LETTER *Special Section on Data Engineering and Information Management*

Hybrid Parallel Inference for Hierarchical Dirichlet Processes*

Tsukasa OMOTO[†], ***Nonmember***, **Koji EGUCHI**[†a)], ***Member***, ***and*** **Shotaro TORA**[††], ***Nonmember***

SUMMARY 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.

key words: *hierarchical dirichlet process, topic models, parallelization*

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) [1] is a well known topic model. In a scenario where the number of topics is unknown, the hierarchical Dirichlet process (HDP) [2] 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 [3], [4]. 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).

Manuscript received July 5, 2013.
Manuscript revised October 29, 2013.

[†]The authors are with the Graduate School of System Informatics, Kobe University, Kobe-shi, 657–8501 Japan.
[††]The author is with NTT Software Innovation Center, NTT Corporation, Musashino-shi, 180–8585 Japan.

*This work was done when Tora was a graduate student at Kobe University, Japan.

a) E-mail: eguchi@port.kobe-u.ac.jp

DOI: 10.1587/transinf.E97.D.815

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. [2]. It is a hierarchical extension of the Dirichlet process (DP) [5]. 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 [1], H and F can be expressed as

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

which is called HDP-LDA.

2.2 Chinese Restaurant Franchise Scheme

The Chinese restaurant franchise (CRF) inference scheme is widely used for HDP [2]. While other inference schemes

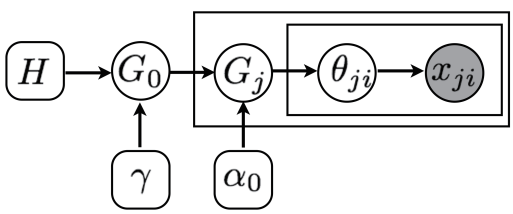

Fig. 1 Graphical model of HDP.

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

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) [2] 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). Table 1 lists the notation used. The CRF is used to construct HDP as follows [2], [6].

(1) Sampling t_{ji}:

A table at which the i-th customer sits in the j-th restaurant is drawn in accordance with

$$p(t_{ji} = t|\boldsymbol{t}^{-ji}, \boldsymbol{k}) \propto \begin{cases} n_{jt\cdot} & \text{if } t \text{ is previously used.} \\ \alpha_0 & \text{if } t = t^{new} \end{cases} \quad (6)$$

(2) Sampling k_{jt}:

A dish on table t in the j-th restaurant is drawn in accordance with

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

(3) Sampling x_{ji}:

Finally, the customers are drawn in accordance with

$$p(\boldsymbol{x}|\boldsymbol{t}, \boldsymbol{k}) = \prod_k f_k(\{x_{ji} : k_{ji} = k\}) \quad (8)$$

$$f_k(\{x_{ji} : k_{ji} = k\}) = \frac{\Gamma(V\beta)}{\Gamma(n_{\cdot\cdot k} + V\beta)} \frac{\prod_v \Gamma(n^v_{\cdot\cdot k} + \beta)}{\Gamma(\beta)} \quad (9)$$

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.

2.3 Distributed Inference Algorithms for HDP

Newman et al. developed an approximate (synchronous) distributed inference algorithm for HDP (AD-HDP) [3]. AD-HDP is based on the hypothesis that dependencies between random variables are weak. In AD-HDP, each thread (or processor core) 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. (9). AD-HDP generally produces comparable or even more accurate perplexity compared to non-parallel HDP.

Asuncion et al. developed an asynchronous distributed inference algorithm for HDP (Async-HDP), assuming a heterogeneous computing environment [4]. 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 [7]. 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 algorithms 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 [3]. 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

[†]There are several ways to merge newly generated topics on each thread. We adopted a simple way that merges new topics based on their topic indices [3].

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 α_0, γ
  broadcast n_{kv}, α_0, γ
until convergence

Algorithm 2 Hybrid-Diff-AD-HDP

repeat
  for each node p in parallel **do**
    run Diff-AD-HDP
    calculate Δn_{kvp} derived from the node
    report Δn_{kvp}, n_{jt} to master node
  end for
  merge n_{jt}
  update $n_{kv} \leftarrow n_{kv} + \sum_p \Delta n_{kvp}$
  sample α_0, γ
  broadcast n_{kv}, α_0, γ
until convergence

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}. 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 three datasets: Enron Emails, NIPS full papers and KOS blog entries [†]. The statistics of these datasets are shown in Table 2. We split each dataset into a training set and a test set by randomly sampling 10% of the words in each document for the test set, while the remaining words are used as the training set for estimating the model parameters [8]. We then repeated this procedure 10 times in the experiments that we will describe in Sects. 4.1 and 4.2.1, and 5 times in the experiments in Sect. 4.2.2. For testing model accuracy, we used (test-set) perplexity as the evaluation metric:

$$\exp\left\{-\frac{1}{N}\log p(\boldsymbol{w}|Training\ set)\right\} \tag{10}$$

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

Table 2 Dataset statistics.

	Enron	NIPS	KOS
Number of documents (D)	39,861	1,500	3,430
Size of vocabulary (V)	28,102	12,419	6,906
Number of words (N)	6,412,172	1,932,365	467,714

4.1 Initialization

Preliminary experiments revealed the effects of the two initialization methods for the Chinese restaurant franchise sampling scheme for HDP-LDA:

(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 [9].
(2) Initialize in accordance with the CRF generative process.

We set the hyperparameters in accordance with Teh [2]: $\alpha = 1/K$ and $\beta = 0.5$ for LDA and $\alpha_0 = \mathbb{E}[Gamma(1,1)] = 1$, $\gamma = \mathbb{E}[Gamma(1,0.1)] = 10$, and $\beta = 0.5$ for HDP-LDA. Note that Each Gamma distribution was specified by a shape parameter and a rate parameter, in this paper. We updated the hyperparameters for HDP-LDA after each Gibbs sweep [10].

Figure 2 shows that, for NIPS dataset, initialization method (1) with the initial number of topics $K = 120$ [††] 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 Fig. 2). Figure 3 shows that, for KOS dataset, both initialization methods did not work as well as the best performance for LDA (i.e., the perplexity is 1550 at $K = 55$ as shown in Fig. 3). This is probably because the total number of words was small compared with the number of documents for KOS dataset. The perplexity with initialization method (1) was slightly better than that with method (2). We found that 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 experiments.

4.2 Scalability

We performed experiments with two datasets to confirm scalability of our hybrid parallel inference algorithms. The first one is NIPS dataset used in the previous experiments. The other is Enron dataset, which is much larger than NIPS dataset as shown in Table 2.

[†]http://archive.ics.uci.edu/ml/datasets/Bag+of+Words

[††]We also observed the same tendency when $K = 170$ and 220.

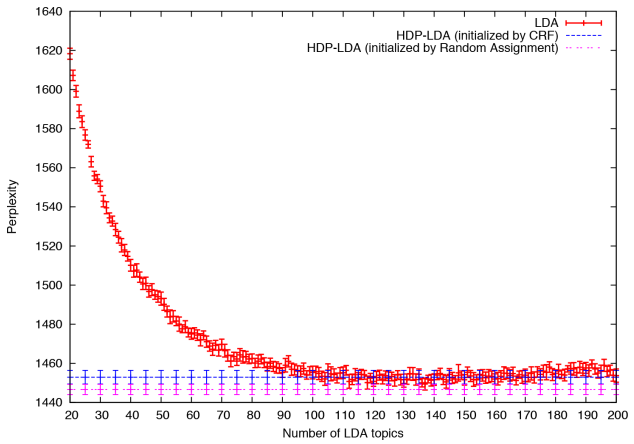

Fig. 2 Perplexity of HDP for two initialization methods and LDA using NIPS dataset. Number of topics for LDA varied between 20 and 200. Results were averaged over 10 runs; error bars represent one standard error.

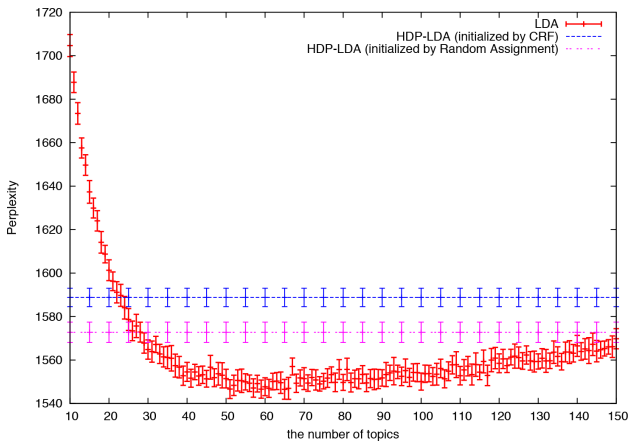

Fig. 3 Perplexity of HDP for two initialization methods and LDA using KOS dataset. Number of topics for LDA varied between 10 and 150. Results were averaged over 10 runs; error bars represent one standard error.

Table 3 Experimental environment for small dataset.

CPU	Clock	Cores	Sockets	Memory	Network
Xeon E5410	2.33 GHz	4	2	32 GB	10 GbE

GCC	Open MPI	Boost
4.7.2	1.6.3	1.52

4.2.1 Small Dataset

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 perplexity of the hybrid parallel algorithms was almost the same as that of the non-parallel algorithm, and also that of the parallel algorithm with MPI-HDP, which was a full MPI implementation based on AD-HDP.

Figure 4 clearly shows that the Hybrid-AD-HDP and Hybrid-Diff-AD-HDP algorithms learned topic models much faster than MPI-HDP. MPI-HDP did not achieve

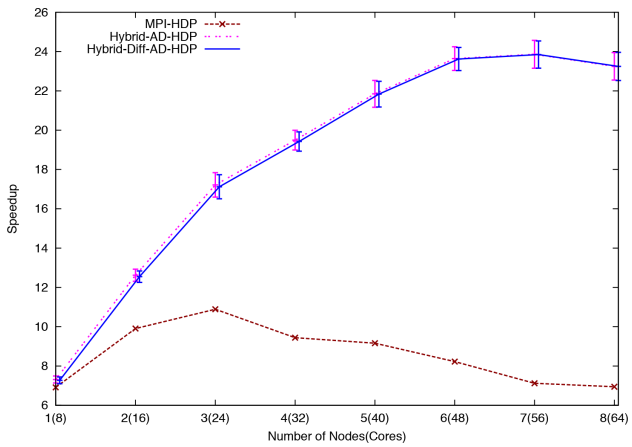

Fig. 4 Speedup rate of Hybrid-AD-HDP, Hybrid-Diff-AD-HDP, and MPI-HDP using NIPS dataset, compared to non-parallel HDP. Results were averaged over 10 runs; error bars represent one sample standard deviation.

Table 4 Experimental environment for large dataset (on Amazon EC2).

CPU	Clock (Turbo)	Cores	Sockets	Memory	Network
Xeon E5-2670	2.6(3.3) GHz	16	1	60.5 or 244 GB †	10 GbE

GCC	Open MPI	Boost
4.7.3	1.6.4	1.54

speed-up under conditions exceeding ‘4(32)’ (4 nodes with 32 processor cores) because its communication and synchronization costs were then larger than the speed-up due to parallelization. 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 dataset was small. Better performance should be obtained with the hybrid algorithms if larger datasets are used.

As shown in Fig. 4, 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.

4.2.2 Large Dataset

We further evaluated scalability to a larger dataset, Enron. For the experiments, we used cloud computing clusters on Amazon EC2. The experimental environment is summarized in Table 4. We only experimented with Hybrid-Diff-AD-HDP, since the performance of Hybrid-AD-HDP was comparable with that of Hybrid-Diff-AD-HDP in the experiments in Sect. 4.2.1. At that time, the perplexity of Hybrid-AD-HDP was almost the same as that of the non-parallel algorithm. MPI-HDP was terminated unsuccessfully due to the lack of memory even when using 16 cores within one node.

Figure 5 supports the idea that Hybrid-Diff-AD-HDP

†We used two instance types referred to as cr1.8xlarge (for the master node) and cc2.8xlarge (for the other nodes). For details, visit http://aws.amazon.com/ec2/.

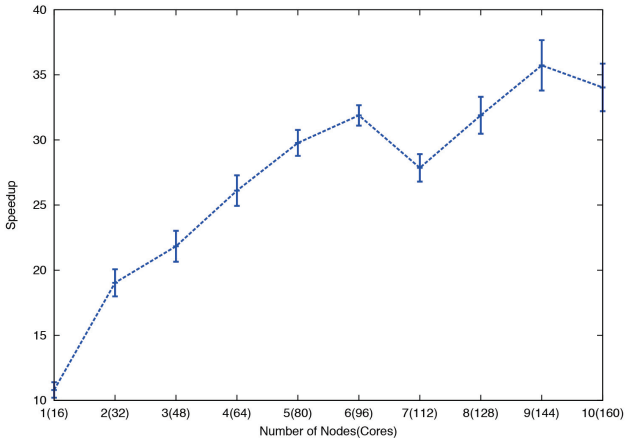

Fig. 5 Speedup rate of Hybrid-Diff-AD-HDP using Enron dataset, compared to non-parallel HDP. Results were averaged over 5 runs; error bars represent one sample standard deviation.

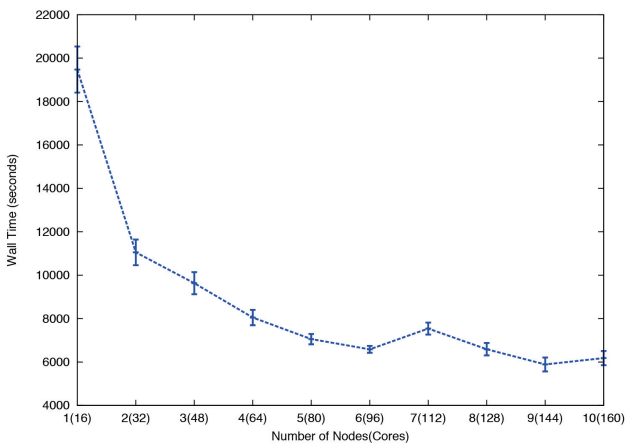

Fig. 6 Wall time of Hybrid-Diff-AD-HDP using Enron dataset. Results were averaged over 5 runs; error bars represent one sample standard deviation.

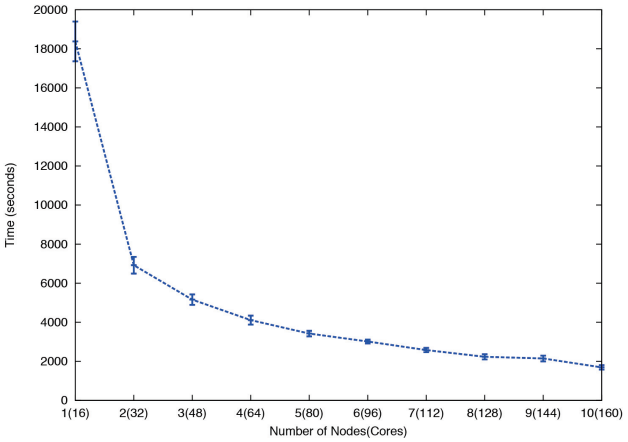

Fig. 7 Time for local inference. Results were averaged over 5 runs; error bars represent one sample standard deviation.

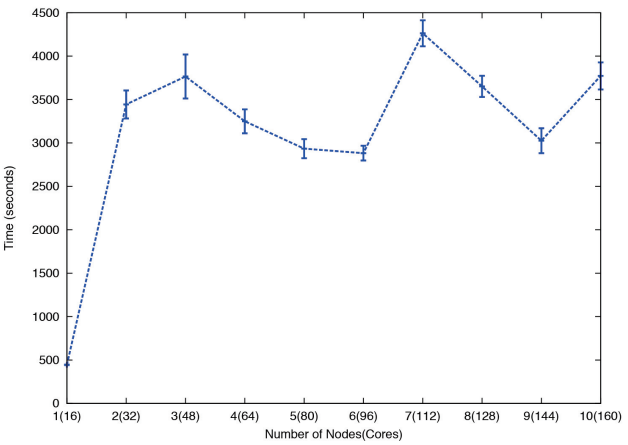

Fig. 8 Time for communications from all nodes to a master node. Results were averaged over 5 runs; error bars represent one sample standard deviation.

works well for larger datasets. From the experiments, we also found that our approach can successfully learn the HDP model over the large dataset that MPI-HDP cannot deal with.

We looked into more details of the cost for running Hybrid-Diff-AD-HDP. Figure 6 shows the total cost (the wall time in seconds), and Figs. 7 and 8 show two major components of the total cost: the cost for local inference within each node [†] (as in Lines 3 to 4 in Algorithm 2) and the cost for communications from all nodes to the master node (as in Line 5 in Algorithm 2), respectively.

Figure 7 demonstrates that the local inference cost decreases as the number of nodes increases, in accordance with our expectation. On the other hand, contrary to our expectation, Fig. 8 shows that the total communication cost is correlated poorly with the number of nodes. For instance, the total cost for the communications with seven nodes is the largest and the total cost with nine nodes is smaller than that. This unpredictable cost is probably due to the cloud environment, where the network load is affected by other users as well.

Even when the total cost is affected by the network load, our approach works efficiently in total, thanks to the cost reduction of the local inference.

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 first 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. For a larger dataset, the conventional method was even terminated unsuccessfully. In contrast, our hybrid par-

[†]We measured the local inference cost with the master node, as an example.

allel 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. [11]; i.e., while they can learn exact models, they incur a certain amount of communication costs when running on SMP clusters.

Acknowledgments

This work was supported in part by the Grant-in-Aid for Scientific Research (#23300039) from JSPS, Japan.

References

[1] D. Blei, A. Ng, and M. Jordan, "Latent Dirichlet allocation," Journal of Machine Learning Research, vol.3, pp.993–1022, 2003.
[2] Y. Teh, M. Jordan, M. Beal, and D. Blei, "Hierarchical Dirichlet processes," Journal of the American Statistical Association, vol.101, no.476, pp.1566–1581, 2006.
[3] D. Newman, A. Asuncion, P. Smyth, and M. Welling, "Distributed algorithms for topic models," Journal of Machine Learning Research, vol.10, pp.1801–1828, 2009.
[4] A. Asuncion, P. Smyth, and M. Welling, "Asynchronous distributed learning of topic models," Advances in Neural Information Processing Systems, vol.21, pp.81–88, 2008.
[5] T.S. Ferguson, "A Bayesian analysis of some nonparametric problems," Annals of Statistics, pp.209–230, 1973.
[6] C. Wang and D. Blei, "A split-merge mcmc algorithm for the hierarchical Dirichlet process," arXiv:1201.1657, 2012.
[7] S. Tora and K. Eguchi, "MPI/OpenMP hybrid parallel inference methods for latent Dirichlet allocation: Approximation and evaluation," IEICE Trans. Inf. & Syst., vol.E96-D, no.5, pp.1006–1015, May 2013.
[8] Y. Teh, D. Newman, and M. Welling, "A collapsed variational Bayesian inference algorithm for latent Dirichlet allocation," Advances in Neural Information Processing Systems, vol.19, p.1353–1360, 2007.
[9] T.L. Griffiths and M. Steyvers, "Finding scientific topics," Proceedings of the National Academy of Sciences of the United States of America, vol.101, pp.5228–5235, 2004.
[10] M. Escobar and M. West, "Bayesian density estimation and inference using mixtures," Journal of the American Statistical Association, vol.90, no.430, pp.577–588, 1995.
[11] S. Williamson, A. Dubey, and E. Xing, "Parallel Markov chain Monte Carlo for nonparametric mixture model," Proceedings of the 30th International Conference on Machine Learning, pp.98–106, 2013.