Hybrid Parallel Inference for Hierarchical Dirichlet Process

Tsukasa ŌMOTO omoto@cs25.scitec.kobe-u.ac.jp
Koji EGUCHI eguchi@port.kobe-u.ac.jp
Kobe University, 1-1 Rokkodaicho, Nada, Kobe 657–8501, Japan

Shotaro TORA[1] tora.shotaro@lab.ntt.co.jp
NTT Software Innovation Center, NTT Corp., Tokyo, Japan

Abstract

The hierarchical Dirichlet process (HDP) can provide a nonparametric prior for a mixture model with grouped data, where mixture components are shared across groups. However, the computational cost is generally very high in terms of both time and space complexity. Therefore, developing a method for fast inference of HDP remains a challenge. In this paper, we assume a symmetric multiprocessing (SMP) cluster, which has been widely used in recent years. To speed up the inference on an SMP cluster, we explore hybrid two-level parallelization of the Chinese restaurant franchise sampling scheme for HDP, especially focusing on the application to topic modeling. The methods we developed, Hybrid-AD-HDP and Hybrid-Diff-AD-HDP, make better use of SMP clusters, resulting in faster HDP inference. While the conventional parallel algorithms with a full message-passing interface does not benefit from using SMP clusters due to higher communication costs, the proposed hybrid parallel algorithms have lower communication costs and make better use of the computational resources.

1. Introduction

Topic modeling is one of the approaches to analyzing grouped data, such as words in documents. Topic models (a.k.a. mixed membership models) are based on the idea that each group can be represented as a mixture model, where mixture components called *topics* are shared across groups. Latent Dirichlet allocation (LDA) (Blei et al., 2003) is a well known topic model. In a scenario where the number of topics is unknown, the hierarchical Dirichlet process (HDP) (Teh et al., 2006) can provide a prior for a topic model such as LDA.

However, inference of the unknown HDP parameters remains a significant challenge in terms of computation time and memory requirements. Fast inference for HDP via parallelization was developed for this purpose (Newman et al., 2009; Asuncion et al., 2008). We assume in this paper a symmetric multiprocessing (SMP) cluster, which has been widely used in recent years, and explore how to achieve hybrid two-level parallelization for HDP inference on an SMP cluster. We demonstrate through experiments using an SMP cluster that the proposed hybrid parallel algorithms increase inference speed substantially while maintaining inference accuracy, compared to the conventional parallel algorithms with a full message-passing interface (MPI).

2. Related Work

In this section, we briefly introduce HDP and the Chinese restaurant franchise (CRF) sampling scheme. We then review prior studies on distributed inference methods for HDP.

2.1. Hierarchical Dirichlet Process

HDP is a non-parametric Bayesian approach developed by Teh et al. (Teh et al., 2006). It is a hierarchi-

Presented at the International Conference on Machine Learning (ICML) workshop on *Inferning: Interactions between Inference and Learning*, Atlanta, Georgia, USA, 2013. Copyright 2013 by the author(s).

[1] This work was done when he was a graduate student at Kobe University, Japan.

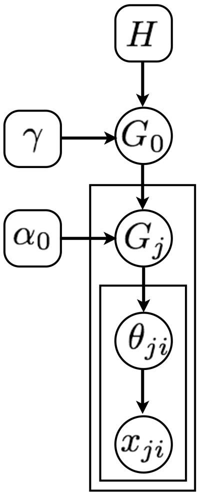

Figure 1. Graphical model of HDP.

cal extension of the Dirichlet process (DP) (Ferguson, 1973). HDP's generative process is represented as

$$G_0|\gamma, H \sim DP(\gamma, H) \quad (1)$$
$$G_j|\alpha_0, G_0 \sim DP(\alpha_0, G_0) \quad (2)$$
$$\theta_j|G_j \sim G_j \quad (3)$$
$$x_{ji}|\theta_j \sim F(\theta_{ji}), \quad (4)$$

where H is a base distribution, and both α_0 and γ are hyperparameters. $DP(\cdot)$ indicates drawing a sample from DP using the parameters in parentheses. Figure 1 shows a graphical model representation of HDP.

HDP can be used as a prior for a mixture model with grouped data (such as words in documents), where mixture components or topics are shared across groups. When HDP is used as a prior for a standard topic model, LDA (Blei et al., 2003), H and F can be expressed as

$$H = Dir(\beta) \quad (5)$$
$$F = Mult(\theta), \quad (6)$$

which is called HDP-LDA.

2.2. Chinese Restaurant Franchise Scheme

The Chinese restaurant franchise (CRF) inference scheme is widely used for HDP (Teh et al., 2006). While other inference schemes can be used for HDP, we use CRF here because it is relatively accurate and intuitively understandable.

CRF naturally extends the Chinese restaurant process (CRP) (Teh et al., 2006) to represent dishes shared across multiple restaurants. In topic models, restaurants, dishes, and customers respectively represent groups (e.g., documents), topics, and data points (e.g., words). Figure 2 depicts this metaphor, and Table 1 lists the notation used.

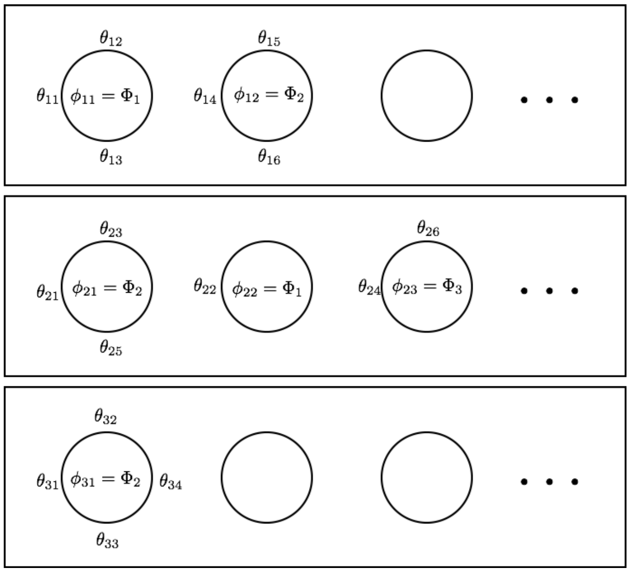

Figure 2. Chinese restaurant franchise metaphor.

The CRF is used to construct HDP as follows (Teh et al., 2006; Wang & Blei, 2012).

Sampling t_{ji}: A table at which the i-th customer sits in the j-th restaurant is drawn in accordance with

$$p(t_{ji} = t|\boldsymbol{t}^{-ji}, \boldsymbol{k}) \propto \begin{cases} n_{jt\cdot} & \text{if } t \text{ is previously used.} \\ \alpha_0 & \text{if } t = t^{new} \end{cases} \quad (7)$$

Sampling k_{jt}: A dish on table t in the j-th restaurant is drawn in accordance with

$$p(k_{jt} = k|\boldsymbol{t}, \boldsymbol{k}^{-jt}) \propto \begin{cases} m_{\cdot k} & \text{if } k \text{ is previously used.} \\ \gamma & \text{if } k = k^{new} \end{cases} \quad (8)$$

Sampling x_{ji}: Finally, the customers are drawn in accordance with

$$p(\boldsymbol{x}|\boldsymbol{t}, \boldsymbol{k}) = \prod_k f_k(\{x_{ji} : k_{ji} = k\}) \quad (9)$$

$$f_k(\{x_{ji} : k_{ji} = k\}) = \frac{\Gamma(V\beta)}{\Gamma(n_{\cdot\cdot k} + V\beta)} \frac{\prod_v \Gamma(n^v_{\cdot\cdot k} + \beta)}{\Gamma(\beta)} \quad (10)$$

where V indicates the size of the vocabulary, β indicates a Dirichlet hyperparameter, and $n^v_{\cdot\cdot k}$ indicates the frequency that customer v has dish k in any restaurant. In the context of topic models, $n^v_{\cdot\cdot k}$ means the frequency with which vocabulary v was assigned to topic k in any document.

Table 1. Notation.

Notation	Description
Φ_k	dish k on global menu (which is shared across all restaurants)
θ_{ji}	dish that customer i has in restaurant j
ϕ_{jt}	dish served at table t in restaurant j
t_{ji}	index of table at which customer i sits in restaurant j
k_{jt}	index of dish served at table t in restaurant j
x_{ji}	index of customer i who sits in restaurant j
n_{jtk}	number of customers having dish k at table t in restaurant j
$n_{jt\cdot}$	number of customers who sit at table t in restaurant j
$n_{\cdot\cdot k}$	number of customers who have dish k in any restaurant
m_{jk}	number of tables on which dish k is served in restaurant j
$m_{\cdot k}$	number of tables on which dish k is served in any restaurant

2.3. Distributed Inference Algorithms for HDP

Newman et al. developed an approximate (synchronous) distributed inference algorithm for HDP (AD-HDP) (Newman et al., 2009). AD-HDP is based on the hypothesis that dependencies between random variables are weak. In AD-HDP, each thread (or node) p first learns a model with the subset data allocated to the thread and then sends the resulting count n_{kvp} to the master thread, which computes n_{kv} using n_{kvp} of all p. Here n_{kv} is the same as $n^{v}_{\cdot\cdot k}$ in Eq.(10). AD-HDP generally produces more accurate perplexity than non-parallel HDP.

Asuncion et al. developed an asynchronous distributed inference algorithm for HDP (Async-HDP), assuming a heterogeneous computing environment (Asuncion et al., 2008). In Async-HDP, each node p first learns a model with the subset data allocated to the node. Then, node p exchanges the resulting count n_{kvp} with another randomly selected node q. Next, n_{kvp} is integrated in q's belief of the counts of all the other processors with which node q has already communicated. As mentioned previously, Async-HDP is designed for a heterogeneous computing environment, which is not our focus in this paper, and therefore, we extend the idea of AD-HDP for SMP clusters.

3. Hybrid Parallel Inference for HDP

Tora et al. developed a hybrid parallel inference approach to LDA that uses a MPI/OpenMP scheme on SMP clusters (Tora & Eguchi, 2011). Here we explore the use of this approach to HDP, especially to HDP-LDA, which is a more complex problem than that of LDA. We developed two hybrid parallel inference algorithms, Hybrid-AD-HDP and Hybrid-Diff-AD-HDP, as extensions of the AD-HDP. Our hybrid algrithms use MPI only to communicate with each node, and multi-threading is used for parallelization within each node.

3.1. Hybrid-AD-HDP

The Hybrid-AD-HDP algorithm is a hybrid parallel inference algorithm based on AD-HDP (Newman et al., 2009). It applies the AD-HDP algorithm to both parallelization within each node and synchronization across nodes, while the original AD-HDP uses an MPI scheme to communicate directly with each processor core.

Algorithm 1 shows the steps in the Hybrid-AD-HDP algorithm. The master node distributes global model parameters to each node, and the nodes then begin to learn the model parameters using the allocated subset data, parallelized by multi-threading based on AD-HDP within the node. The master node then collects the resulting local model parameters from the nodes and computes the difference in those local model parameters from the previous global model parameters to update the global model parameters. This procedure is repeated, and the global model parameters are updated until convergence.

3.2. Hybrid-Diff-AD-HDP

The difference-based Hybrid-AD-HDP (Hybrid-Diff-AD-HDP) algorithm is a modification of the Hybrid-AD-HDP algorithm. Let us first describe the difference-based AD-HDP (Diff-AD-HDP) algorithm: our modification of AD-HDP for robust inference. In Diff-AD-HDP, each thread p first learns a model with the subset data allocated to the thread and then sends the resulting *difference* count Δn_{kvp} to the master thread, which sums up Δn_{kvp} over all p to obtain n_{kv}.

Algorithm 1 Hybrid-AD-HDP

```
repeat
  for each node $p$ in parallel do
    run AD-HDP
    report $n_{kvp}$, $n_{jt}$ to master node
  end for
  merge $n_{jt}$
  update $n_{kv} \leftarrow n_{kv} + \sum_p (n_{kvp} - n_{kv})$
  sample $\alpha_0$, $\gamma$
  broadcast $n_{kv}$, $\alpha_0$, $\gamma$
until convergence
```

Algorithm 2 Hybrid-Diff-AD-HDP

```
repeat
  for each node $p$ in parallel do
    run Diff-AD-HDP
    calculate $\hat{n}_{kvp}$ derived from the node
    report $\hat{n}_{kvp}$, $n_{jt}$ to master node
  end for
  merge $n_{jt}$
  update $n_{kv} \leftarrow \sum_p \hat{n}_{kvp}$
  sample $\alpha_0$, $\gamma$
  broadcast $n_{kv}$, $\alpha_0$, $\gamma$
until convergence
```

Note that Δn_{kvp} is the difference count between n_{kv} that was distributed from the master thread and n_{kvp} that was updated from n_{kv} at node p. In Hybrid-Diff-AD-HDP, the manner of communications in Diff-AD-HDP is applied not only to parallelization within each node, but also to synchronization across nodes.

The Hybrid-AD-HDP algorithm has to synchronize after every Gibbs sweep. Otherwise, some estimated models may be inaccurate and some count variables may turn into negative values. The Hybrid-Diff-AD-HDP algorithm avoids such problems. In Hybrid-Diff-AD-HDP, the master node collects from each node the difference in the local model parameters from the previous global model parameters rather than collecting the local model parameters themselves. The master node then sums up the differences over all nodes to obtain the global model parameters. Algorithm 2 shows the steps in the Hybrid-Diff-AD-HDP algorithm.

4. Experiments

In this paper, we used two data sets: KOS blog entries and NIPS full papers.[1] The statistics of these data sets are shown in Table 2.

Table 2. Dataset statistics.

	KOS	NIPS
Number of documents (D)	3,430	1,500
Size of vocabulary (V)	6,906	12,419
Number of words (N)	467,714	1,932,365

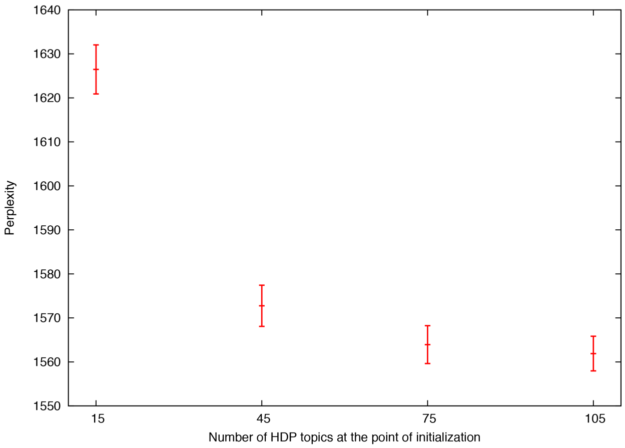

Figure 3. Effect of number of topics that are set at time of initialization using KOS dataset. Results were averaged over 10 runs; error bars represent one standard error.

We evaluated the estimated models by using 10-fold cross-validation. Here we split both datasets into a training set and a test set by assigning 10% of the words in each document to the test set, in accordance with Teh et al. (Teh et al., 2007), and repeated this procedure 10 times. We used (test-set) perplexity as the evaluation metric:

$$\exp\left\{-\frac{1}{N}\log p(\boldsymbol{w}|Training\ set)\right\}, \qquad (11)$$

where $\boldsymbol{w}$ indicates a test set, and N indicates the total number of words in the test set.

4.1. Initialization

Preliminary experiments revealed the effects of the two initialization methods:

(1) Start with a predefined number of topics and randomly assign a topic to each word as the initialization of collapsed Gibbs sampling for LDA (Griffiths & Steyvers, 2004).

(2) Initialize in accordance with the CRF generative process.

We set the hyperparameters in accordance with Teh (Teh et al., 2006): $\alpha = 1/K$ and $\beta = 0.5$ for LDA and $\alpha_0 = E[Gamma(1, 1)] = 1$, $\gamma =$

[1] These data sets are available at http://archive.ics.uci.edu/ml/datasets/Bag+of+Words

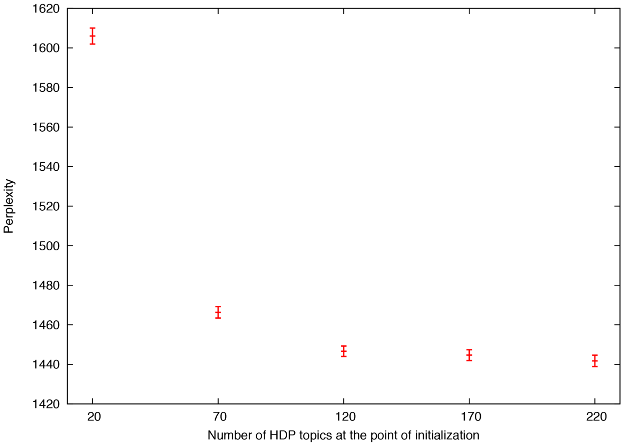

Figure 4. Effect of number of topics that are set at time of initialization using NIPS dataset. Results were averaged over 10 runs; error bars represent one standard error.

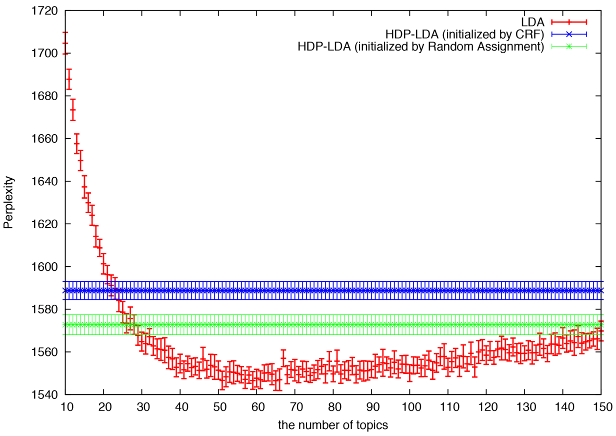

Figure 5. Perplexity of HDP for two initialization methods and LDA using KOS dataset ($K = 45$). Number of topics for LDA varied between 10 and 150. Results were averaged over 10 runs; error bars represent one standard error.

$E[Gamma(1, 0.1)] = 10$, and $\beta = 0.5$ for HDP-LDA. Each Gamma distribution was specified by a shape parameter and a rate parameter. We updated the hyperparameters for HDP-LDA after each Gibbs sweep (Escobar & West, 1995).

Figures 3 and 5 show that, for KOS, both initialization methods did not work as well as the best performance for LDA (i.e., $perplexity = 1550$ at $K = 55$ as shown in Figure 5). This is probably because the total number of words was small compared with the number of documents for KOS. The perplexity with initialization method (1) was slightly better than that with initialization method (2).

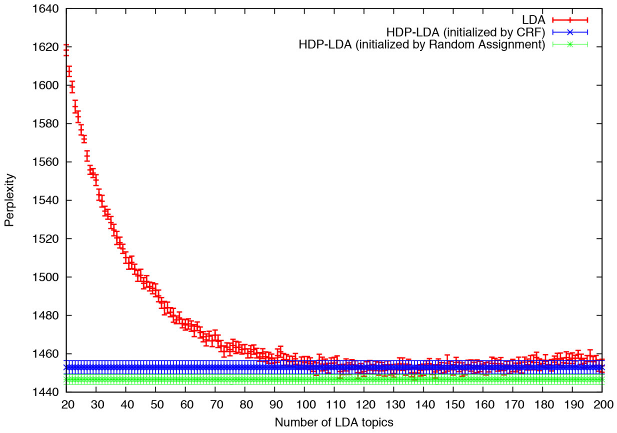

Figure 6. Perplexity of HDP for two initialization methods and LDA using NIPS dataset ($K = 120$). Number of topics for LDA varied between 20 and 200. Results were averaged over 10 runs; error bars represent one standard error.

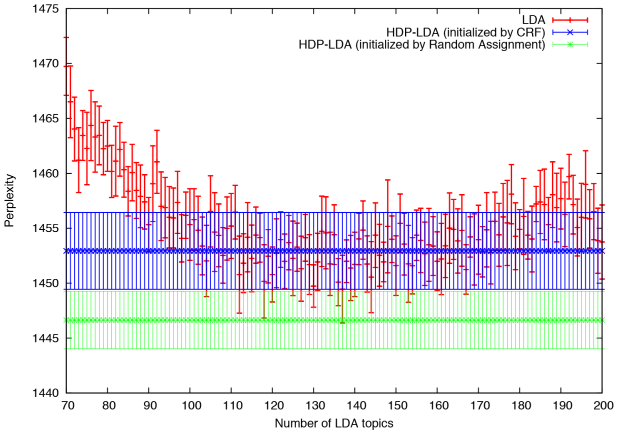

Figure 7. Perplexity of HDP for two initialization methods and LDA using NIPS dataset ($K = 120$), in a finer scale. Number of topics for LDA varied between 70 and 200. Results were averaged over 10 runs; error bars represent one standard error.

Figures 4, 6, and 7 show that, for NIPS, initialization method (1) with $K = 120$, 170, or 220 and initialization method (2) performed as well as or even better than the best performance of LDA (i.e., the perplexity is 1450 at $K = 130$ as shown in Figures 6 and 7). However, initialization method (1) with $K = 20$ or 70 did not work well because fewer topics were learned than with $K = 120$, 170, or 220. The perplexity with (1) was slightly better than that with (2), as with KOS.

The convergence speed with NIPS is shown in Figure 8. The convergence speed with initialization method (2) was comparable to that of LDA, and convergence with initialization method (1) was the fastest. However, initialization method (1) used much more memory than initialization method (2). This indicates that the number of tables was learned more efficiently with (2). We thus used initialization method (2) for our scalability

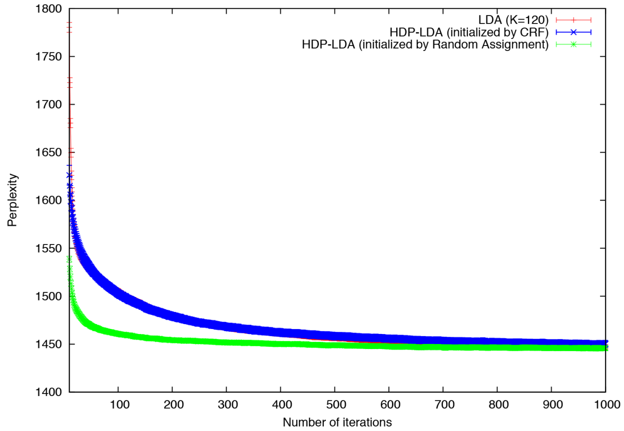

Figure 8. Convergence speed of HDP for two initialization methods ($K = 120$) and LDA. Results were averaged over 10 runs; error bars represent one standard error.

Table 3. Experimental environment.

CPU	Speed (GHz)	Sockets	Mem	Network
Xeon E5410	2.33	2	32 GB	10 GbE

GCC	Open MPI	Boost
4.7.2	1.6.3	1.52

experiment.

4.2. Scalability

We experimentally measured the speed-up rate with NIPS dataset for our hybrid parallel inference algorithms using the experimental environment, including toolchain versions, summarized in Table 3. At that time, the test-set perplexity of the hybrid parallel algorithms was almost the same as that of the parallel algorithm with MPI-HDP algorithm, which was a full MPI implementation based on AD-HDP.

Figure 9 clearly shows that the Hybrid-AD-HDP and Hybrid-Diff-AD-HDP algorithms learned topic models much faster than the MPI-HDP algorithm. MPI-HDP did not achieve speed-up under conditions exceeding ‘4(32)’ (4 nodes with 32 processor cores) because its communication and synchronization costs were larger than the speed-up due to parallelization somewhere between ‘3(24)’ and ‘4(32).’ This did not happen with either hybrid parallel inference algorithm, and speed-up was observed until ‘6(48).’ The speed-up rate decreased after ‘7(56)’ probably because the data set was small. Better performance should be obtained with the hybrid algorithms if larger data sets are used.

As shown in Figure 9, the performances of the two hybrid parallel inference algorithms were comparable. While Hybrid-AD-HDP has to synchronize with all nodes at every Gibbs sweep, Hybrid-Diff-AD-HDP does not. This means that Hybrid-Diff-AD-HDP has room for further speed-up.

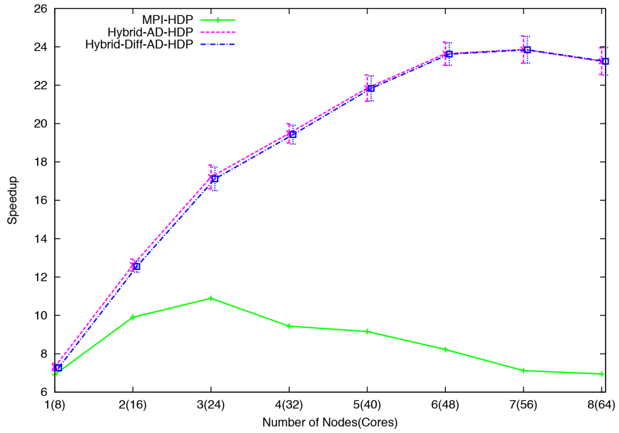

Figure 9. Speedup rate with Hybrid-AD-HDP, Hybrid-Diff-AD-HDP, and MPI-HDP

5. Conclusions

We developed two different hybrid two-level parallel algorithms for HDP, Hybrid-AD-HDP and Hybrid-Diff-AD-HDP, that make better use of SMP clusters. We demonstrated that initialization in accordance with the CRF generative process achieves good cost performance in terms of model accuracy and memory usage. We then showed that the conventional parallel algorithm with full MPI does not benefit from using SMP clusters due to higher communication costs. In contrast, our hybrid parallel algorithms cut communication costs and make better use of the computational resources.

Future work includes developing algorithms for use under more challenging network bandwidth conditions. It also includes evaluating the effectiveness of Hybrid-Diff-AD-HDP as an approach to solving the problem inherent in non-approximate parallelization methods like that of Williamson et al. (Williamson et al., 2012); i.e., while they can learn exact models, they incur a certain amount of communication costs when running on SMP clusters.

References

Asuncion, Arthur, Smyth, Padhraic, and Welling, Max. Asynchronous distributed learning of topic models. *Advances in Neural Information Processing Systems*, 21: 81–88, 2008.

Blei, D.M., Ng, A.Y., and Jordan, M.I. Latent dirichlet allocation. *the Journal of machine Learning research*, 3: 993–1022, 2003.

Escobar, M.D. and West, M. Bayesian density estimation and inference using mixtures. *Journal of the american statistical association*, 90(430):577–588, 1995.

Ferguson, Thomas S. A bayesian analysis of some nonparametric problems. *The annals of statistics*, pp. 209–230, 1973.

Griffiths, Thomas L. and Steyvers, Mark. Finding scientific topics. *Proceedings of the National Academy of Sciences of the United States of America*, 101:5228–5235, 2004.

Newman, D., Asuncion, A., Smyth, P., and Welling, M. Distributed algorithms for topic models. *The Journal of Machine Learning Research*, 10:1801–1828, 2009.

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

Teh, Y.W., Newman, D., and Welling, M. A collapsed variational bayesian inference algorithm for latent dirichlet allocation. *Advances in neural information processing systems*, 19:1353, 2007.

Tora, S. and Eguchi, K. Mpi/openmp hybrid parallel inference for latent dirichlet allocation. In *Proceedings of the Third Workshop on Large Scale Data Mining: Theory and Applications*, pp. 5. ACM, 2011.

Wang, C. and Blei, D.M. A split-merge mcmc algorithm for the hierarchical dirichlet process. *arXiv preprint arXiv:1201.1657*, 2012.

Williamson, S.A., Dubey, A., and Xing, E.P. Exact and efficient parallel inference for nonparametric mixture models. *arXiv preprint arXiv:1211.7120*, 2012.