Parallel Markov Chain Monte Carlo for Pitman-Yor Mixture Models

Avinava Dubey
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Sinead A. Williamson
McCombs School of Business
University of Texas at Austin
Austin, TX 78712

Eric P. Xing
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

The Pitman-Yor process provides an elegant way to cluster data that exhibit power law behavior, where the number of clusters is unknown or unbounded. Unfortunately, inference in Pitman-Yor process-based models is typically slow and does not scale well with dataset size. In this paper we present new auxiliary-variable representations for the Pitman-Yor process and a special case of the hierarchical Pitman-Yor process that allows us to develop parallel inference algorithms that distribute inference both on the data space and the model space. We show that our method scales well with increasing data while avoiding any degradation in estimate quality.

1 INTRODUCTION

Bayesian nonparametric priors such as the Dirichlet process allow us to create flexible probabilistic models with an unbounded number of parameters. These models are appropriate when the latent dimensionality of our data is unknown or may grow with sample size. Unfortunately, inference in such models is often unwieldy, due to the high number of instantiated parameters and the need to discover the appropriate number of parameters for a given data set.

There has been growing interest in scalable inference algorithms for Bayesian nonparametric models. Earlier attempts at distributing inference were either highly model-specific (Doshi-Velez et al., 2009), or introduced additional approximation into the inference procedure and were mostly concentrated at distributed learning on data and not on the model (Asuncion et al., 2008). More recent approaches (Williamson et al., 2013; Lovell et al., 2012; Chang and Fisher III, 2013) have used both model- and data-parallel design to infer the latent structure without introducing additional approximation.

Most previous research on scalable inference in Bayesian nonparametrics has focused on parallel inference for Dirichlet process-based models. Dirichlet process models are not ideal for modeling language, as they do not capture the power-law behavior often found in text data (Zipf, 1935). The Pitman-Yor process (Perman et al., 1992; Pitman and Yor, 1997) is a two-parameter extension to the Dirichlet process that allows heavier-tailed distributions over partitions. It is therefore often used in text and language applications, because it more accurately matches the statistics of natural language (Goldwater et al., 2006; Teh, 2006a), and can be used to build hierarchical models for text that out-perform their Dirichlet process-based counterparts (Teh, 2006b; Wood et al., 2009; Blunsom and Cohn, 2011). However, inference remains a bottleneck.

In this paper, we address this issue using an approach pioneered for the Dirichlet process by Williamson et al. (2013) and Lovell et al. (2012): We construct an alternative representation of a nonparametric process that incorporates conditional independencies, and use these conditional independencies to divide our model (and in doing so, our data) into sub-models that can be learned in parallel.

The key to achieving this lies in the introduction of new representations for the Pitman-Yor process and a hierarchical extension, presented in Section 3. These representations afford the conditional independence structure required to develop model- and data-parallel inference algorithms, as demonstrated in Section 4. In Section 5, we perform a thorough evaluation of our inference algorithms and of the modeling assumptions made.We show that our hierarchical model, which is a special case of the hierarchical Pitman-Yor process (Teh, 2006b), is a good fit for natural language. We empirically demonstrate that we can speed up computation in Pitman-Yor process mixture models and hierarchical Pitman-Yor process models with no deterioration in performance, and show good results across a range of dataset sizes and data dimensionalities.

2 BACKGROUND

In this section, we will review the Pitman-Yor process and the hierarchical Pitman-Yor process, and discuss existing approaches for parallelization in Bayesian nonparametric models.

2.1 THE PITMAN-YOR PROCESS

The Dirichlet process (Ferguson, 1973) is a distribution over probability measures of the form $D := \sum_{k=1}^{\infty} \pi_k \delta_{\phi_k}$, parametrized by a concentration parameter $\alpha > 0$ and a probability measure H. The order statistics of the atom sizes π_k are described by the following stick-breaking distribution:

$$\begin{aligned} \pi_k =& w_k \prod_{j=1}^{k-1} (1 - w_j) \\ w_j \sim& \text{Beta}(1, \alpha) \end{aligned} \tag{1}$$

and the atom locations ϕ_k are sampled i.i.d. from H. The resulting probability measure D can be used to cluster observations; a finite number of observations will belong to a finite (but random) number of clusters.

The Pitman-Yor process (Perman et al., 1992; Pitman and Yor, 1997) is a two-parameter extension of the Dirichlet process, parametrized by a discount parameter $0 \leq d \leq 1$, a concentration parameter $\alpha > -d$, and a probability measure H. When the discount parameter is zero, we recover the Dirichlet process. As the discount parameter increases, we get increasingly heavy-tailed distributions over the atom sizes in the resulting probability measure. We can see this behavior by considering the stick-breaking process for the Pitman-Yor process:

$$\begin{aligned} \pi_k =& w_k \prod_{j=1}^{k-1} (1 - w_j) \\ w_j \sim& \text{Beta}(1 - d, \alpha + jd). \end{aligned} \tag{2}$$

As d increases, the rate of decay of the ordered atom sizes will decrease. When $d = 0$, we recover the stick-breaking construction for the Dirichlet process given in Equation 2. This behavior makes the Pitman-Yor process particularly appropriate for applications in language modeling. Natural language has long been known to exhibit power-law behavior (Zipf, 1935), and the Pitman-Yor process is able to capture this (Teh, 2006a).

We can use the Pitman-Yor process to cluster data using the following mixture model:

$$D \sim \text{PY}(\alpha, d, H) \qquad \theta_i | D \sim D \qquad x_i | \theta_i \sim f(\theta_i). \tag{3}$$

We can also construct a hierarchy of Pitman-Yor processes (Teh, 2006a) that allows us to jointly cluster multiple related groups of data. Each group is associated with a Pitman-Yor process-distributed random measure, and the group-specific Pitman-Yor processes are coupled via a shared, Pitman-Yor process-distributed base measure. For M groups, each containing N_m data points, the generative process is

$$\begin{aligned} D_0 &\sim \text{PY}(\alpha, d, H) \\ D_m | D_0 &\sim \text{PY}(\gamma, c, D_0), \quad m = 1, \ldots, M \\ \theta_{mi} | D_m &\sim D_m, \quad i = 1, \ldots, N_m \\ x_{mi} | \theta_{mi} &\sim f(\theta_{mi}). \end{aligned} \tag{4}$$

This distribution has found a number of applications in text and language modeling (Teh, 2006b; Wood et al., 2009; Blunsom and Cohn, 2011).

2.2 PARALLEL METHODS FOR BAYESIAN NONPARAMETRICS

Bayesian nonparametric models allow an unbounded number of parameters, and can increase the number of parameters used as we see more data. This makes them appealing for large, complex, and potentially growing datasets. Unfortunately, naive implementation of Gibbs samplers, such as those developed in Ishwaran and James (2001), Neal (1998) and Teh et al. (2006), do not scale well to such datasets.

To counter issues of scalability, a number of authors have attempted to parallelize inference in Bayesian nonparametric models. Such algorithms typically rely on *data parallelization* – data is split onto multiple processors, and messages are passed between processors. Often, this involves making approximations that break long-range dependencies. For example in Asuncion et al. (2008) and Doshi-Velez et al. (2009), each processor maintains local sufficient statistics for the data stored on it, and approximates the full sufficient statistics by combining the local statistics with snapshots of the local statistics from other processors.

Such an approach typically leads to inaccuracies in the estimates of the global parameters or sufficient statistics. This is particularly true in Bayesian nonparametric models, where we have many components with a small number of observations. Combining the local statistics for these components is difficult, and Williamson et al. (2013) show that this leads to estimate deterioration in the case of Dirichlet processes and hierarchical Dirichlet processes. In models with power law behavior, this effect is likely to be more pronounced, due to the larger number of components with very few associated data points.

An alternative approach is to explicitly partition the model into sub-models that are independent or conditionally independent. Inference is performed on each sub-model independently, and the results are combined globally. We call such algorithms *model-parallel*. Such models are typically also data-parallel, with different sub-models governing different subsets of the data. They also have the advantage that the sub-models typically have a smaller space of latent parameters than the full model.

Recent examples of algorithms that are both data-parallel and model-parallel are given by Williamson et al. (2013) and Lovell et al. (2012), who use auxiliary variable model representations for Dirichlet processes and hierarchical Dirichlet processes to obtain conditional independence. These algorithms hinge on the fact that we can write a Dirichlet process mixture model as a mixture of Dirichlet process mixture models, as follows:

$$\begin{aligned} D_j &\sim \text{DP}(\alpha_j, H_j), \\ \phi &\sim \text{Dirichlet}(\alpha_1, \dots, \alpha_P), \\ \mu_i|\phi &\sim \phi, \\ \theta_i|\mu_i, D_1, \dots, D_P &\sim D_{\mu_i}, \\ x_i &\sim f(\theta_i). \end{aligned} \tag{5}$$

The marginal distribution over the x_is is the equal in distribution to that obtained by the Dirichlet process mixture model

$$\begin{aligned} D &\sim \text{DP}\left(\sum_j \alpha_j, \frac{\sum_j \alpha_j H_j}{\sum_j \alpha_j}\right), \\ \theta_i|D &\sim D, \\ x_i|\theta_i &\sim f(\theta_i). \end{aligned} \tag{6}$$

Conditioned on the μ_is in Equation 5, we can split our model into conditionally independent sub-models involving disjoint subsets of the data, achieving both model- and data-parallelization.

3 AUXILIARY VARIABLE REPRESENTATIONS

In this section, we introduce new representations for the Pitman-Yor process and hierarchical Pitman-Yor process, that will allow us to develop model- and data-parallel inference algorithms.

3.1 AUXILIARY VARIABLE REPRESENTATION FOR THE PITMAN-YOR PROCESS

To obtain an auxiliary variable representation, we first show that a Pitman-Yor mixture model with positive concentration parameter α and continuous base measure H can be constructed as a finite mixture of Pitman-Yor mixture models. We start with a little-used representation of the atom sizes of the Pitman-Yor process.

Theorem 1 (Mixture model representation of a Pitman-Yor process)**.** *Let $G_0 := \sum_k \rho_k \delta_{\theta_k} \sim DP(\alpha, H_0)$, and let $G_k := \sum_j \pi_{j,k}\delta_{\phi_{j,k}} \overset{i.i.d.}{\sim} PY(0, d, H)$, where H is a continuous probability measure (note that this is a normalized stable process with stable parameter d). Then $D = \sum_k \rho_k G_k$ is distributed according to a Pitman-Yor process with concentration parameter α, discount parameter d, and base measure H.*

Proof. This is a direct consequence of Proposition 22 in Pitman and Yor (1997). □

By extension, we can express a Pitman-Yor mixture model as a Dirichlet process mixture of normalized stable process mixture models, provided the concentration parameter α of the Pitman-Yor process is strictly positive and the base measure H is continuous.

Corollary 1. *The marginal distribution over the data $(x_i, i = 1, \dots, N)$ implied by the generative procedure*

$$\begin{aligned} G &\sim GEM(\alpha) \\ D_j &\sim PY(d, 0, H) \\ t_i|G &\sim G \\ \theta_i|t_i, D_1, D_2, \dots &\sim D_{t_i} \\ x_i|\theta_i &\sim f(\theta_i) \end{aligned} \tag{7}$$

is the same as the marginal distribution over the x_i obtained using the Pitman-Yor mixture model of Equation 3.

Proof. The proof is a straightforward extension of Theorem 1. □

We have therefore reduced a Pitman-Yor mixture model with concentration parameter $\alpha > 0$ to a Dirichlet process mixture model. This allows us to apply Equation 5 and write our Pitman-Yor mixture model as a finite Dirichlet mixture of Pitman-Yor mixture models, providing the conditional independence required to construct a model-parallel sampler.

Theorem 2 (Auxiliary variable representation for Pitman-Yor mixture models)**.** *Provided the concentration parameter $\alpha > 0$ and the base probability measure H is continuous, we can rewrite the generative process for the Pitman-Yor mixture model given in Equation 3 as:*

$$\begin{aligned} D_j &\sim PY\left(\frac{\alpha}{P}, d, H\right), \\ \phi &\sim Dirichlet\left(\frac{\alpha}{P}, \dots, \frac{\alpha}{P}\right), \\ \mu_i|\phi &\sim \phi, \\ \theta_i|\mu_i, D_1, \dots, D_P &\sim D_{\mu_i}, \\ x_i|\theta_i &\sim f(\theta_i), \end{aligned} \tag{8}$$

for $j = 1, \dots P$ and $i = 1, \dots, N$. The marginal distribution over the x_i remains the same.

Proof. Since we can write the Pitman-Yor mixture model as a Dirichlet process mixture model, this follows as a direct application of Equation 5. An alternative proof is given in the supplement. □

3.2 AUXILIARY VARIABLE REPRESENTATION FOR THE HIERARCHICAL PITMAN-YOR PROCESS

The results in Section 3.1 can be extended to certain special cases of the hierarchical Pitman-Yor process described in Equation 4. Unfortunately, we can only apply Theorem 1 and Corollary 1 when the base measure of the Pitman-Yor process is continuous. For the group-level Pitman-Yor processes in Equation 4, this is not the case.

The auxiliary variable representation for the Dirichlet process given in Equation 5, however, does not require a continuous base measure. We note that the Dirichlet process is a special case of the Pitman-Yor process, with discount parameter $d = 0$. We therefore work with the following special case of the hierarchical Pitman-Yor process:

$$\begin{aligned} D_0 &\sim \text{PY}(\alpha, d, H) \\ \gamma &\sim \text{Gamma}(\alpha) \\ D_m|D_0 &\sim \text{DP}(\gamma, D_0), \quad m = 1, \ldots, M \\ \theta_{mi}|D_m &\sim D_m, \quad i = 1, \ldots, N_m \\ x_{mi}|\theta_{mi} &\sim f(\theta_{mi}). \end{aligned} \tag{9}$$

We will refer to this construction as a hierarchical Pitman Yor/Dirichlet process (HPY/DP). The use of a gamma dependence between the concentration parameters was first introduced by Williamson et al. (2013) in the context of the hierarchical Dirichlet process.

In Section 5, we investigate the performance of this special case of the hierarchical Pitman-Yor process on a text corpus. We find that it performs nearly as well as the more general model of Equation 4, and out-performs the hierarchical Dirichlet process (Teh et al., 2006). We therefore propose this model for large-scale text data, since it allows scalable parallel inference without significant deterioration in performance.

Theorem 3 extends the auxiliary variable representation of Theorem 2 to the hierarchical model of Equation 9.

Theorem 3 (Auxiliary variable representation for the hierarchical Pitman-Yor process)**.** *We can rewrite the generative process for the hierarchical model of Equation 9 as:*

$$\begin{aligned} \zeta_j &\sim \text{Gamma}(\alpha/P)\,, \\ D_{0j} &\sim \text{PY}(\alpha/P, d, H)\,, \\ \nu_m &\sim \text{Dirichlet}(\zeta_1, \ldots, \zeta_P)\,, \\ D_{mj}|D_{0j} &\sim \text{DP}(\zeta_j, D_{0j})\,, \\ \mu_{mi}|\nu_m &\sim \nu_m \\ \theta_{mi}|\mu_{mi}, D_{m1}, \ldots, D_{mP} &\sim D_{m\mu_{mi}} \\ x_{mi}|\theta_{mi} &\sim f(\theta_{mi})\,, \end{aligned} \tag{10}$$

for $j = 1, \ldots, P$, $m = 1, \ldots, M$, and $i = 1, \ldots, N_m$. The marginal distribution over the x_i remains the same as in Equation 9.

Proof. Let $\gamma := \sum_j \zeta_j$. The normalized vector $\frac{\zeta_1, \ldots, \zeta_P}{\gamma}$ is distributed according to Dirichlet $\left(\frac{\alpha}{P}, \ldots, \frac{\alpha}{P}\right)$, so from Theorem 1 we find that

$$D_0 := \sum_{j=1}^{P} \frac{\zeta_j}{\gamma} D_{0j} \;\sim\; PY(\alpha, d, H).$$

Now, for $m = 1, \ldots, M$ and $j = 1, \ldots, P$, let $\eta_{mj} \sim \text{Gamma}(\zeta_j)$ and $D_{mj} \sim \text{DP}(\zeta_j, D_{0j})$. The normalized vector $(\eta_{m1}, \ldots, \eta_{mP}) / \sum_{j=1}^{P} \eta_{mj}$ is therefore distributed according to Dirichlet$(\zeta_1, \ldots, \zeta_P)$. From Equation 5, we see that

$$D_m := \sum_{j=1}^{P} \eta_{mj} D_{mj} \;\sim\; \text{DP}(\gamma, D_0).$$

□

The representation in Theorem 3 provides the conditional independence structure required to construct a data- and model-parallel inference algorithm.

4 INFERENCE

The auxiliary variable representation introduced in Theorem 2 makes the cluster allocations for data points $\{x_i : \mu_i = j\}$ conditionally independent of the cluster allocations for data points $\{x_i : \mu_i \neq j\}$. A similar conditional independence relationship for the hierarchical model is implied by Theorem 3. We can therefore split the data onto P parallel processors or cores, based on the values of μ_i (or μ_{mi} in the hierarchical case). We will henceforth call μ_i (μ_{mi}) the "processor indicator" for the ith data point (ith data point in the mth group).

The resulting samplers allow both model and data parallelization. Inference in Pitman-Yor mixture models and hierarchical Pitman-Yor processes scales with both the number of data points and the number of clusters. Since each conditionally-independent sub-model only uses a subset of the data points and of the clusters, we are able to obtain significant computational advantage, as we will show empirically in Section 5.

4.1 PARALLEL INFERENCE IN THE PITMAN-YOR PROCESS

We consider first the Pitman-Yor mixture model of Equation 3. Under the auxiliary variable representation of Equation 8, each data point x_i is associated with a processor indicator μ_i and parameter θ_i. We introduce cluster indicator variables z_i, such that $z_i = z_j$ iff $\theta_i = \theta_j$. Provided the base measure H is continuous, all data points associated with a single cluster will have the same processor indicator, meaning that we can assign each cluster to one of the P processors (i.e., all data points in a single cluster are assigned to the same processor). Note that the jth processor will typically be associated with multiple clusters, corresponding to the local Pitman-Yor process D_j. Conditioned on the assignments of the processor indicators μ_i, the data points x_i in Equation 8 depend only on the local Pitman-Yor process D_{μ_i} and the associated parameters.

We can easily marginalize out the D_j and ϕ. Assume that each data point x_i is assigned to a processor $\mu_i \in \{1, \dots, P\}$, and a cluster z_i residing on that processor. We will perform *local* inference on the cluster assignments z_i, and intermittently we will perform *global* inference on the μ_i.

4.1.1 Local inference: Sampling the z_i

Conditioned on the processor assignments, the distribution over cluster assignments z_i is given by

$$P(z_i = k|\{z_j : j \neq i, \mu_j = \mu_i\}, x_i, \text{rest})$$
$$\propto \begin{cases} \frac{n^{\neg i}_{\mu_i,k} - d}{\alpha + n^{\neg i}_{\mu_i,\cdot}} f_k(x_i) & \text{for existing cluster k} \\ \frac{\alpha + Kd}{\alpha + n^{\neg i}_{\mu_i,\cdot}} f^*(x_i) & \text{new cluster k} \end{cases}$$

where $n_{j,k}$ is the number of data points in the kth cluster on processor j, $f_k(x)$ is the likelihood of data point x for the kth cluster, and $f^*(x)$ is the likelihood of data point x under a new cluster.

4.1.2 Global inference: Sampling the μ_i

Under the auxiliary variable scheme, each cluster is associated with a single processor. We jointly resample the processor allocations of all data points within a given cluster, allowing us to move an entire cluster from one processor to another. We use a Metropolis Hastings step with a proposal distribution $Q(k, j_1, j_2)$ that independently assigns cluster k from processor j_1 to processor j_2. We discuss choices of proposal distribution $Q(k, j_1, j_2)$ in Section 4.3. The accept/reject probability is given by $r \cdot \frac{Q(k,j_2,j_1)}{Q(k,j_1,j_2)}$ where r is the likelihood ratio

$$r = \prod_{j=1}^{P} \frac{\Gamma(N^*_j + \alpha/P)}{\Gamma(N_j + \alpha/P)} \frac{(\alpha/P)^{(d;K^*_j - 1)}}{(\alpha/P)^{(d;K_j - 1)}} \frac{(\alpha/P + 1 - d)^{(1;N_j - 1)}}{(\alpha/P + 1 - d)^{(1;N^*_j - 1)}} \prod_{i=1}^{\max(N_j, N^*_j)} [(1-d)^{(1;i-1)}]^{(a^*_{ij} - a_{ij})} \frac{a_{ij}!}{a^*_{ij}!}, \tag{11}$$

where N_j is the number of data points on processor j, a_{ij} is the number of clusters of size i on processor j and

$$(a)^{(b;c)} = \begin{cases} 1 & \text{if } c = 0 \\ a(a+b)\dots(a+(c-1)b) & \text{for } c = 1, 2, \dots \end{cases}$$

A derivation of Equation 11 is given in the supplement. In fact, we can simplify Equation 11 further, since many of the terms in the ratio of factorials will cancel.

The reassignment of clusters can be implemented in a number of different manners. Actually transferring data from one processor to another will lead to bottlenecks, but may be appropriate if the entire data set is too large to be stored in memory on a single machine. If we can store a copy of the dataset on each machine, or we are using multiple cores on a single machine, we can simply transfer updates to lists of which data points belong to which cluster on which machine. We note that the reassignments need not occur at the same time, reducing the bandwidth required.

4.2 PARALLEL INFERENCE IN THE HIERARCHICAL PITMAN-YOR/DIRICHLET PROCESS

Again, we can assign tokens x_{mi} to one of P processors according to μ_{mi}. Conditioned on the processor assignment and the values of ζ_j, the data on each processor is distributed according to an HPY/DP. We instantiate the processor allocations μ_{mi} and the bottom-level DP parameters, plus sufficient representation to perform inference in the processor-specific HPY/DPs. We assume a Chinese restaurant franchise representation (Teh et al., 2006) – each group is represented using a "restaurant"; data points in the lower-level Dirichlet processes are clustered into "tables"; in the upper-level Pitman-Yor process, these "tables" are clustered and each cluster is assigned a "dish".

4.2.1 Local inference: Sampling the table and dish allocations

Conditioned on the processor assignments, we simply have P independent HPY/DPs, and can use any existing inference algorithm for the hierarchical Pitman-Yor process. In our experiments, we used the Chinese restaurant franchise sampling scheme (Teh et al., 2006; Teh, 2006a); other representations could also be used.

4.2.2 Global inference: Sampling the μ_{mi} and the ζ_j

We can represent the ζ_j as $\zeta_j := \gamma\xi_j$, where $\gamma \sim \text{Gamma}(\alpha, 1)$ and $\boldsymbol{\xi} := (\xi_1, \dots, \xi_P) \sim \text{Dirichlet}(\alpha/P, \dots, \alpha/P)$. We sample $\boldsymbol{\xi}$ and the μ_{mi} jointly, and then sample γ, in order to improve the acceptance ratio of our Metropolis Hastings steps.

Again, we want to reallocate whole clusters rather than independently reallocate individual tokens. So, our proposal distribution again assigns cluster k from processor j_1 to processor j_2 with probability $Q(k, j_1, j_2)$. Note that this means that a single data point does not necessarily reside on a single processor – its tokens may be split among multiple processors. We also propose $\boldsymbol{\xi}^* \sim \text{Dirichlet}(\alpha/P, \dots, \alpha/P)$, and accept the resulting state with probability $\min(1, r\frac{Q(k,j_2,j_1)}{Q(k,j_1,j_2)})$, where

$$\begin{aligned} r = &\prod_{j=1}^{P} \frac{(\xi_j^*)^{(T^*_{\cdot j}+\alpha/P)}}{((\xi_j)^{(T_{\cdot j}+\alpha/P)})} \frac{T^*_{\cdot j}!}{T^*_{\cdot j}!} \frac{(\alpha/P)^{(d;U^*_j-1)}}{(\alpha/P)^{(d;U_j-1)}} \\ &\cdot \frac{(\alpha/P+1-d)^{(1;T_{\cdot j}-1)}}{(\alpha/P+1-d)^{(1;T^*_{\cdot j}-1)}} \\ &\cdot \left\{ \prod_{i=1}^{\max(T_{\cdot j},T^*_{\cdot j})} [(1-d)^{(1;i-1)}]^{b^*_{ji}-b_{ji}} \frac{b_{ji}!}{b^*_{ji}!} \right\} \\ &\cdot \prod_{m=1}^{M} \prod_{i=1}^{\max(N_j,N^*_j)} \frac{a_{jmi}!}{a^*_{jmi}!}. \end{aligned} \tag{12}$$

Here, T_{mj} is the total number of occupied tables from the mth restaurant on processor j, U_j is the total number of unique dishes on processor j, a_{jmi} is the total number of tables in restaurant m on processor j with exactly i customers, and b_{ji} is the total number of dishes on processor j served at exactly i tables. Many of the ratios can be simplified further, reducing computational costs. A derivation of Equation 12 is given in the supplement.

As with the sampler described in Section 4.1, we can either transfer the data between machines, or simply update lists of which data points are "active" on each machine. We can resample γ after sampling $\boldsymbol{\xi}$ and the μ_{mi} using a standard Metropolis Hastings step.

4.3 CHOICE OF PROPOSAL DISTRIBUTION

There are many valid choices for the proposal distributions $Q(k, j_1, j_2)$ used to sample the μ_i and μ_{mi} in Sections 4.1 and 4.2. We tried several different proposal distributions and found we obtained good mixing when

$$Q(k, j_1, j_2) = \begin{cases} \frac{1}{P-1} s_{j_1} & \text{if } j_1 \neq j_2 \\ 1 - s_{j_1} & \text{if } j_1 = j_2 \end{cases} \tag{13}$$

where s_{j_1} is the fraction of data in processor j_1. As discussed by Gal and Ghahramani (2013), due to cluster size imbalance inherent to the Pitman-Yor process, we do not expect to see even load balance; however this proposal distribution encourages processors with a larger proportion of the data (higher load) to transfer data to under-used data. Since at any point the fraction of points in any cluster is small it also reduces the number of transfer and hence network load. We will show in Section 5 that this not only gives good performance but also gives good load sharing among multiple processors.

5 EVALUATION

We expect the inference algorithms presented in Section 4 to yield faster inference than a non-parallel implementation. Each processor is locally performing inference in a Pitman-Yor mixture model or a hierarchical Pitman-Yor/Dirichlet process. These models only contain a subset of the total number of clusters and of the total data set. Since inference in these models scales at least linearly (depending on likelihood) with both the number of data points and the number of latent components, we expect them to converge much more quickly than a Pitman-Yor mixture model or hierarchical Pitman-Yor/Dirichlet process on the entire data set. Provided the computational cost of the global steps remains relatively low, and provided the global steps achieve sufficiently fast mixing, we expect to see overall speed-ups. Further, we expect the estimate quality to remain high, since our algorithms do not introduce approximations.

In both cases, we evaluate via comparison with non-parallel implementations of the model. We are unaware of any other parallelizable inference algorithms for the Pitman-Yor process and its extensions.

5.1 PITMAN-YOR MIXTURE OF GAUSSIANS

We first evaluate performance of the parallel sampler for the Pitman-Yor mixture model, described in Section 4.1, using synthetic data. We generated data sets of varying size N and data dimensionality D, to evaluate performance on different sized datasets. Each data set contained $N/2000$ clusters of size 500, $N/4000$ clusters of size 1000, $N/10000$ clusters of size 2500, and $N/20000$ clusters of size 5000, giving $K = 9N/10000$ clusters in total. This was designed to give many smaller clusters and fewer larger clusters. Each cluster parametrized a univariate Gaussian with unit variance and mean sampled according to a Uniform$((-K/2, K/2)^D)$ distribution.

We ran our algorithm on 90% of the resulting data sets using the algorithm described in Section 4.1, on 1,2,4 and 8 cores of a multi-core machine. Each algorithm was initialized by clustering the data into 80 clusters. We performed

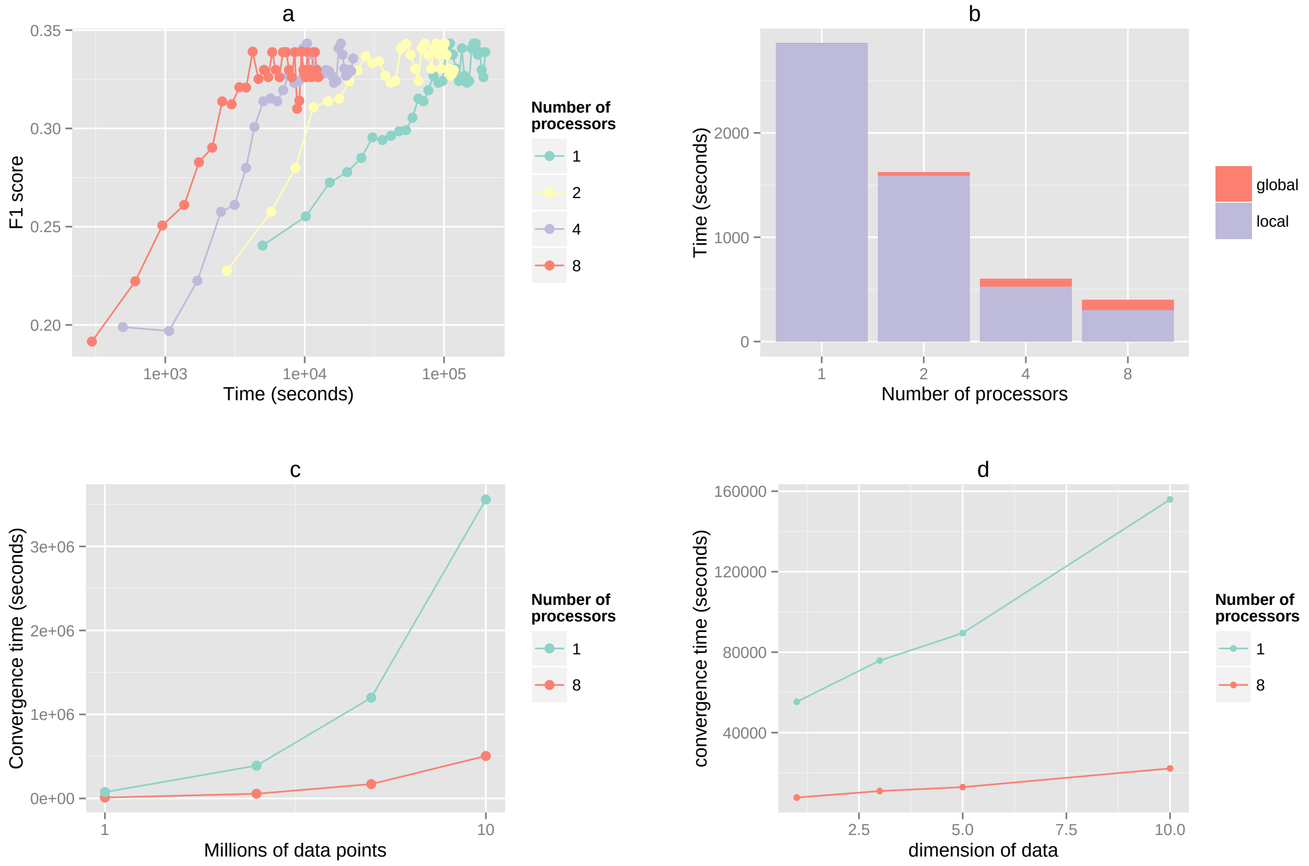

Figure 1: Evaluation on synthetic data modeled using a Pitman-Yor mixture model. a: F1 score vs run time; b: Amount of time spent on global vs local computation; c: Time taken to reach convergence ($< 0.1\%$ change in training set log likelihood) vs number of data points; d: Time taken to reach convergence vs data dimension.

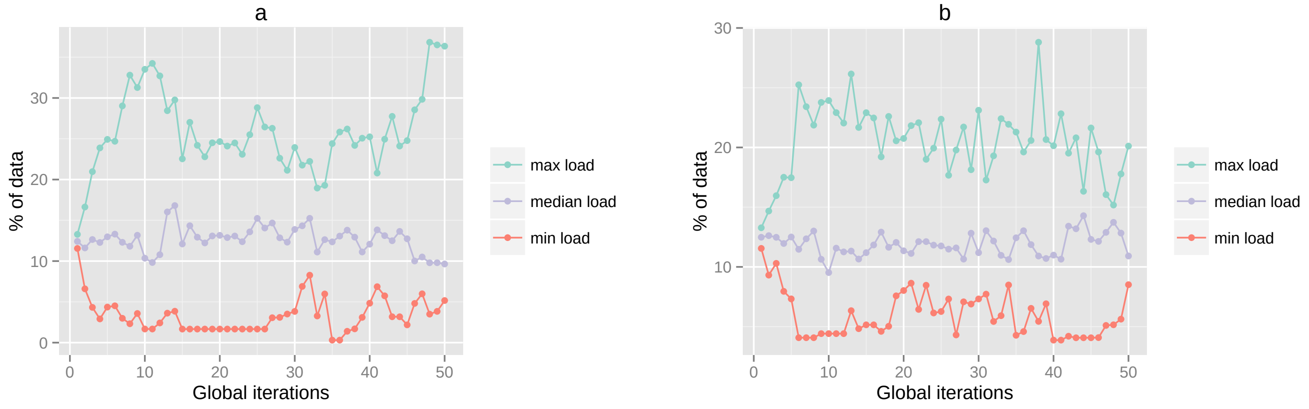

Figure 2: Maximum, median, and minimum loads per global iteration, using (a) a uniform proposal distribution; (b) the proposal distribution given in Equation 13.

100 iterations of the local inference step for each iteration of the global inference step. The concentration parameter α was set to 0.5 and d was set to 0.1; these values were selected via grid search. We evaluated by calculating the test set log likelihood for the remaining 10% of the data.

Figure 1(a) shows how the log likelihood of the test sample varies with time, for one million datapoints with $D = 3$. We see that we get good speedup by increasing the number of processors, while converging to (approximately) the same value for all experiments. Figure 1(b) shows that the amount of time spent on local computation far exceeds that

Data Size	1M	2.5M	5M	10 M
Efficiency	0.864	0.873	0.877	0.879
Dimension	1	3	5	10
Efficiency	0.893	0.864	0.866	0.877
Processors		2	4	8
Efficiency		0.880	0.865	0.864

Table 1: Efficiency with varying data size (with $D = 3$ and $P = 8$), varying data dimension (with $N = 1\text{M}$ $P = 8$) and number of processor (with $N = 1\text{M}$ and $D = 3$).

spent on global steps, explaining why we have a faster per-iteration time. Figures 1(c) and 1(d) show that the decrease in computational speed is apparent at different sizes of data set N and data dimensionality D.

Following Asuncion et al. (2008), we report the efficiency of our model. If a model running on P processor converges in time T_p while the single processor model converges in time T then the efficiency is calculated as $\frac{T}{P \cdot T_p}$. Table 1 shows how efficiency varies if we change the number of data points, the dimensionality of each data point, and the number of processors. An efficiency of 1 would indicate a linear speed-up – using P processors is P times as fast as using one processor. We get efficiency very close to the linear speedup as shown in Table 1.

Next we evaluate how evenly computation is split between the cores. Figure 2 shows the how the data is split between cores over time. Figure 2(b) shows the load distribution obtained using the proposal distribution of Equation 13, and Figure 2(a) shows the load distribution obtained using the uniform distribution used by Williamson et al. (2013). The maximum load governs the amount of time spend performing local computation (which was seen in Figure 1(b) to dominate the computation time). While, as we would expect (Gal and Ghahramani, 2013), we have uneven loads in both cases, we achieve better load balancing using the new proposal.

5.2 HIERARCHICAL PITMAN-YOR/DIRICHLET PROCESS

In this section, we evaluate the sampler described in Section 4.2 on two text data sets:

- NIPS[1]: A collection of 2470 papers from the NIPS conference, collected between 1988 and 2003 which includes 14300 unique words and a total of $3,280,697$ words.
- ENRON[2]: A collection of 39861 emails including 28102 unique words and a total of $6,400,000$ words.

Dataset	HDP	HPY/DP	HPY
NIPS	1706.52	1650.44	1621.34
ENRON	2110.98	2054.85	2018.36

Table 2: Test set perplexity for different models.

Processors	2	4	8
NIPS	0.855	0.805	0.824
ENRON	0.854	0.807	0.817

Table 3: Efficiency of the HPY/DP algorithm with varying number of processors.

In each case, we held out 10% of the data for testing, and evaluated our algorithms using perplexity on the held out test set, as calculated in Asuncion et al. (2008).

We begin by considering how much performance is gaining by restricting the lower-level stochastic processes to be Dirichlet processes, rather than Pitman-Yor processes. Table 2 compares the full hierarchical Pitman-Yor process described in Equation 4 (denoted HPY), the model implemented using our algorithm and described in Equation 9 (denoted HPY/DP), and the hierarchical Dirichlet process Teh et al. (2006) (denoted HDP).

We find that, while the full hierarchical Pitman-Yor process obtains the best perplexity, the HPY/DP model still performs better than the hierarchical Dirichlet process. Since there is not, currently, a scalable inference algorithm for the full hierarchical Pitman-Yor process, we argue that the proposed algorithm and model offer a good trade-off between scalability and performance

Having established the applicability of the model, we consider scalability. Figures 3(a) and 3(b) show how the sample test set perplexity changes with time using 1,2,4 and 8 processors on the NIPS and ENRON data sets, respectively. As with the Pitman-Yor mixture model, we see that increasing the number of processors yields improvements in computation time. Figures 3(c) and 3(d) show that, as before, this occurs because the cost of the local computations decreases as we add more processors, and remains high relative to the cost of the global computations. This is reflected in the efficiencies obtained for different numbers of processors (Table 3), which remain close to one.

6 DISCUSSION AND FUTURE WORK

In this paper, we have presented new auxiliary variable representations for certain cases of the Pitman-Yor process and the hierarchical Pitman-Yor process. These representations allowed us to make use of conditional independencies to develop inference schemes that are both data- and model-parallel.

While this paper provides a significant step forward in

[1] http://ai.stanford.edu/ $\sim$ gal/data.html

[2] https://archive.ics.uci.edu/ml/datasets/Bag+of+Words

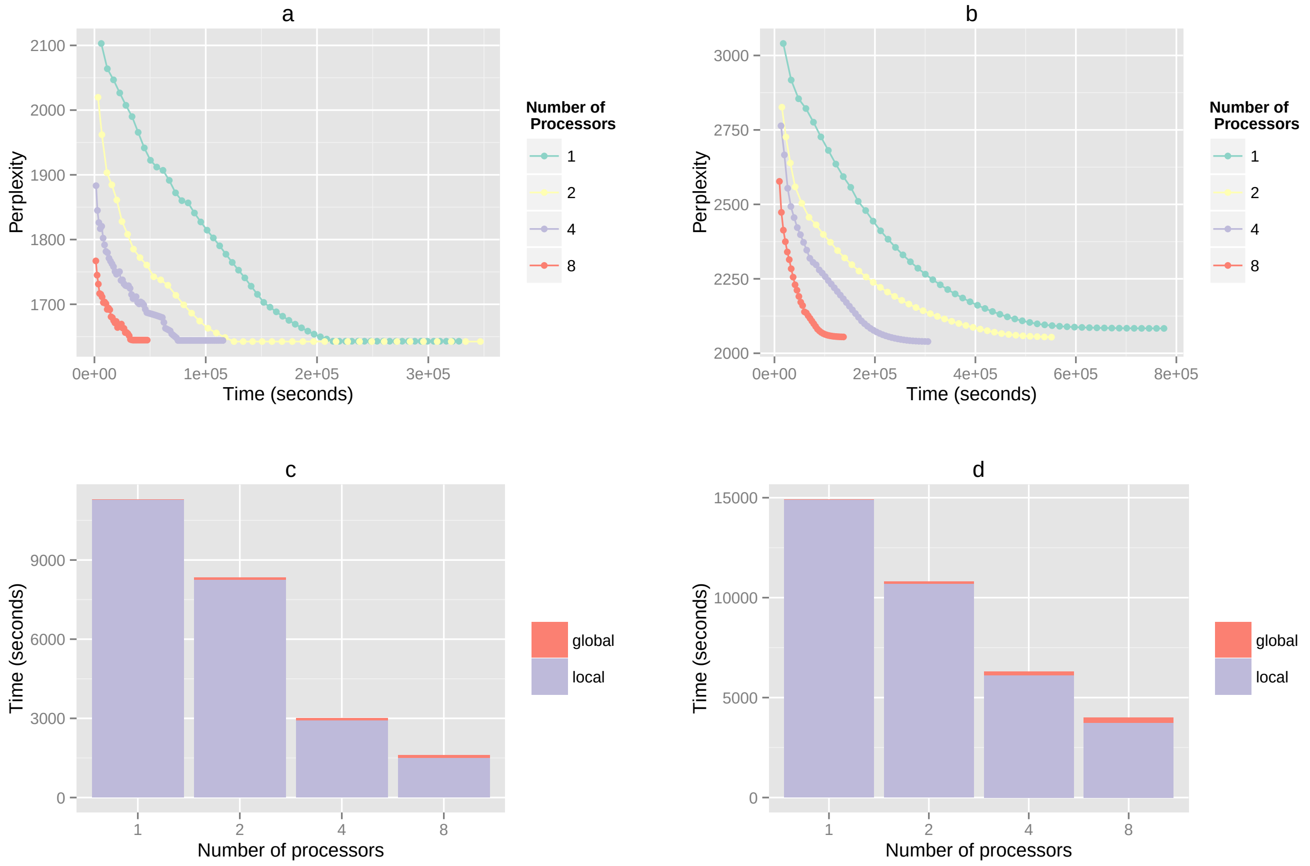

Figure 3: Evaluation on text corpora using the HPY/DP model. a: Test set perplexity vs run time (NIPS); b: Test set perplexity vs run time (ENRON); c: Amount of time spent on global vs local computation (NIPS); d: Amount of time spent on global vs local communication (ENRON).

the development of parallel inference algorithms for the Pitman-Yor process, it does not cover all algorithms of interest. The auxiliary variable representation introduced in Theorem 2 requires a positive concentration parameter. It remains an open question whether there exist alternative representations for $\alpha < 0$ that yield the desired conditional independence structure. Further, our auxiliary variable representation requires a continuous base measure H. While this is typically the case for Pitman-Yor mixture models, it is not the case for the more general hierarchical Pitman-Yor process described in Equation 4. We hope that this work inspires further research into scalable inference for models beyond the Dirichlet process, allowing parallel algorithms for this and other models.

Acknowledgements

This research was supported by *NSF Social IIS1111142, DARPA XDATA FA87501220324* and *NIH GWAS R01GM087694*.

References

Asuncion, A., Smyth, P., and Welling, M. (2008). Asynchronous distributed learning of topic models. In *Advances in Neural Information Processing Systems*.

Blunsom, P. and Cohn, T. (2011). A hierarchical Pitman-Yor process HMM for unsupervised part of speech induction. In *Association for Computational Linguistics: Human Language Technologies (HLT)*.

Chang, J. and Fisher III, J. W. (2013). Parallel sampling of DP mixture models using sub-clusters splits. In *Advances in Neural Information Processing Systems*.

Doshi-Velez, F., Knowles, D., Mohemed, S., and Ghahramani, Z. (2009). Large scale nonparametric Bayesian inference: Data parallelisation in the Indian buffet process. In *Advances in Neural Information Processing Systems*.

Ferguson, T. S. (1973). A Bayesian analysis of some nonparametric problems. *Annals of Statistics*, 1(2):209–230.

Gal, Y. and Ghahramani, Z. (2013). Pitfalls in the use of parallel inference for the Dirichlet process. In *Workshop in Big Learning, NIPS*.

Goldwater, S., Griffiths, T., and Johnson, M. (2006). Interpolating between types and tokens by estimating power-law generators. In *Advances in Neural Information Processing Systems*.

Ishwaran, H. and James, L. (2001). Gibbs sampling methods for stick-breaking priors. *Journal of the American Statistal Association*, 96(453):161–173.

Lovell, D., Adams, R., and Mansinghka, V. (2012). Parallel Markov chain Monte Carlo for Dirichlet process mixtures. In *Workshop on Big Learning, NIPS*.

Neal, R. (1998). Markov chain sampling methods for Dirichlet process mixture models. Technical Report 9815, Dept. of Statistics, University of Toronto.

Perman, M., Pitman, J., and Yor, M. (1992). Size-biased sampling of Poisson point processes and excursions. *Probability Theory and Related Fields*, 92(1):21–39.

Pitman, J. and Yor, M. (1997). The two-parameter Poisson-Dirichlet distribtion derived from a stable subordinator. *Annals of Probability*, 25(2):531–1010.

Teh, Y.-W. (2006a). A Bayesian interpretation of interpolated Keyser-Ney. Technical Report TAR2/06, National University of Singapore.

Teh, Y.-W. (2006b). A hierarchical Bayesian language model based on Pitman-Yor processes. In *ACL*.

Teh, Y.-W., Jordan, M., Beal, M., and Blei, D. (2006). Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.

Williamson, S., Dubey, A., and Xing, E. (2013). Parallel Markov chain Monte Carlo for nonparametric mixture models. In *International Conference on Machine Learning*.

Wood, F., Archambeau, C., Gasthaus, J., James, L., and Teh, Y.-W. (2009). A stochastic memoizer for sequence data. In *International Conference on Machine Learning*.

Zipf, G. (1935). *The Psychobiology of Language*. Houghton-Mifflin.