UNIVERSITY OF CAMBRIDGE

Pitfalls in the use of Parallel Inference for the Dirichlet Process

Yarin Gal, Zoubin Ghahramani

mlg.eng.cam.ac.uk/yarin

December 2013

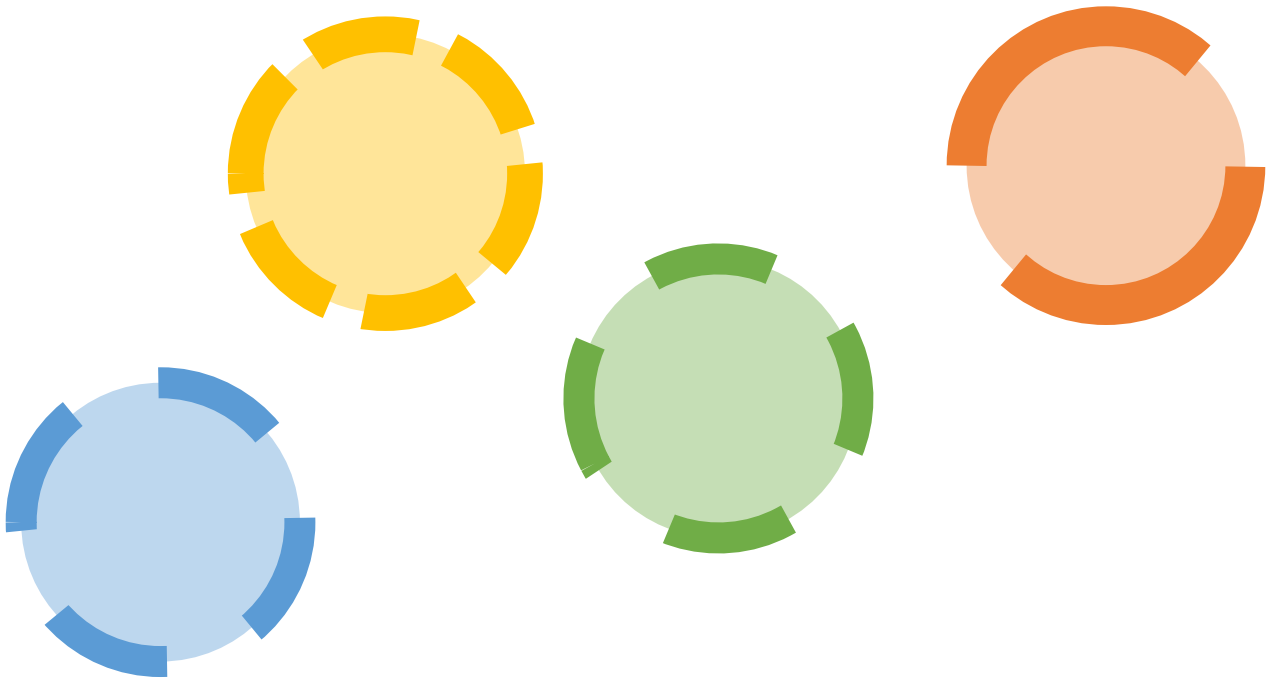
Outline

UNIVERSITY OF CAMBRIDGE

- The Dirichlet process
- Parallel inference
- Non-approximate parallel inference in the Dirichlet process
- What can go wrong
- How can we try to fix it
- Conclusions

UNIVERSITY OF CAMBRIDGE

Sampling from the Dirichlet process – the Chinese restaurant process



- A restaurant with 4 tables and 2, 4, 4, and 6 customers sitting around each one

Real world applications – Natural Language Processing

- Language modelling
  - A derivative model (the Hierarchical Pitman–Yor process) was shown to correspond to the state-of-the-art in language modelling
- Machine Translation
  - Used to obtain state-of-the-art results in Bayesian word alignment
- Working with huge datasets (tens of GBs)
- Development cycle taking weeks at a time
- Usually using small values for the concentration parameter ($\alpha = 0.1$ is common)

- Inference is slow!
- A common problem with non-parametric techniques
- Possible solutions:
  - Variational inference - an approximate approach
  - Parallel MCMC inference

Parallel inference

UNIVERSITY OF CAMBRIDGE

Given a network with many nodes (computers in a network or cores in a cluster), we would like to have an inference that:

- distributes the computational load evenly across the nodes,
- scales favourably with the number of nodes,
- has low overhead in the global steps,
- and converges to the true posterior distribution

Parallel inference in the DP

UNIVERSITY OF CAMBRIDGE

- Approximate parallel inference (Asuncion, Smyth, and Welling [2008])
  - Gives slower convergence (Williamson et al., [2013])
- Non-approximate parallel inference using a re-parametrisation of the Dirichlet process
  - Recently suggested, independently, by Lovell, Adams, and Mansingka [2012] and Williamson, Dubey, and Xing [2013]

Parallel inference in the DP

UNIVERSITY OF CAMBRIDGE

Given $\alpha > 0$ and base distribution H

$$\gamma \sim \text{Dir}_K(\alpha\mu_1, ..., \alpha\mu_K)$$
$$G_k \sim \text{DP}(\alpha\mu_k, H)$$
$$G = \sum_{k=1}^{K} \gamma_k G_k$$

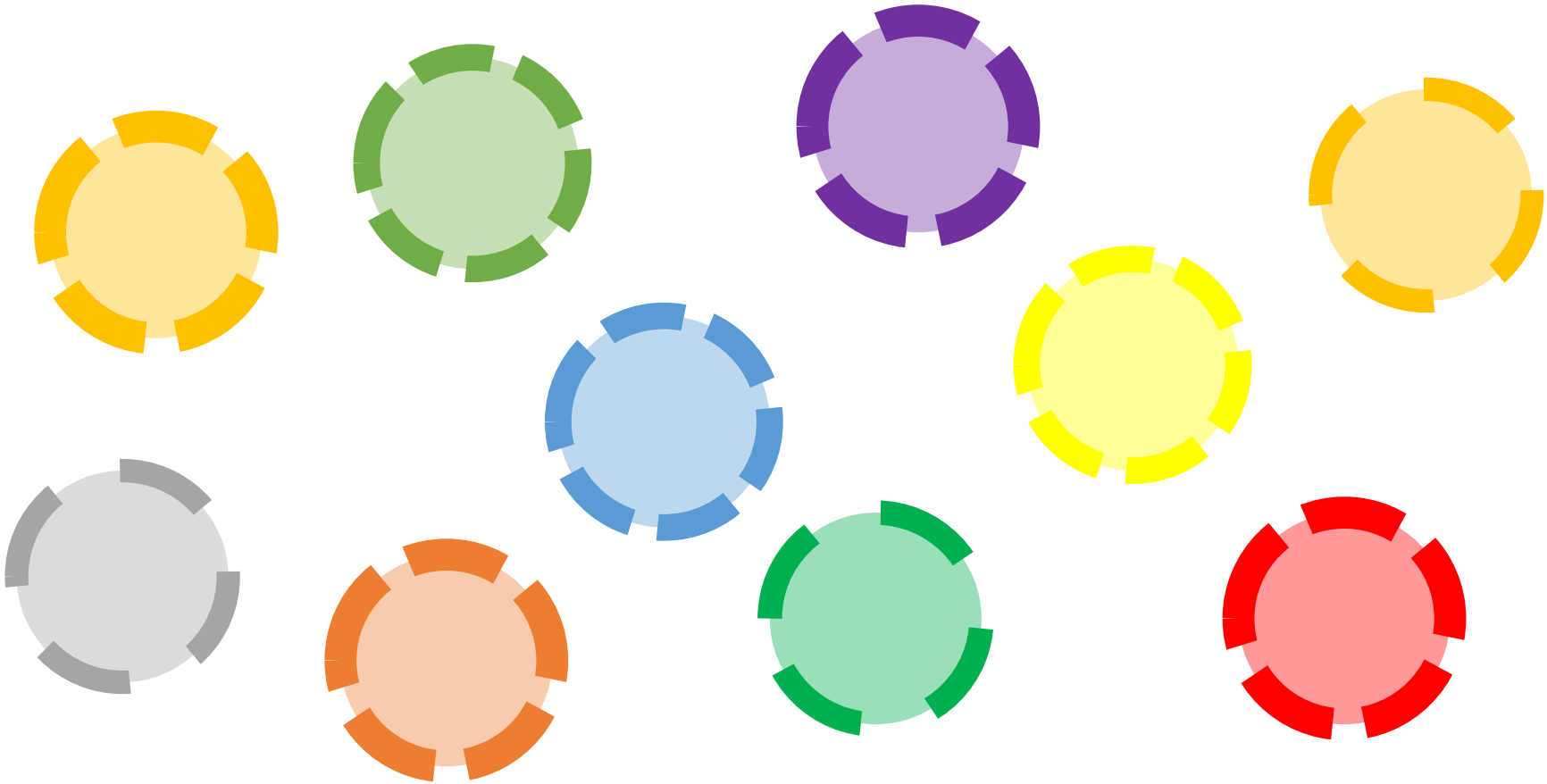
for some given $(\mu_k)_1^K$ where $\sum_{k=1}^K \mu_k = 1$ and $\mu_k \geq 0$ (one would usually choose $\mu_1 = \frac{1}{K}, ..., \mu_K = \frac{1}{K}$)

- We sample γ to decide how much data to send to each node,
- perform DP inference independently on each one,
- and collect the results at the end

Parallel inference in the DP

UNIVERSITY OF CAMBRIDGE

For a network with 10 nodes we split the data using a sample from a Dirichlet distribution with 10 components:

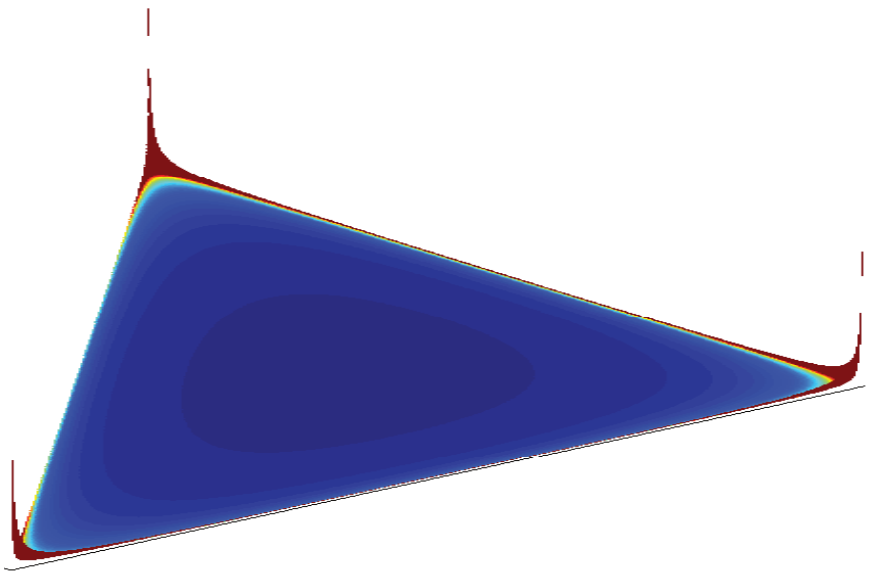


- Each table corresponding to a single node and each customer to a data point sent to that node

UNIVERSITY OF CAMBRIDGE

However...

- Samples from the Dirichlet distribution with parameter smaller than 1 have most of the mass concentrated around the corners of the simplex



However...

UNIVERSITY OF CAMBRIDGE

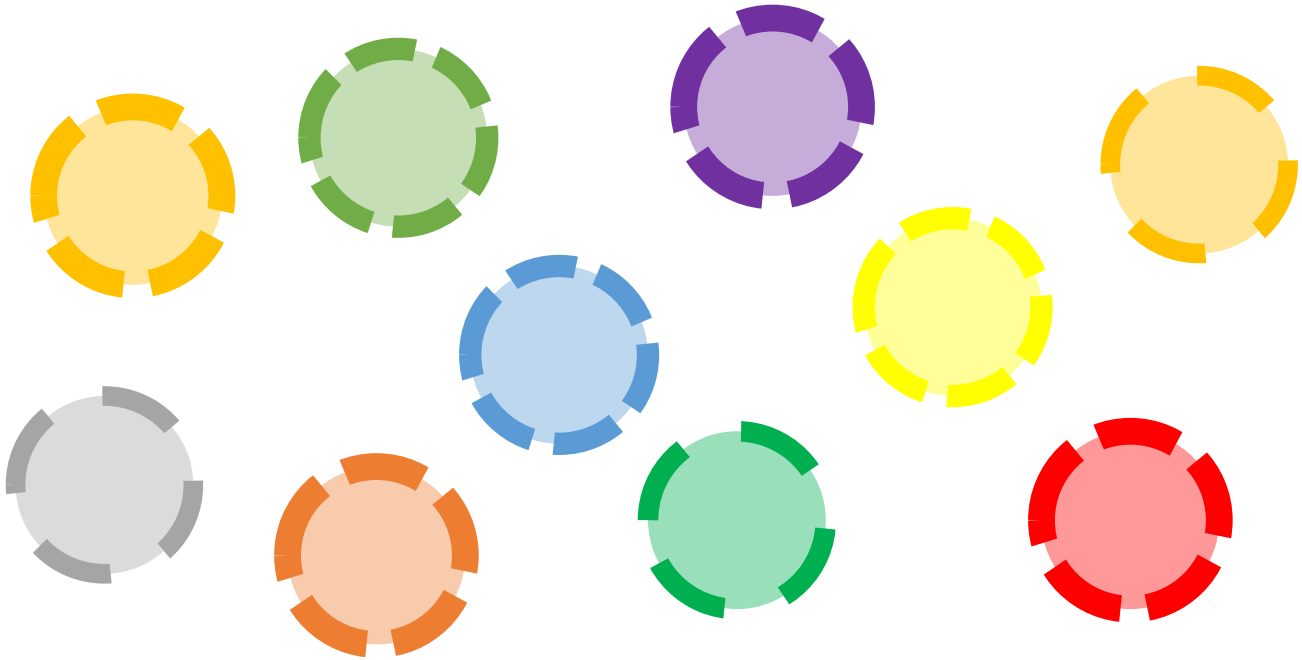
- and in the limit of K we obtain samples from the Dirichlet process with parameter α:

$$\mathrm{DP}(\alpha)$$

- This means that the expected number of nodes used is the same as the expected number of tables in a restaurant with parameter α
  - (we can augment the number of nodes by sending multiple jobs to the same machine)

However...

Actual samples from a Dirichlet process with 50 data points don't look like this:



- The expected number of tables in a restaurant with n customers is given by

$$\alpha \log(n)$$

However...

UNIVERSITY OF CAMBRIDGE

So a sample from a Dirichlet process with 50 data points would look more like this:

Which means that only a constant number of nodes, dependent on the number of data points, would be used.

However...

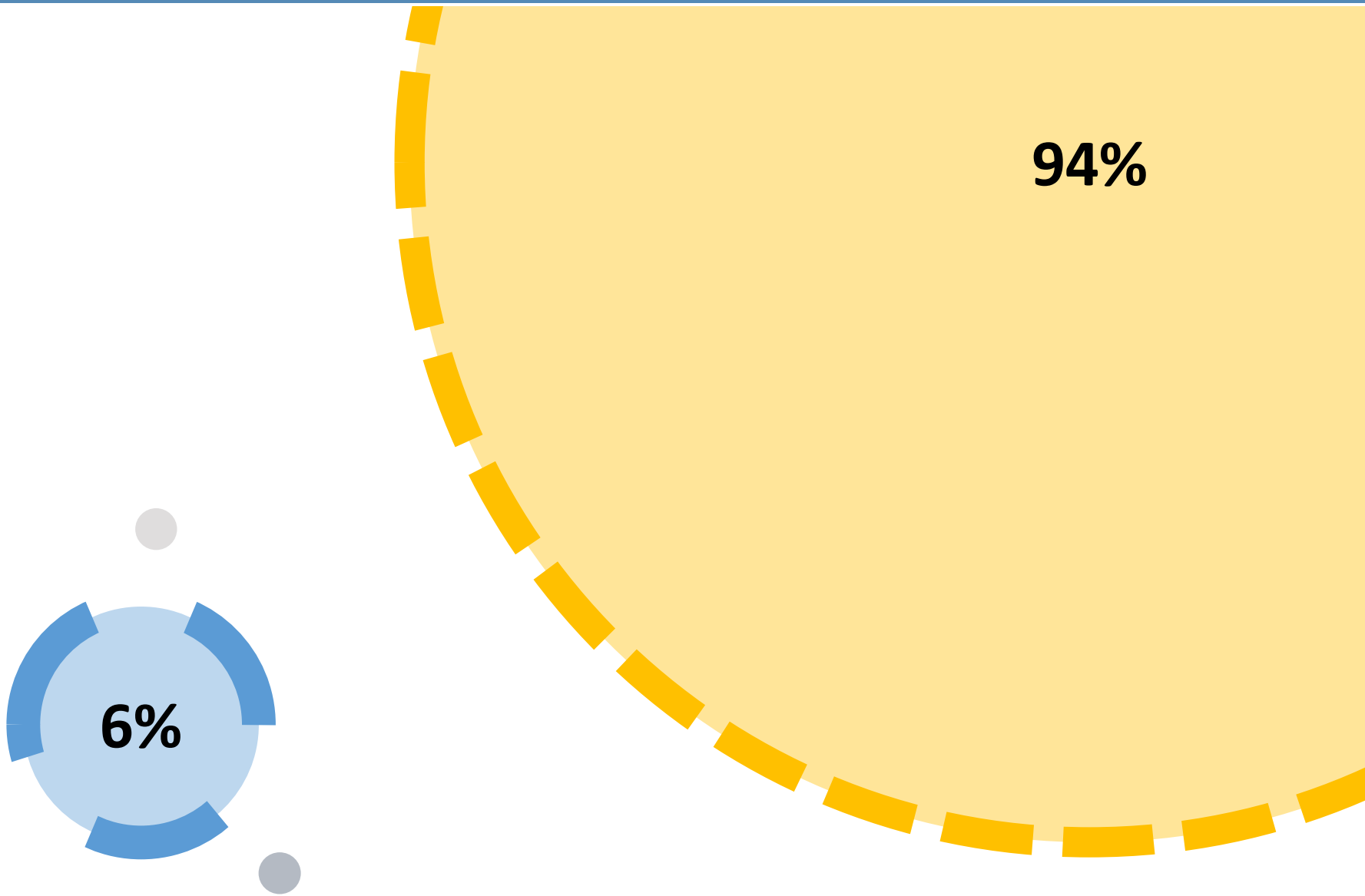
UNIVERSITY OF CAMBRIDGE

Even worse, the sizes of the different tables follows an exponential decay, so the the number of customers sitting next to each table would actually be

$$C, \quad Cq, \quad Cq^2, \quad Cq^3, \quad \cdots$$

for $q = \dfrac{\alpha}{1+\alpha}$ and $C = \dfrac{1}{1+\alpha}$, so an actual sample would be...

However...



for $n = 50$ data points and $\alpha = 0.1$.

However...

UNIVERSITY OF CAMBRIDGE

So for $n = 50$ data points and $\alpha = 0.1$ the parallel inference would send 94% of the data to a single machine.

- Sampling from the *finite* **Dirichlet distribution** with K components (nodes in a network) and different parameter values we get a load balance:

# of nodes	$\alpha = 0.1$	$\alpha = 2$
$K = 10^1$	94%, 6%, 0%, 0%, ...	54%, 23%, 12%, 6%, ...
$K = 10^2$	94%, 6%, 0%, 0%, ...	48%, 22%, 12%, 7%, ...
$K = 10^3$	94%, 6%, 0%, 0%, ...	48%, 21%, 12%, 7%, ...
$K = 10^4$	94%, 6%, 0%, 0%, ...	48%, 21%, 12%, 7%, ...
$K = 10^5$	94%, 6%, 0%, 0%, ...	48%, 21%, 12%, 7%, ...

Figure: Average load on each node in decreasing order

However...

UNIVERSITY OF CAMBRIDGE

And in general, for a Dirichlet process with parameter α, 95% of the data for would be sent to

$$\approx \frac{1.3}{\log(\alpha + 1) - \log(\alpha)}$$

nodes,

- independently of the size of the dataset,
- independently of the number of nodes in the network,
- and dependent only on the **parameter used to model the data**

What can we do?

- We can try to initialise the sampler near the posterior
- We might want to use the Pitman–Yor process to distribute the data
- We could use approximate inference with Metropolis–Hastings corrections
- We can develop better approximate inference approaches
- Don’t use the Dirichlet process
- Use a partly-parametric partly-non-parametric approach

What can we do?

UNIVERSITY OF CAMBRIDGE

We can try to initialise the sampler near the posterior

- When we know the data has **many clusters** which are **evenly balanced**
- Initialise the sampler randomly with many evenly sized clusters
- ... however still doesn't answer many real-world cases
- ... and the distribution of the clusters between the nodes has the same skewed balance

What can we do?

UNIVERSITY OF CAMBRIDGE

We could use the Pitman–Yor process to distribute the data

- This would yield

$$\propto n^{\alpha}$$

  number of tables used

- ... but still an exponential decay in the sizes of the tables

What can we do?

We could also use approximate inference with Metropolis–Hastings corrections

- Was suggested by Doshi, Knowles, Mohamed, and Ghahramani [2009] for the Indian-buffet process
- We sample from the Markov chain and use Metropolis–Hastings corrections to discard some of the samples
- Recently implemented by Chang and Fisher III [2013]
- ... but we should be careful with the global step overhead

What can we do?

UNIVERSITY OF CAMBRIDGE

Develop better approximate inference

- Current approach uses Gibbs sampling after distributing the data evenly across the different nodes and in the global step we sync. the state of the nodes (Asuncion, Smyth, and Welling [2008])
- Was reported by Williamson et al., [2013] to have slow convergence

What can we do?

UNIVERSITY OF CAMBRIDGE

Don’t use the Dirichlet process

- Recently shown that the Dirichlet process is inconsistent in the number of cluster
- An alternative distribution for clustering has been suggested: using a Poisson distribution mixture of Dirichlet distributions
- Might open the door for more efficient parallel inference

What can we do?

UNIVERSITY OF CAMBRIDGE

And finally, use a partly-parametric partly-non-parametric mixture of the Dirichlet-K distribution and Dirichlet process for clustering

- Would allow us to use an unbounded number of clusters with at least K clusters
- The distribution of the load would be partly-balanced

Conclusions —

- Scaling up inference for the Dirichlet process is still an open problem
- ... which has to be solved if we want it to be used in industry and real-world applications!