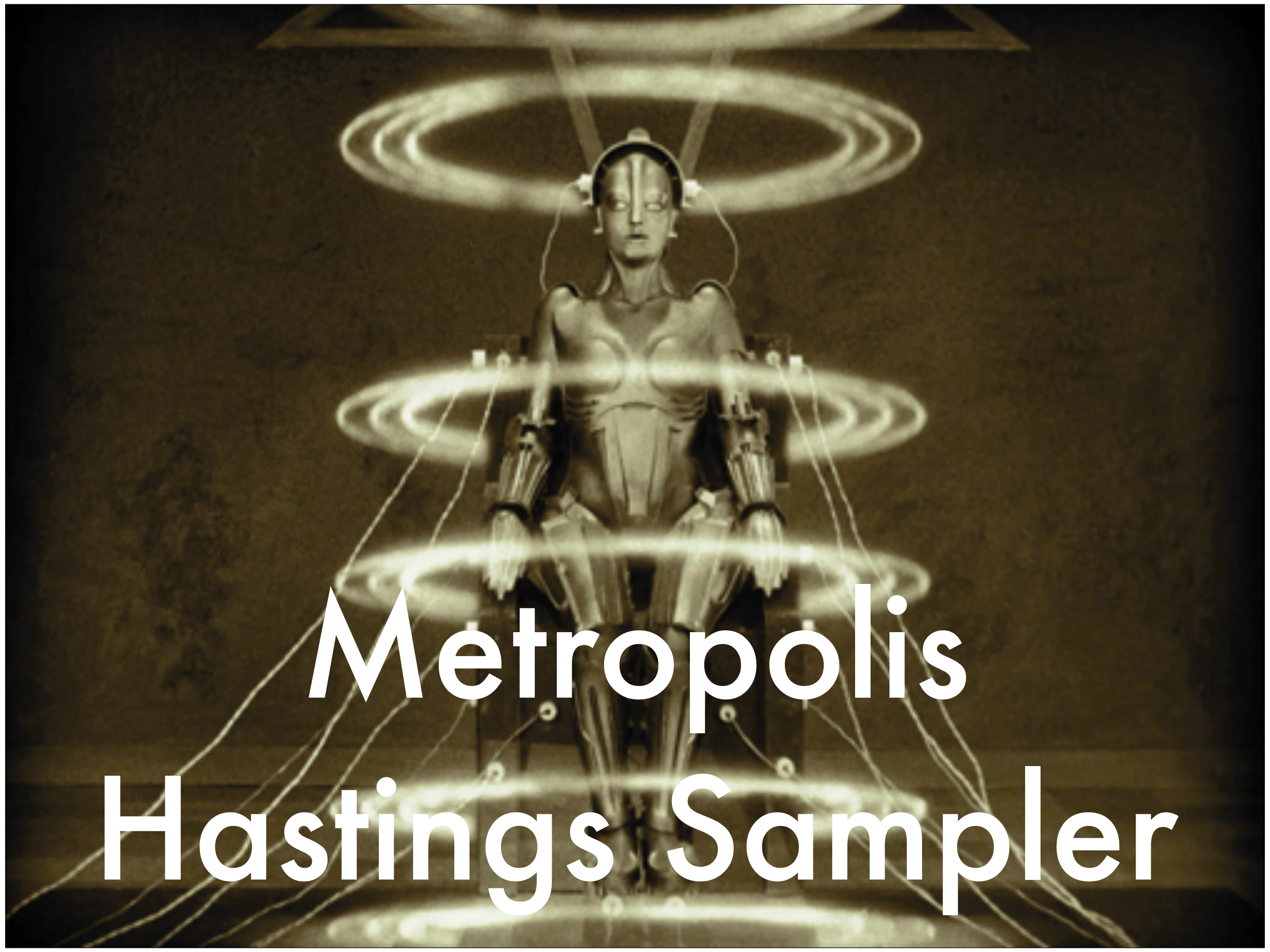


big variation

slow changes

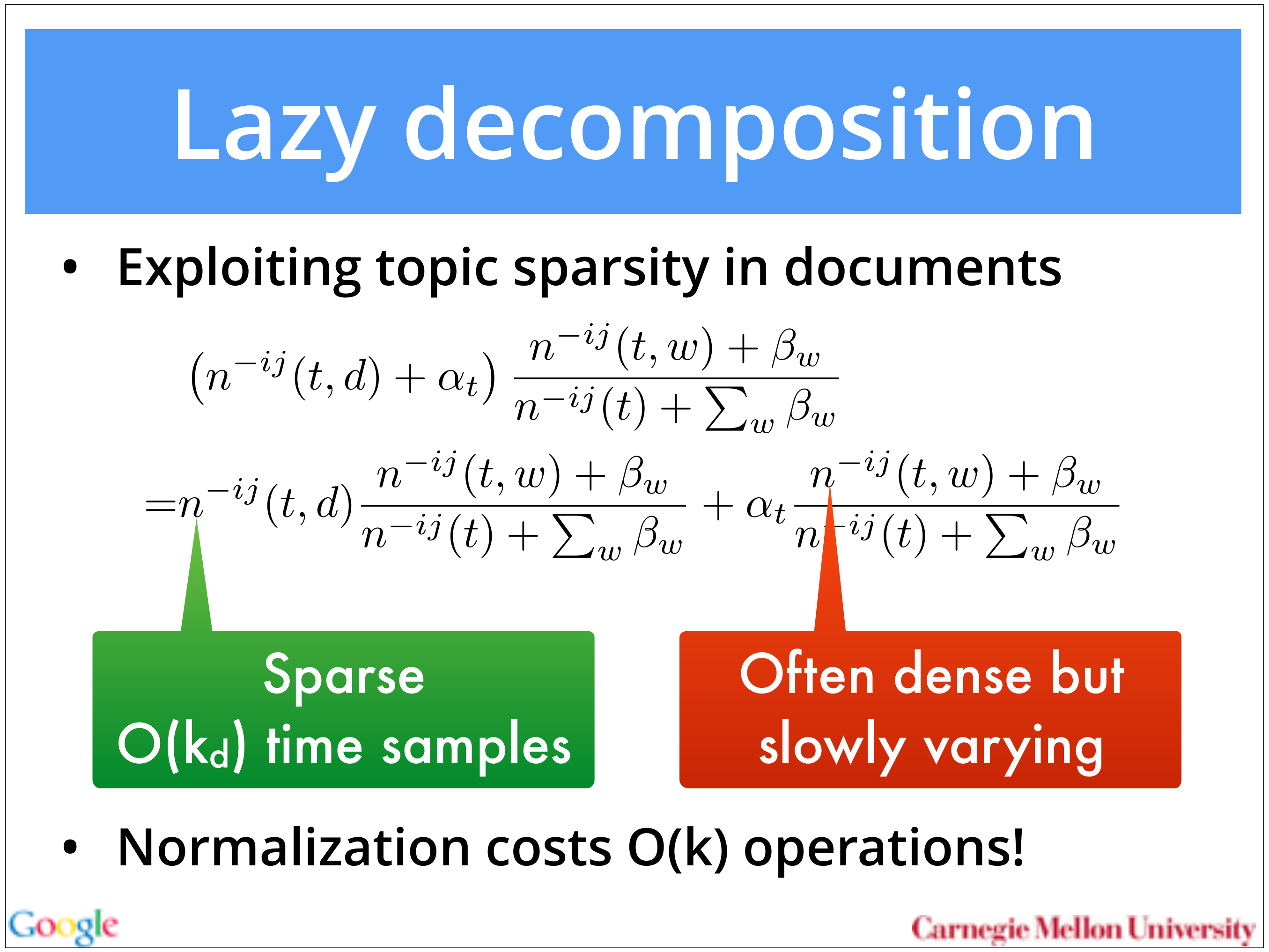
- Approximate slowly changing distribution by fixed distribution. Use Metropolis Hastings
- Amortized O(1) time proposals

In addition to sparse decomposition like in the state-of-the-art, we do something different. An often neglected property in topic models, is the word emission model is slow-changing, unlike the topic distribution for each document. I will use LDA as an example, to show you how we use this property to approximate the slow-changing part, reduce sampling complexity to only one sparse term, and fundamentally reduce the running time for all topic models.



Metropolis Hastings Sampler

The first ingredient is Metropolis Hasting Sampler.



Lazy decomposition

- Exploiting topic sparsity in documents

$$\left(n^{-ij}(t,d)+\alpha_t\right)\frac{n^{-ij}(t,w)+\beta_w}{n^{-ij}(t)+\sum_w \beta_w}$$

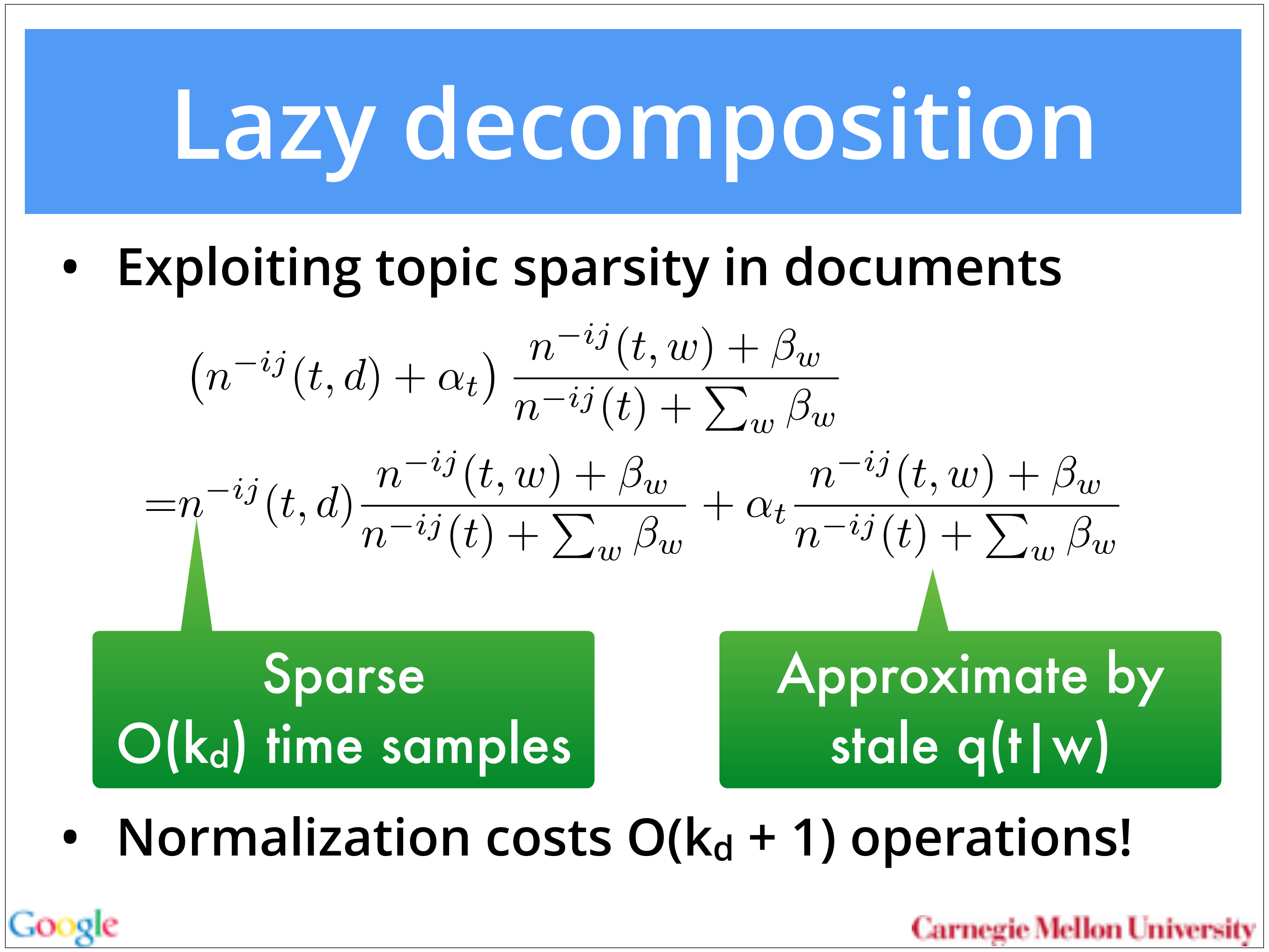
$$=n^{-ij}(t,d)\frac{n^{-ij}(t,w)+\beta_w}{n^{-ij}(t)+\sum_w \beta_w}+\alpha_t\frac{n^{-ij}(t,w)+\beta_w}{n^{-ij}(t)+\sum_w \beta_w}$$

Sparse
O(k_d) time samples

Often dense but
slowly varying

- Normalization costs O(k) operations!

First of all the dense part is exactly the slow changing part. As I discussed before the sparse part can be sample very quickly.



Lazy decomposition

- **Exploiting topic sparsity in documents**

$$\left(n^{-ij}(t,d)+\alpha_t\right)\frac{n^{-ij}(t,w)+\beta_w}{n^{-ij}(t)+\sum_w \beta_w}$$

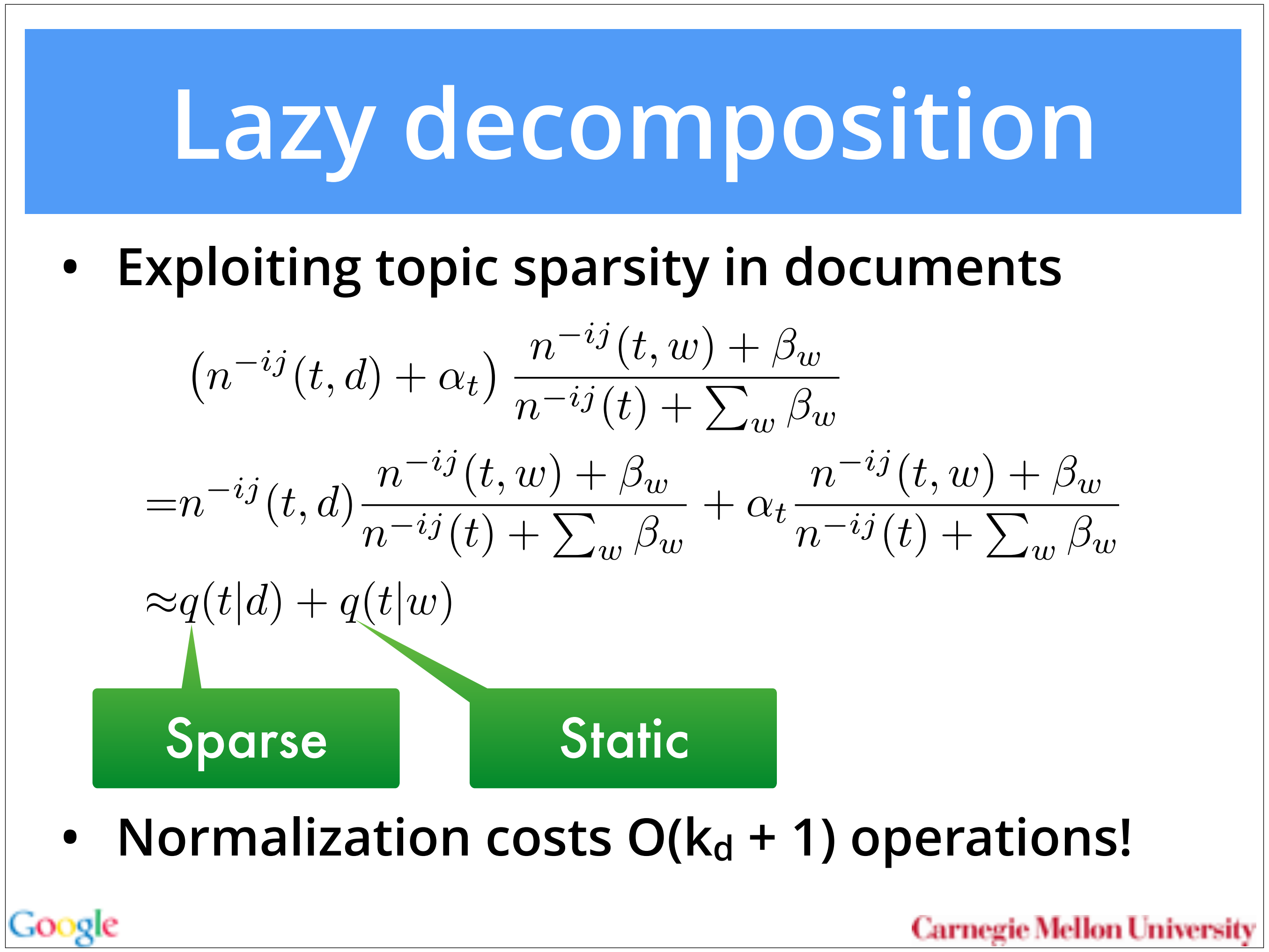
$$=n^{-ij}(t,d)\frac{n^{-ij}(t,w)+\beta_w}{n^{-ij}(t)+\sum_w \beta_w}+\alpha_t\frac{n^{-ij}(t,w)+\beta_w}{n^{-ij}(t)+\sum_w \beta_w}$$

Sparse
O(k_d) time samples

Approximate by
stale q(t|w)

- **Normalization costs O(k_d + 1) operations!**

Instead of sampling from the dense part directly, and recompute the probabilities of each outcome every time, we can draw sample from an approximate static distribution. A static distribution has a constant normalizer, therefore at first we reduced the sampling cost from O(k) down to O(k_d), where k_d is the number of non-zero topic counts in current document.



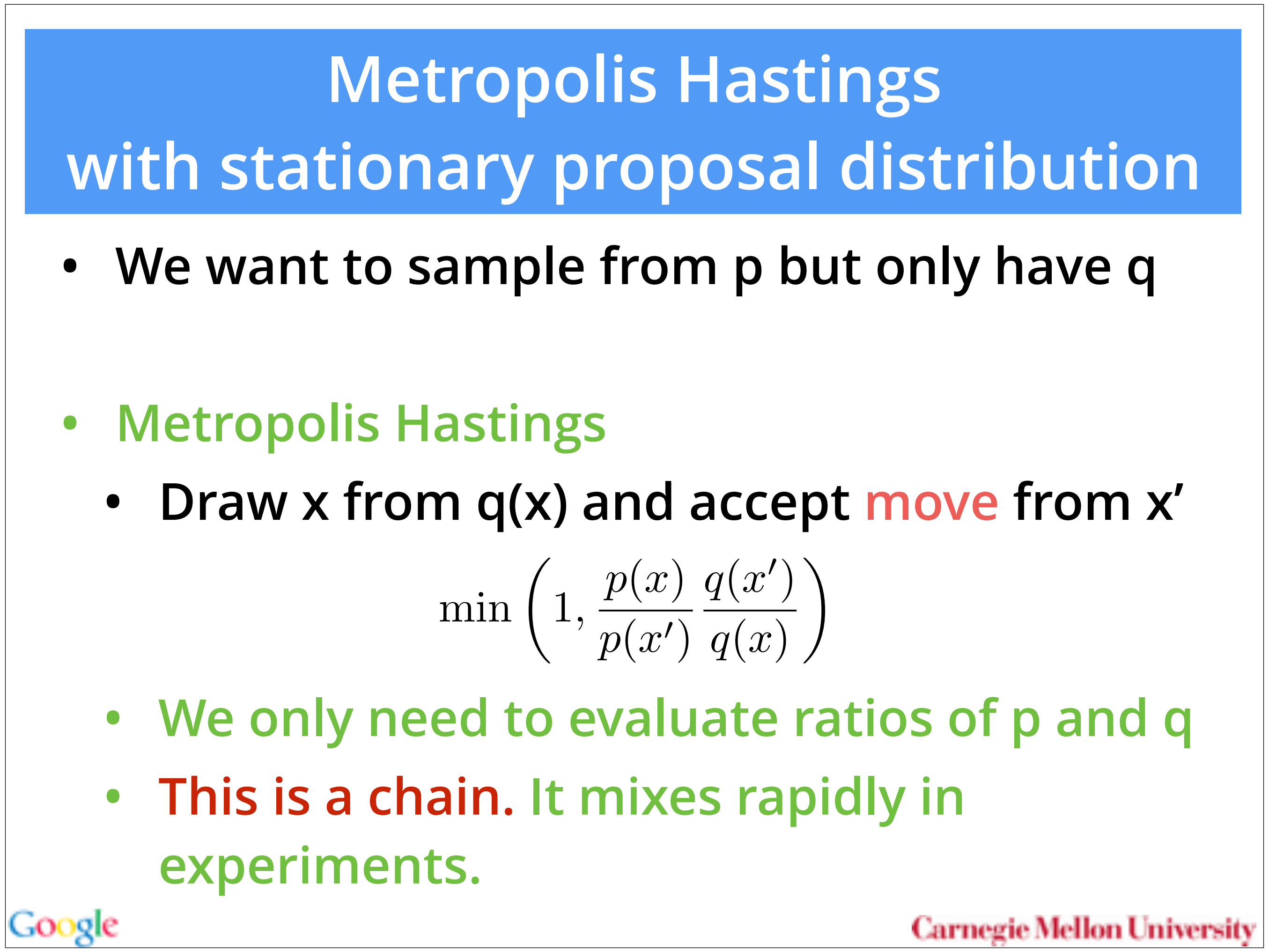
Lazy decomposition

- Exploiting topic sparsity in documents

$$
\begin{aligned}
&\left(n^{-ij}(t,d) + \alpha_t\right) \frac{n^{-ij}(t,w) + \beta_w}{n^{-ij}(t) + \sum_w \beta_w} \\
=& n^{-ij}(t,d) \frac{n^{-ij}(t,w) + \beta_w}{n^{-ij}(t) + \sum_w \beta_w} + \alpha_t \frac{n^{-ij}(t,w) + \beta_w}{n^{-ij}(t) + \sum_w \beta_w} \\
\approx& \underbrace{q(t|d)}_{\text{Sparse}} + \underbrace{q(t|w)}_{\text{Static}}
\end{aligned}
$$

- Normalization costs O(k_d + 1) operations!

To make things clear we can rewrite the equation in a simple way. A sparse term depending on the document and current word, and a dense term depending only on the current word.



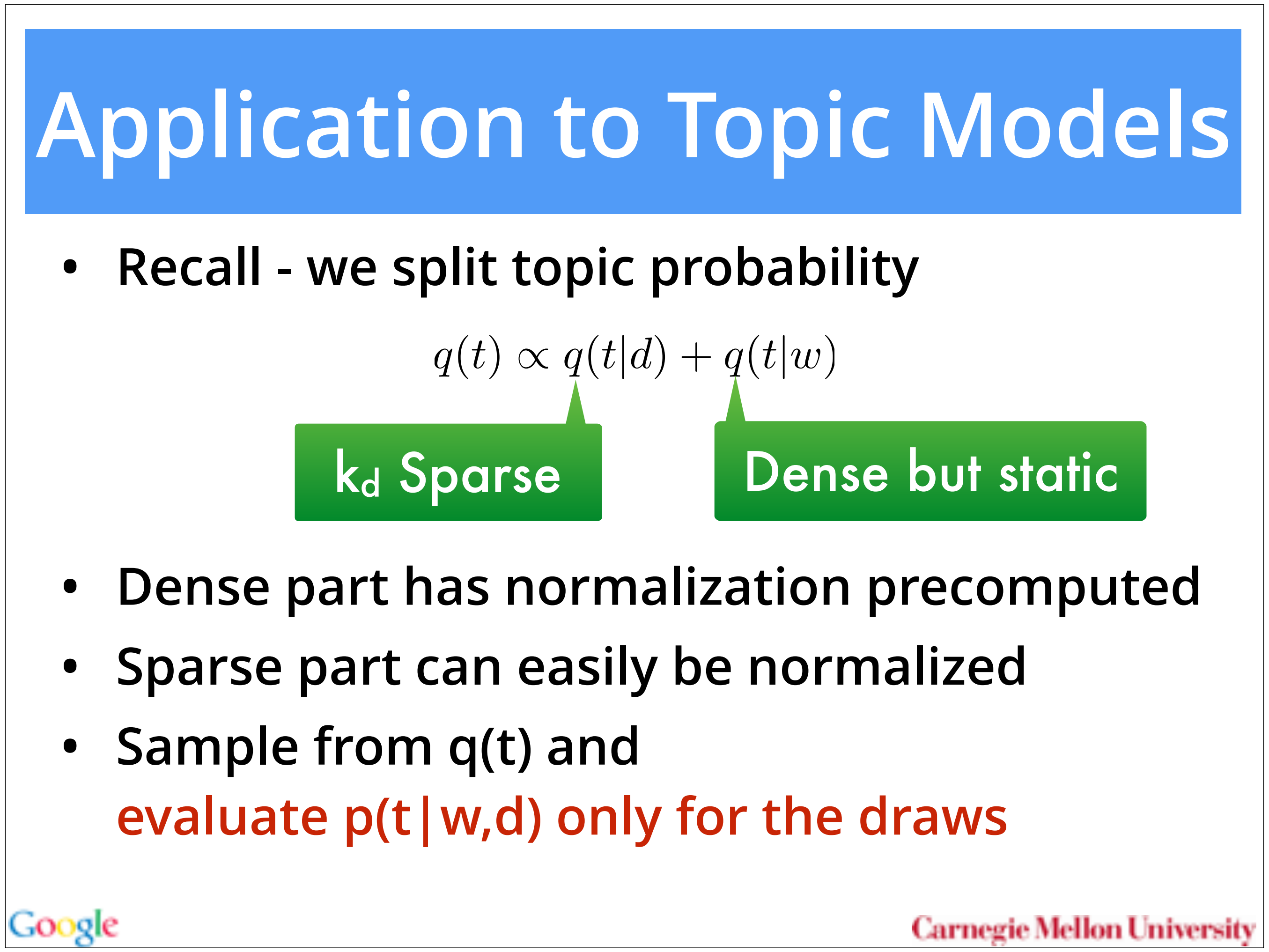
Metropolis Hastings with stationary proposal distribution

- We want to sample from p but only have q
- Metropolis Hastings
  - Draw x from q(x) and accept move from x’

$$\min\left(1, \frac{p(x)}{p(x')}\frac{q(x')}{q(x)}\right)$$

  - We only need to evaluate ratios of p and q
  - This is a chain. It mixes rapidly in experiments.

Having only the approximate distribution, q, at our disposal, with Metropolis Hastings, we are still able sample from the true distribution, p. Here is how it works: assume our old sample is x’, we draw a sample x from q(x). Then we compute the acceptance probability of x’->x with the equation here. This can be done very quickly in constant time, since we only need to evaluate two ratios.



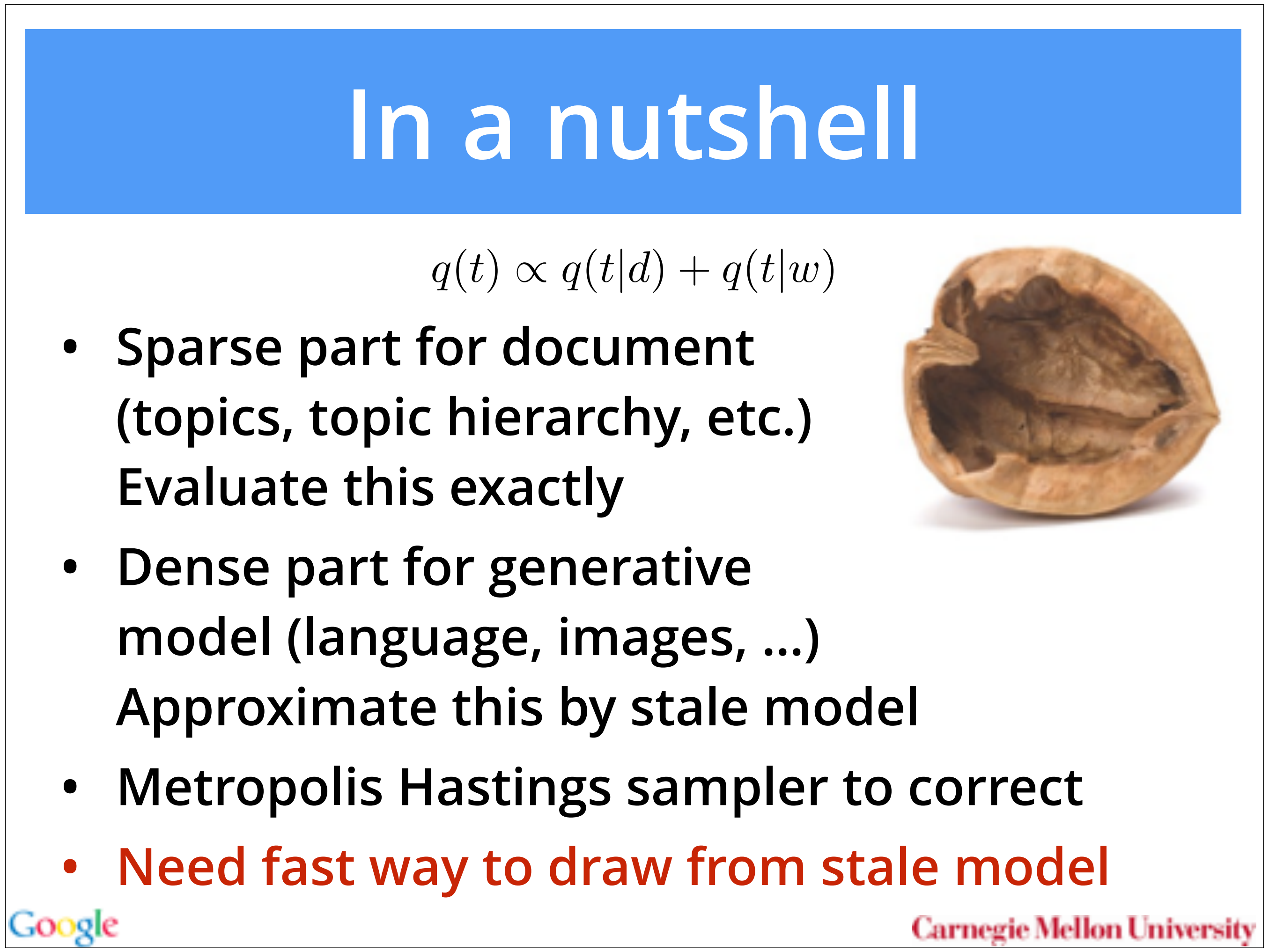
Application to Topic Models

- Recall - we split topic probability

$$q(t) \propto \underbrace{q(t|d)}_{k_d\ \text{Sparse}} + \underbrace{q(t|w)}_{\text{Dense but static}}$$

- Dense part has normalization precomputed
- Sparse part can easily be normalized
- Sample from q(t) and evaluate p(t|w,d) only for the draws

Therefore, if there is a way to sample quickly from the dense part, we would be able to reduce the sampling complexity to O(k_d), regardless of the size of the corpus.



In a nutshell

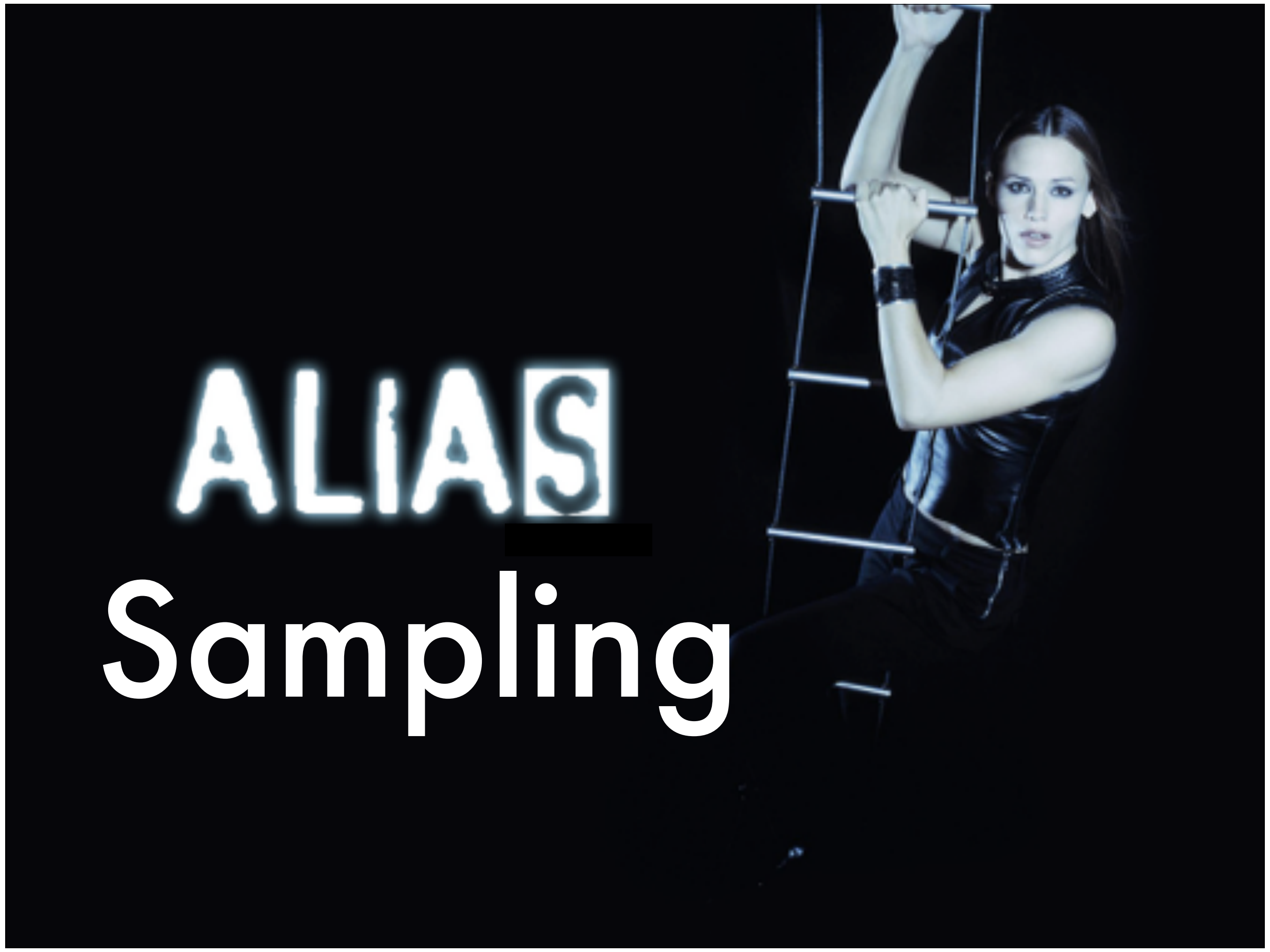
$$q(t) \propto q(t|d) + q(t|w)$$

- Sparse part for document (topics, topic hierarchy, etc.) Evaluate this exactly
- Dense part for generative model (language, images, …) Approximate this by stale model
- Metropolis Hastings sampler to correct
- Need fast way to draw from stale model

Google

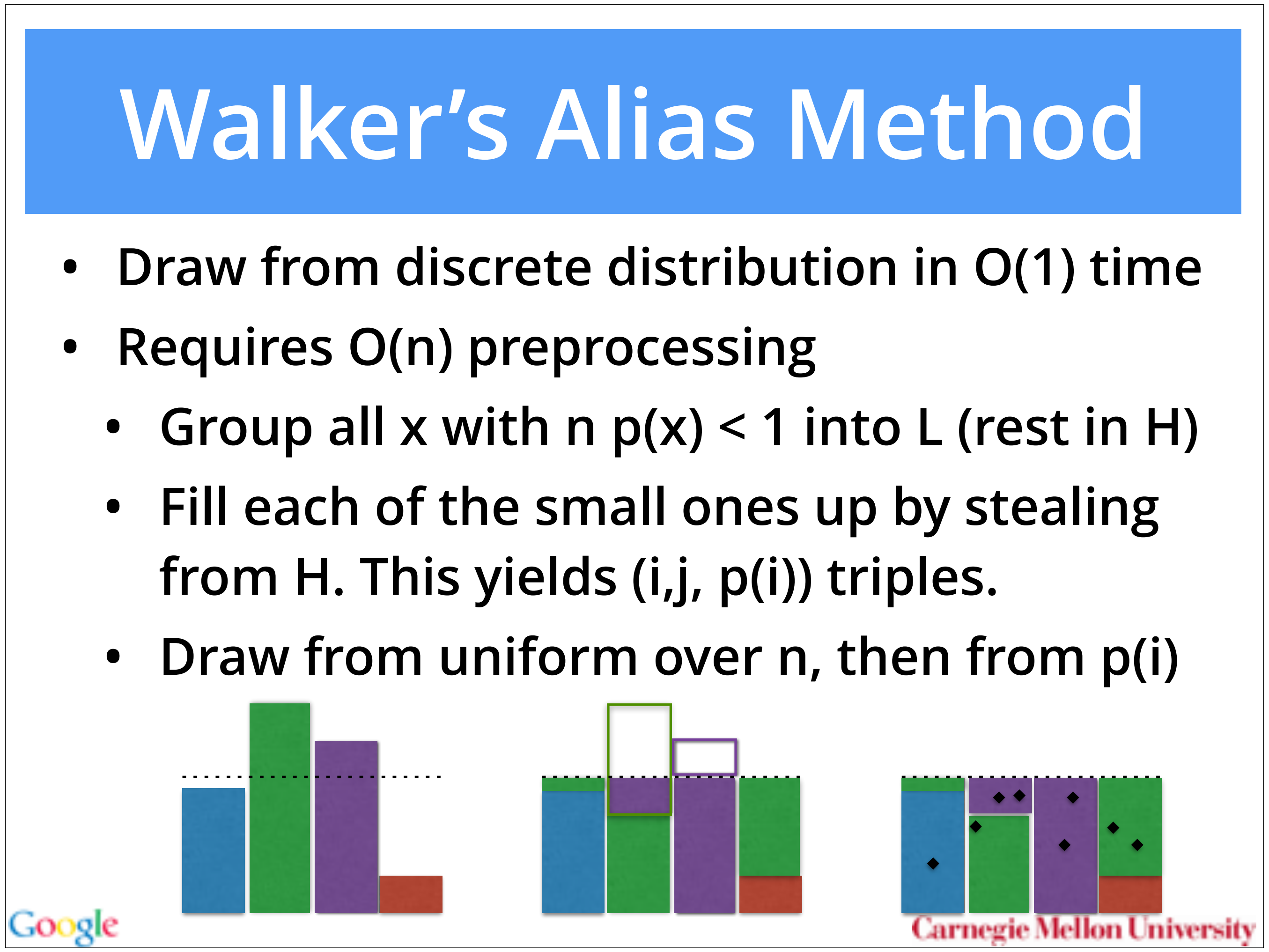
Carnegie Mellon University

In addition that, under the generalized form our method would work on many topic models, simply by rewriting the model as a summation of these two terms.



Sampling

To quickly sample from a static distribution we need the second ingredient – Alias sampling.



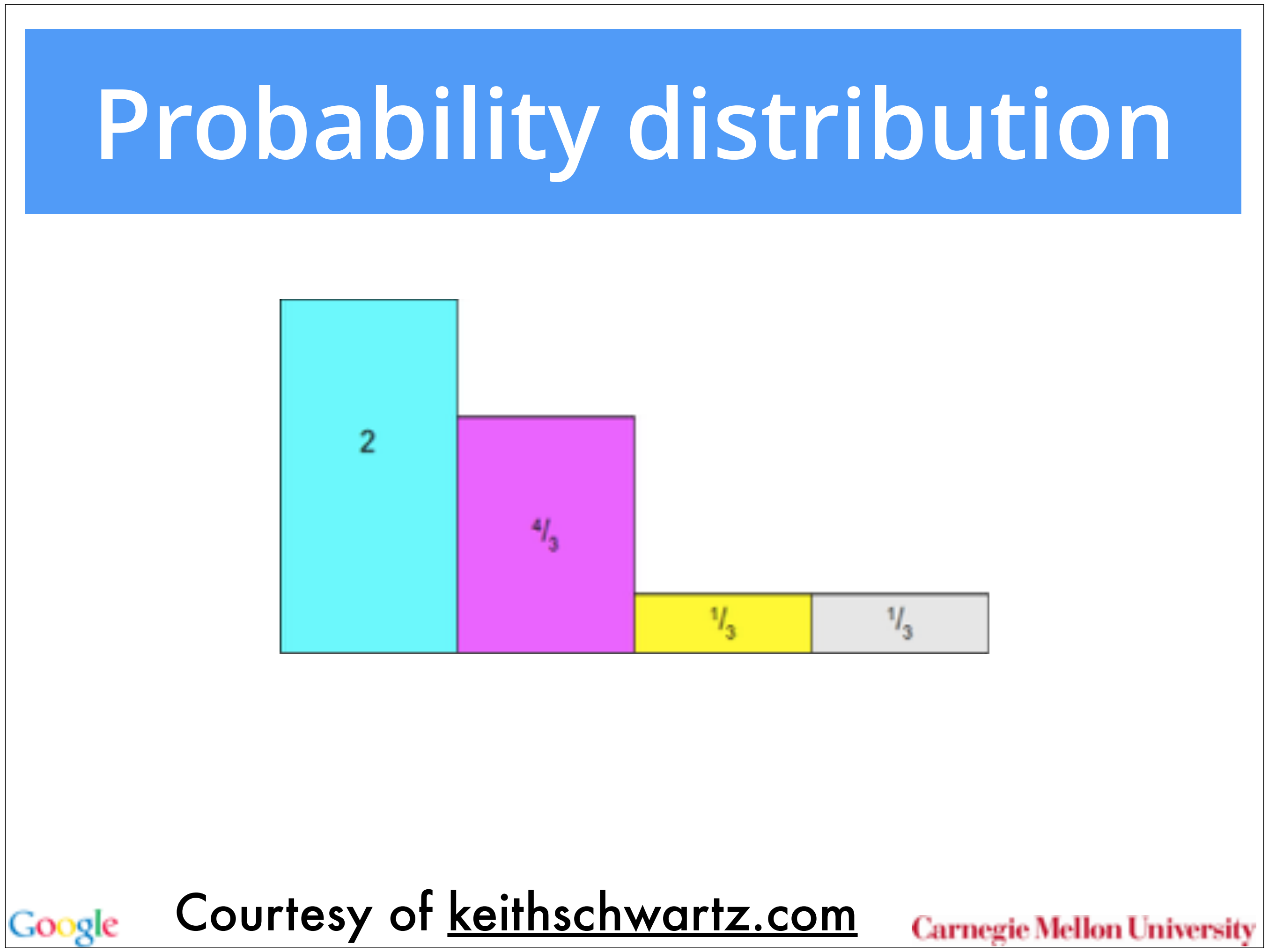
Walker’s Alias Method

- Draw from discrete distribution in O(1) time
- Requires O(n) preprocessing
  - Group all x with n p(x) < 1 into L (rest in H)
  - Fill each of the small ones up by stealing from H. This yields (i,j, p(i)) triples.
  - Draw from uniform over n, then from p(i)

Google

Carnegie Mellon University

Walker’s alias method is developed by Alastair Walker in 1977. Given a discrete distribution, it compiles the distribution into a static table. Afterwards drawing sample from compiled static distribution only takes constant time instead of a time linear to the number of outcomes. Let me briefly go through the algorithm.

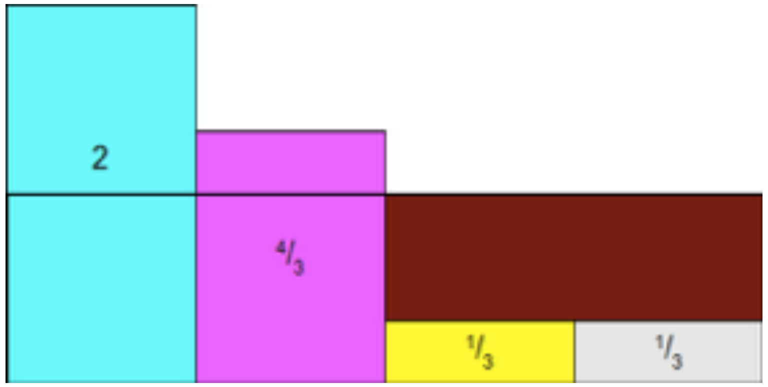


Probability distribution

Courtesy of keithschwartz.com

At first we have a distribution where each outcome has different proportions.

Probability distribution

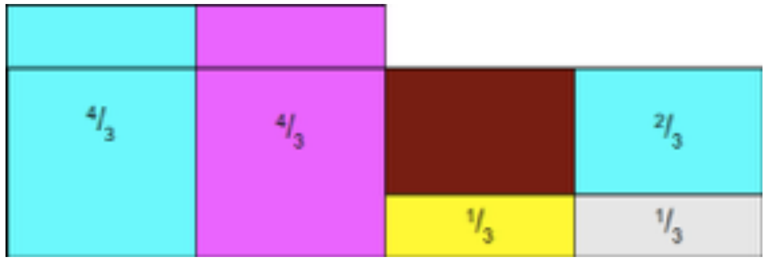


Splitting

Courtesy of keithschwartz.com

We go through all the outcomes and find the average value of the proportions.

Probability distribution

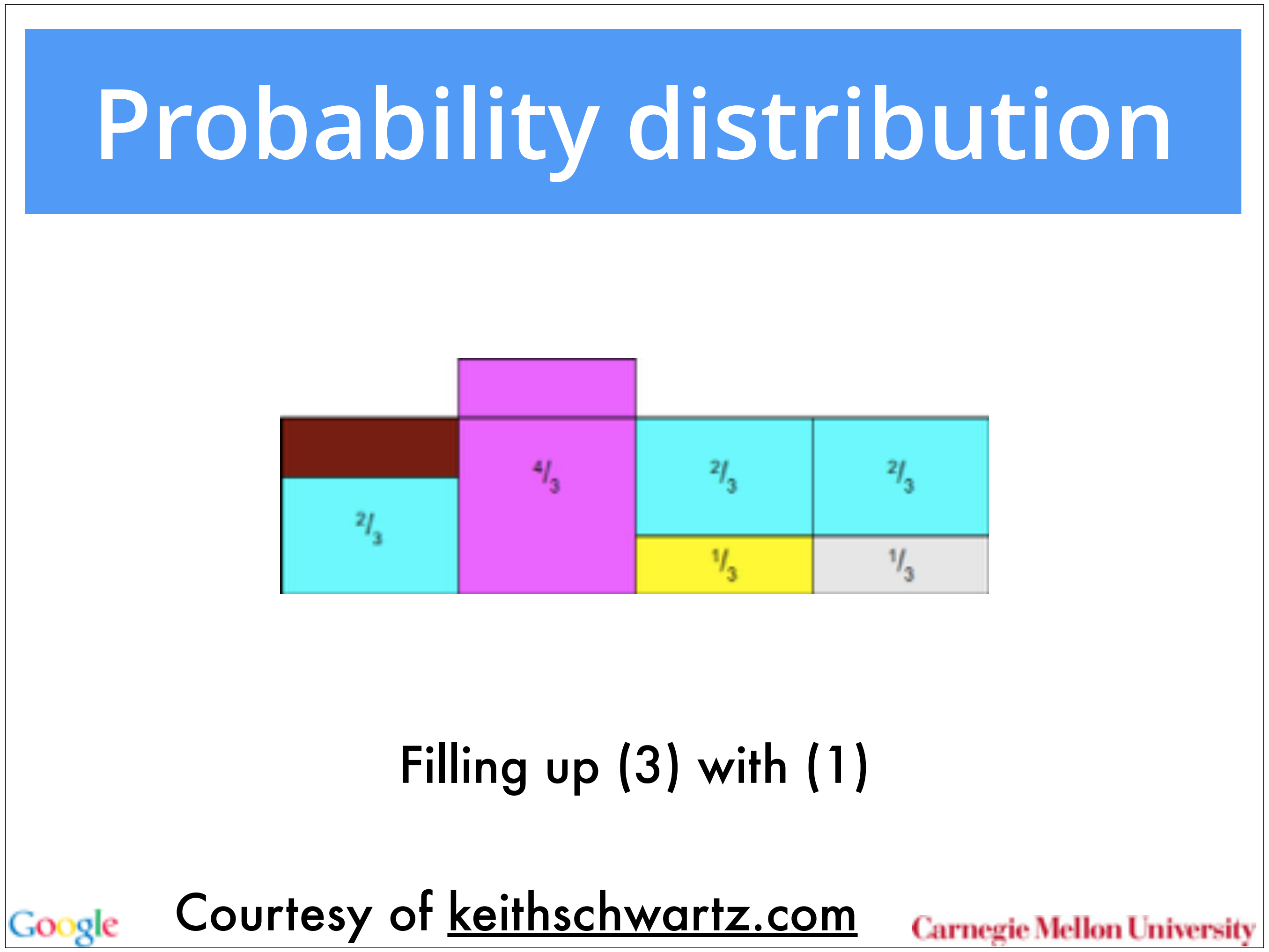


Filling up (4) with (1)

Courtesy of keithschwartz.com

Google Carnegie Mellon University

For each outcome with proportion less than the average, we take part of the proportion of another outcome to make it reach the average. As we take from another outcome, we keep track of its origin.



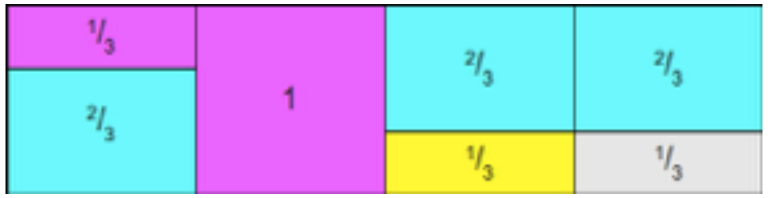
Probability distribution

Filling up (3) with (1)

Courtesy of keithschwartz.com

We keep doing this until all outcomes reach the average proportion. During the process some outcomes originally with more than the average proportion may fall below average.

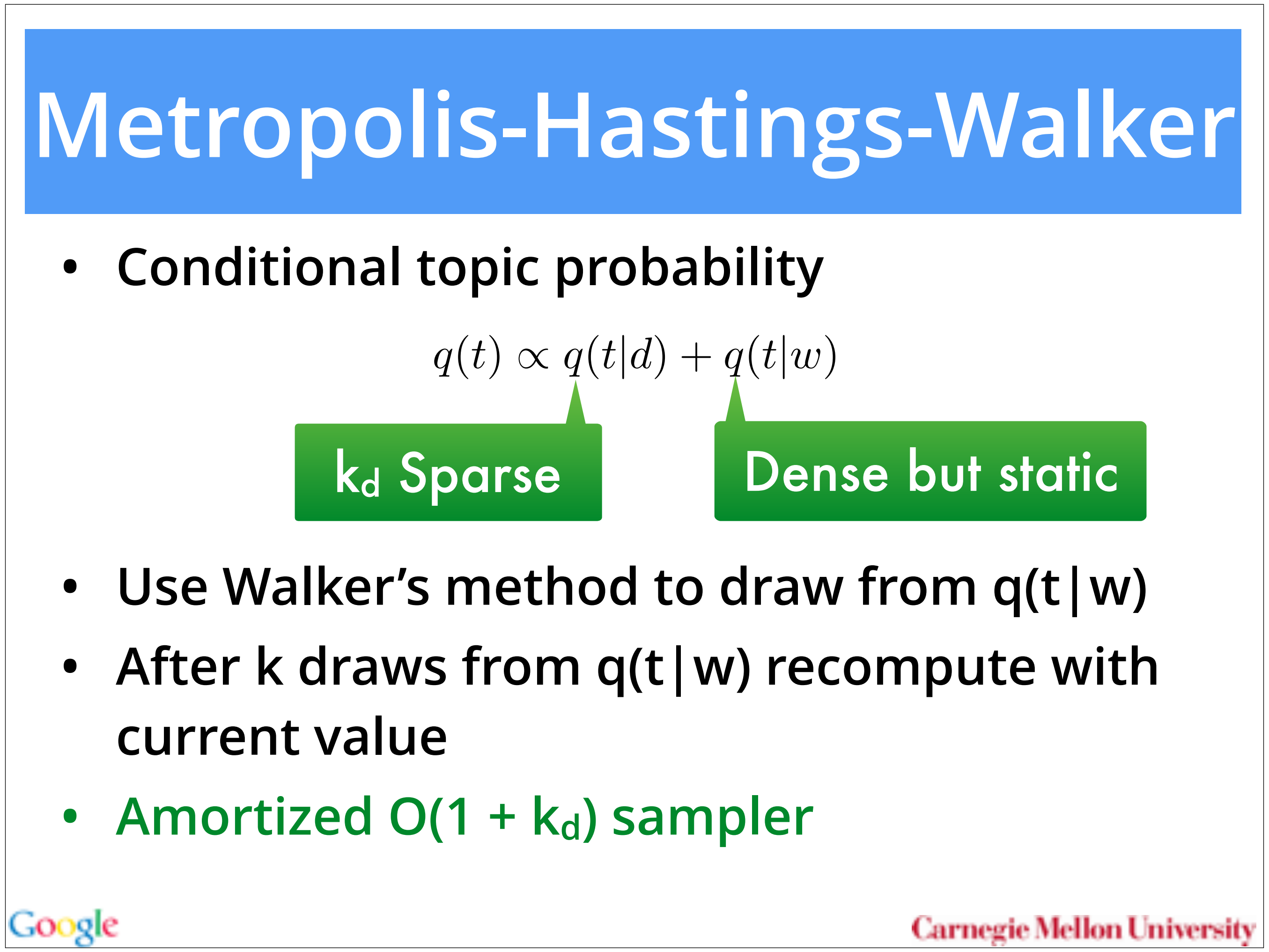
Probability distribution



Filling up (1) with (2)

Courtesy of keithschwartz.com

But it will eventually be compensated by others, at the time all outcomes reach exactly the average proportion. Obviously at this point each outcome is composed by no more than two parts, its original proportion, and part of a proportion from another outcome. To draw samples from this static table we only need to generate two random numbers: one decides which column, and the other decides which part.



Metropolis-Hastings-Walker

- Conditional topic probability

$$q(t) \propto q(t|d) + q(t|w)$$

k_d Sparse

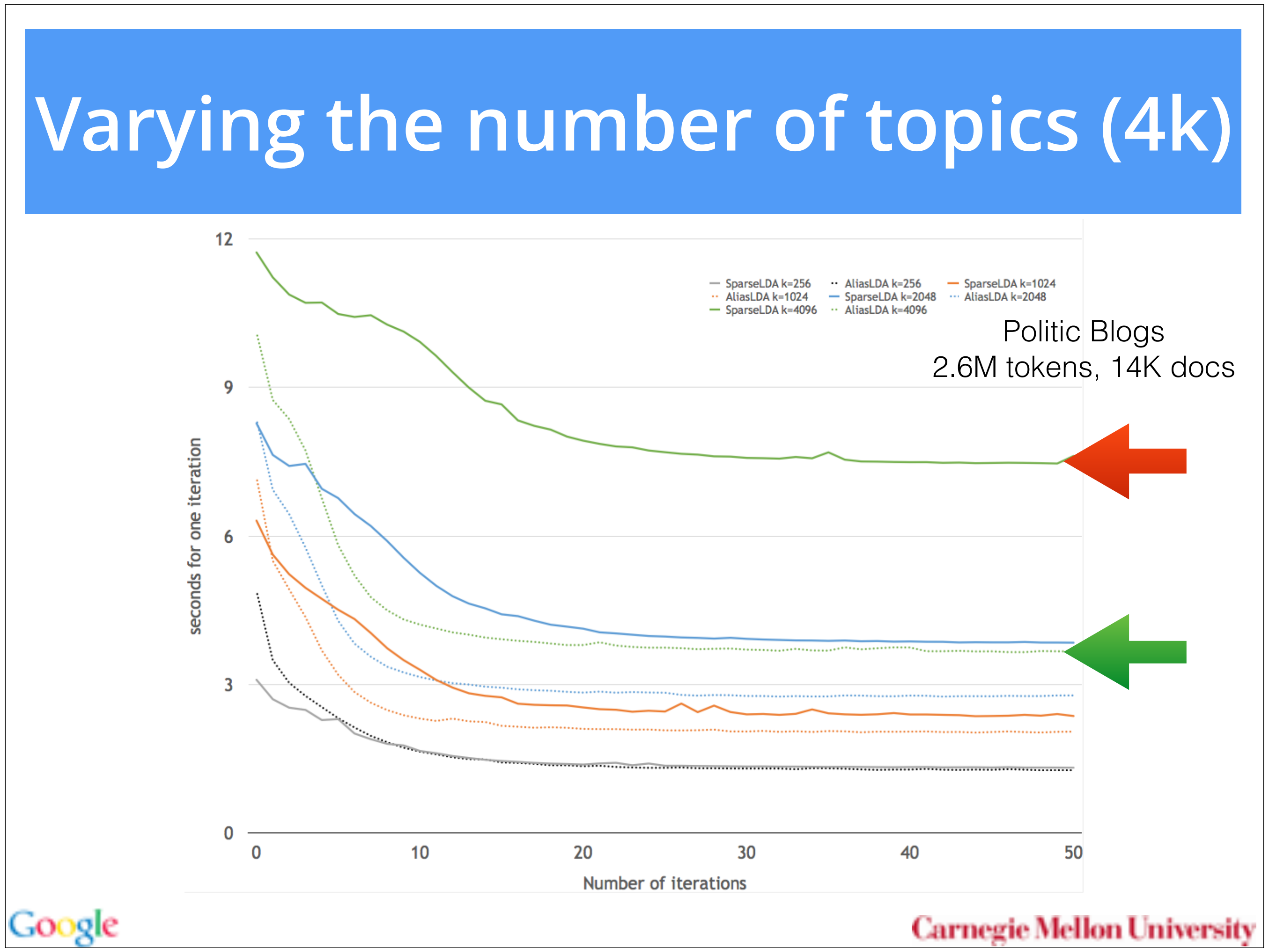
Dense but static

- Use Walker's method to draw from q(t|w)
- After k draws from q(t|w) recompute with current value
- Amortized O(1 + k_d) sampler

Back to our original sampling equation – the Walker's alias method is perfect way to compute a static approximation for the dense term and generate samples from it. The computation can take place in a background thread for every w repeatedly. This ensures the sampling complexity of our alias sampler is no more O(k_d), and the approximation distribution close enough to the real distribution for a good acceptance rate in metropolis hasting sampling.

Experiments

We are running out of time so let me show some key results I got in my implementation. Unless mentioned otherwise all results here are generated from my single thread C++ implementation running on my gaming laptop with 1.73GHz CPU.

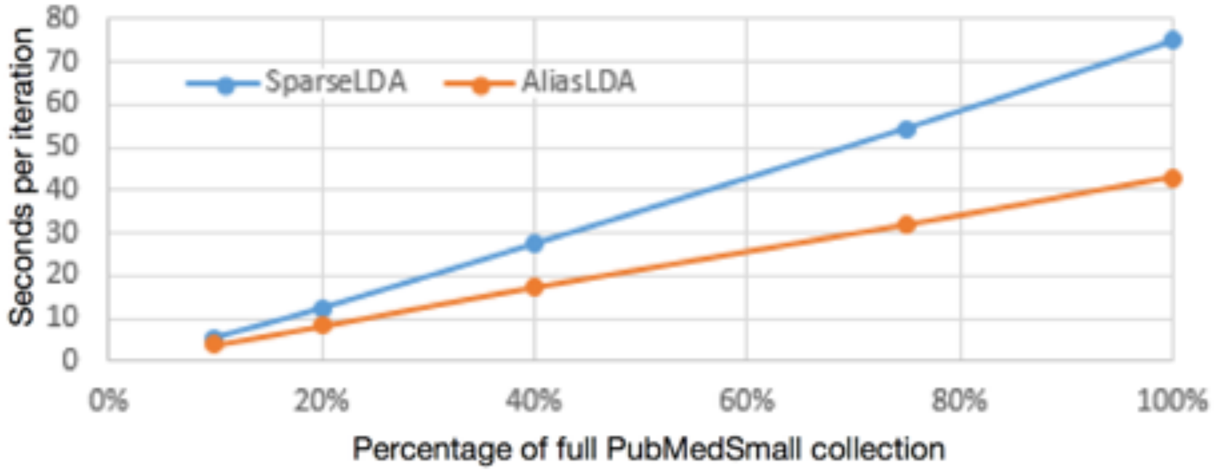


Varying the number of topics (4k)

Politic Blogs
2.6M tokens, 14K docs

This is the result between SparseLDA and AliasLDA, both my own implementation. We compare the running time against the number of iterations for a small collection with different number of topics. When the number of topics is 256, the running speed of AliasLDA and SparseLDA are about the same. They are the two grey lines in the bottom. When the number of topics scale up to 1024 and 2048, AliasLDA gets faster by about 10% and 30%. When the number of topics is 4096, AliasLDA is 100% faster than SparseLDA. The speed up growth is non-linear.

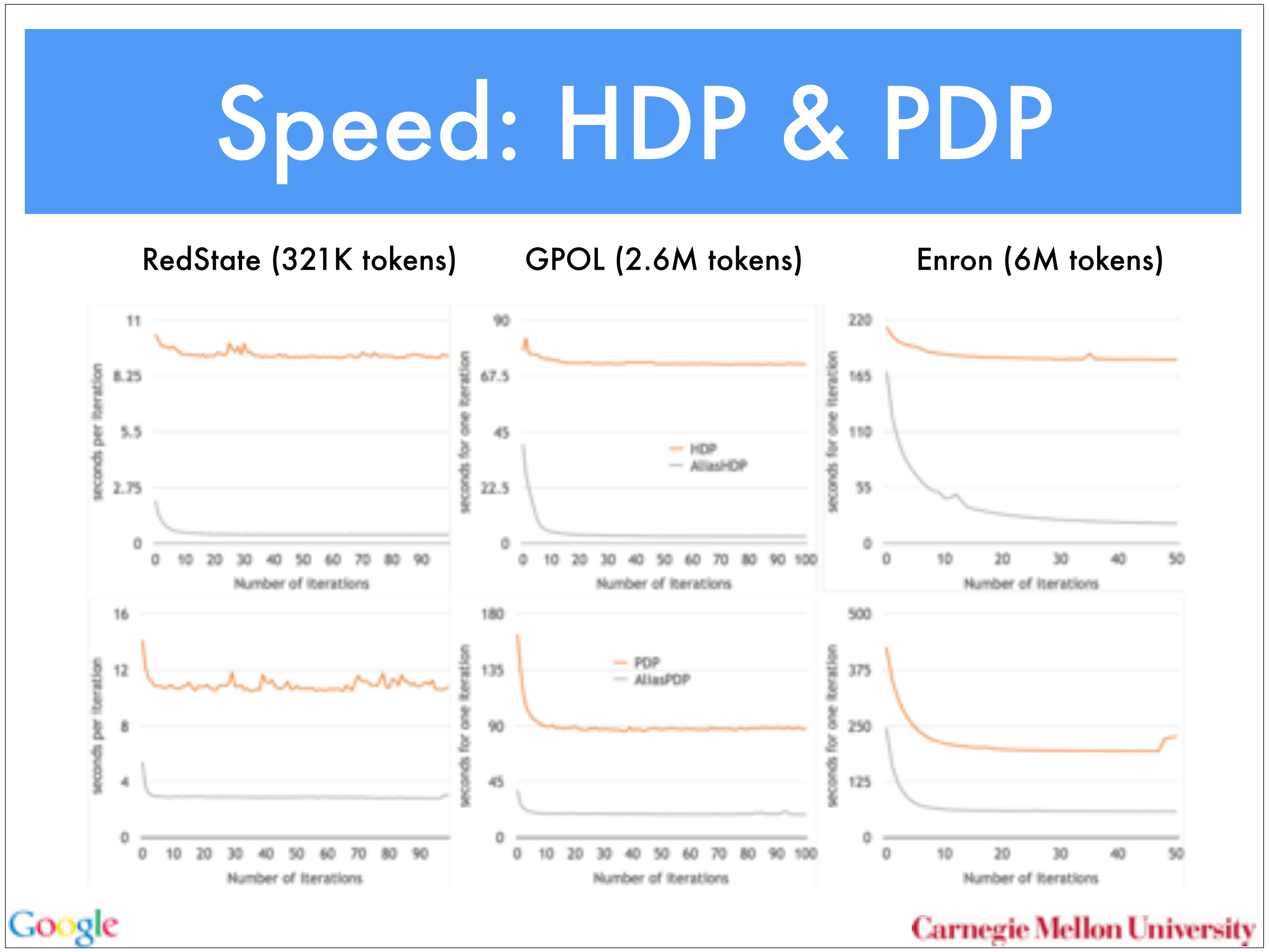
Varying data size



Google

Carnegie Mellon University

Similarly AliasLDA scales a lot better than SparseLDA when the amount of data gets larger.



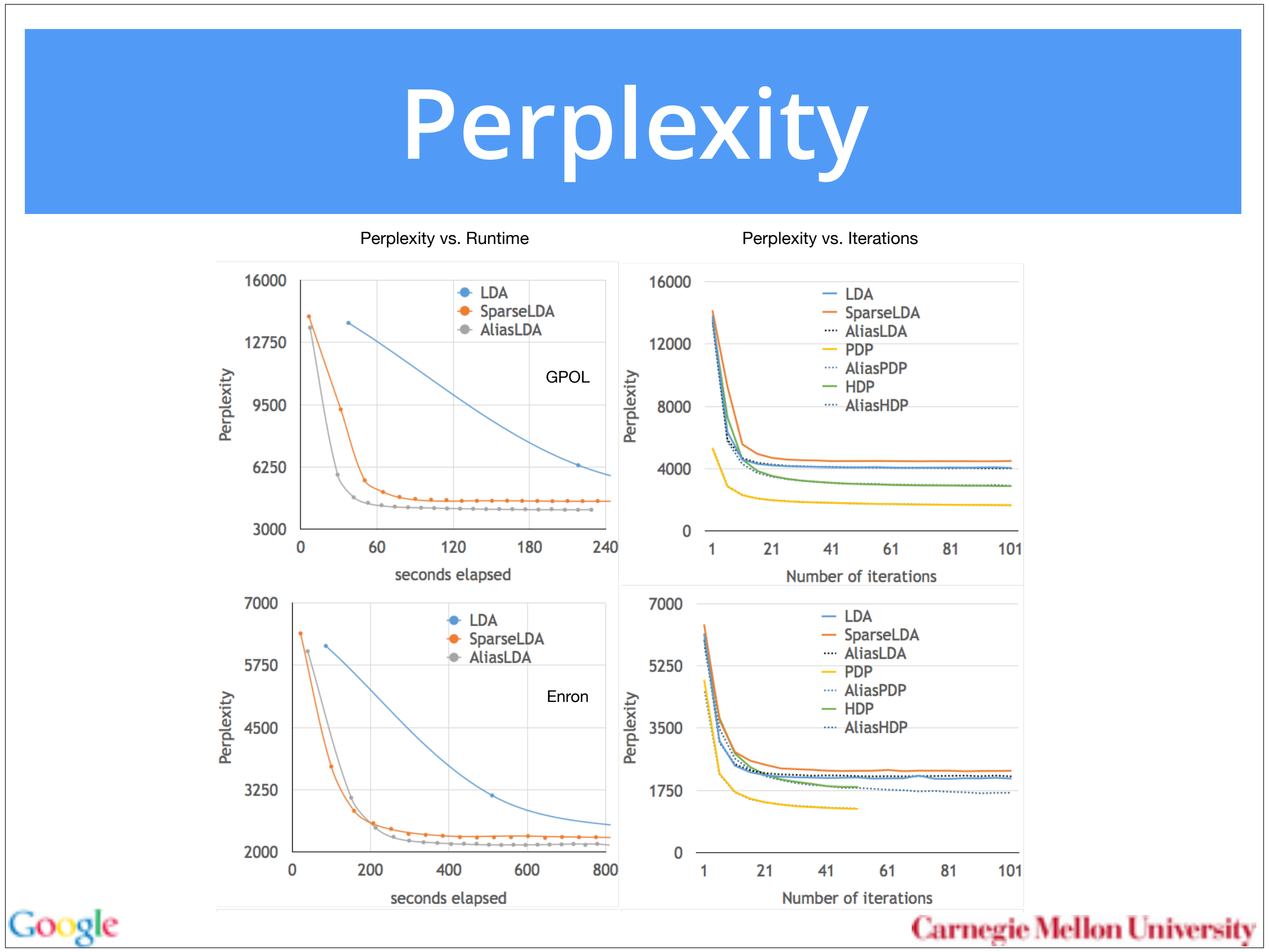
Speed: HDP & PDP

RedState (321K tokens) GPOL (2.6M tokens) Enron (6M tokens)

Google

Carnegie Mellon University

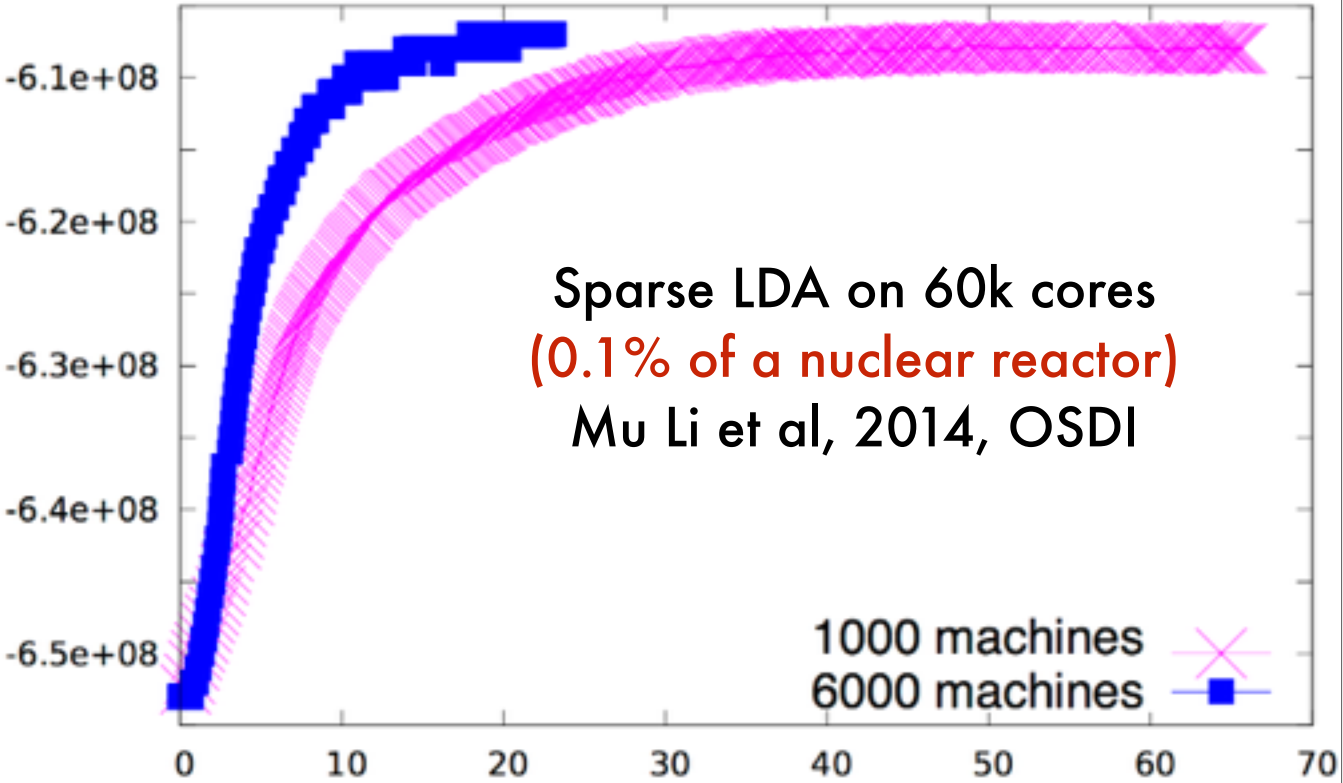
For sophisticated models like HDP and PDP, the time complexity is reduced from O(k) to O(k_d), and the speedup is huge.



Perplexity

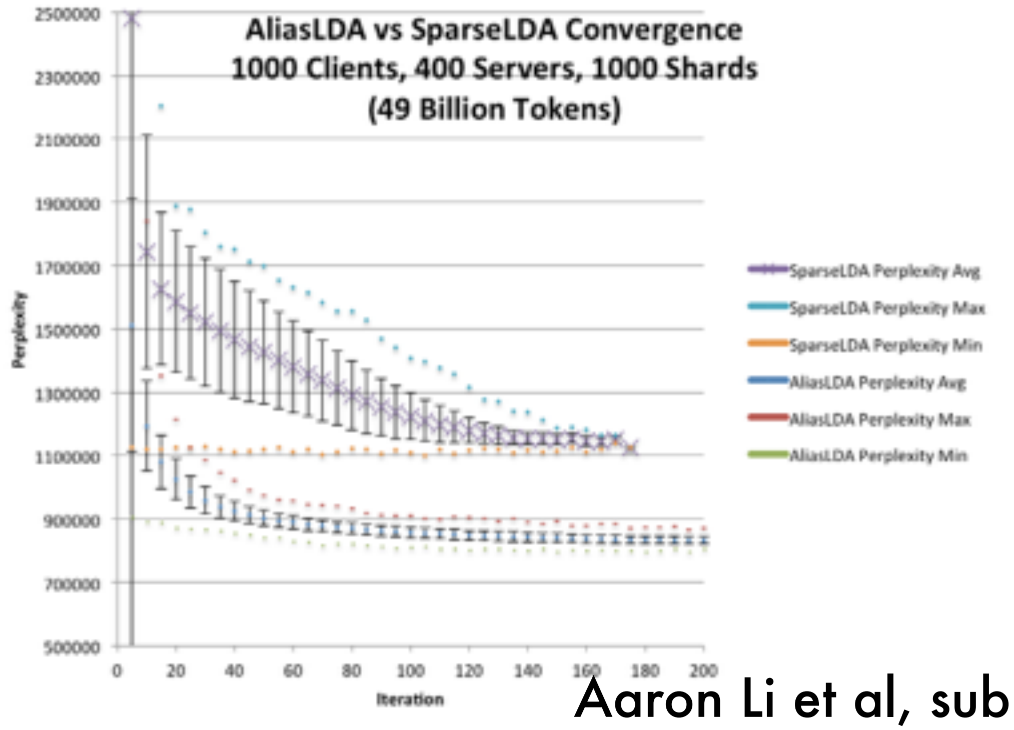
And, all of these speedup, for LDA and other models, comes without any sacrifice in convergence time or quality. The alias method and the original methods converge to the same perplexity.

And now in parallel



Sparse LDA on 60k cores
(0.1% of a nuclear reactor)
Mu Li et al, 2014, OSDI

Saving Nuclear Power Plants

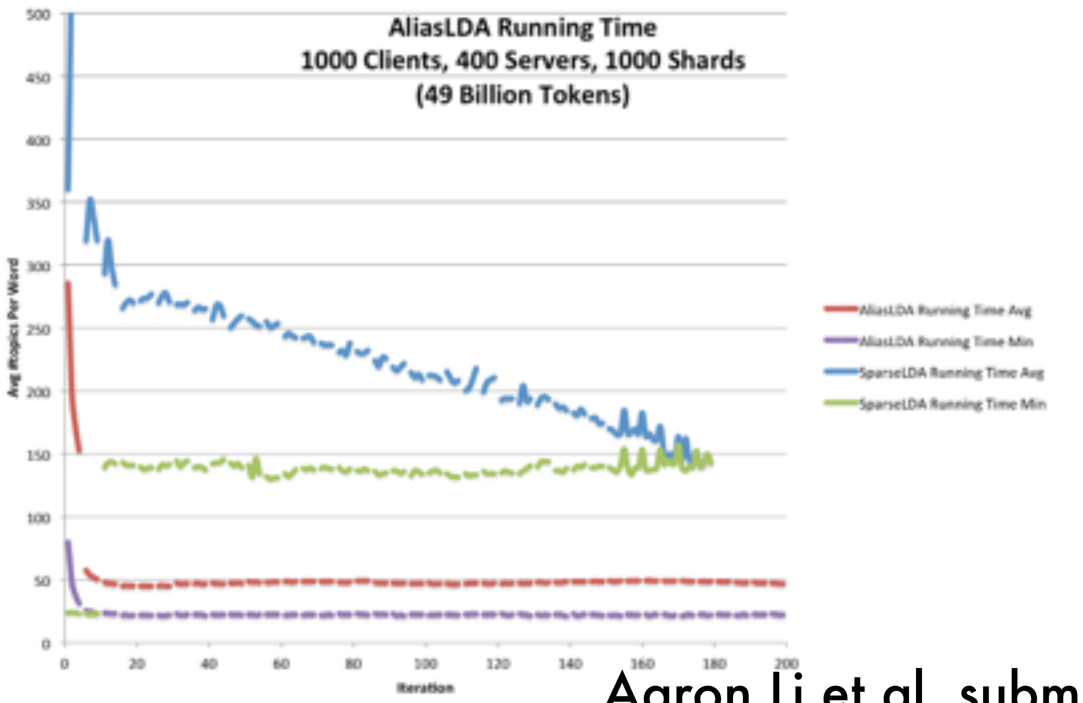


Aaron Li et al, submitted

Google

Carnegie Mellon University

Saving Nuclear Power Plants



Aaron Li et al, submitted

Summary

- Extends Sparse LDA concept of Yao et al.’09
  - Works for any sparse document model
  - Useful for many emissions models (Pitman Yor, Gaussians, etc.)
- Metropolis-Hastings-Walker
  - MH proposals on stale distribution
  - Recompute proposal after k draws for O(1)
- Fastest LDA sampler by a large margin