SAILING LAB

Laboratory for Statistical Artificial InteLligence and INtegrative Genomics

School of Computer Science, Carnegie Mellon University

Large-scale Distributed Dependent Nonparametric Trees

Zhiting Hu[1], Qirong Ho[2], Avinava Dubey[1], Eric Xing[1]

[1]Carnegie Mellon University,

[2]Institute for Infocomm Research, A*STAR

Outline

- ❏ Background
  - ❏ Bayesian nonparametrics
  - ❏ Parallel inference
- ❏ Distributed learning of DNTs
  - ❏ Truncation-free varitional inference
  - ❏ Data- & model-parallelism

Outline

- ❏ Background
  - ❏ Bayesian nonparametrics
  - ❏ Parallel inference
- ❏ Distributed learning of DNTs
  - ❏ Truncation-free varitional inference
  - ❏ Data- & model-parallelism

- Parametric model
  - represents all data using a fixed, finite number of parameters
    e.g., mixture of *K* Gaussians
- Nonparametric model
  - number of parameters can grow with sample size
- Bayesian nonparametric mixture models
  - flexible Bayesian approach to data clustering
  - countably infinite mixture components *a priori*
    - Dirichlet process (DP)
      Dependent nonparametric trees (DNTs) [DHWX'14]

Dirichlet Process (DP)

A random probability measure G drawn from a DP:

$$G = \sum_{i=1}^{\infty} \pi_i \delta_{\theta_i} \qquad \sum_{i=1}^{\infty} \pi_i = 1$$

$$\pi_i = \nu_i \prod_{i'=1}^{i-1} (1 - \nu_{i'}) \qquad \theta_i \sim H$$

$$\nu_i \sim \text{Beta}(1, \alpha) \qquad \pi_1 = \nu_1$$

Dirichlet Process (DP)

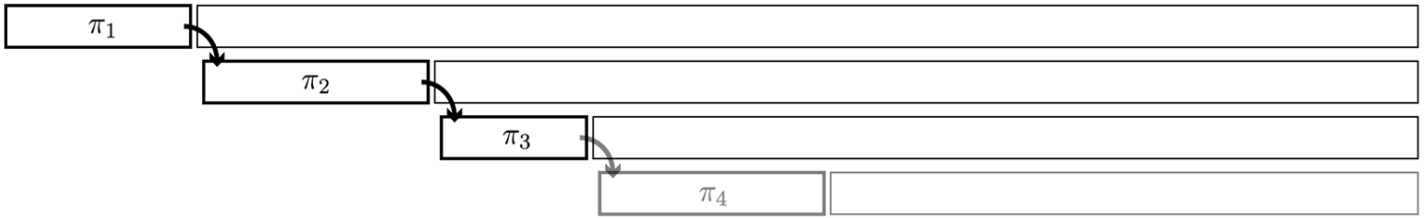
A random probability measure G drawn from a DP:

$$G = \sum_{i=1}^{\infty} \pi_i \delta_{\theta_i} \qquad \sum_{i=1}^{\infty} \pi_i = 1$$

$$\pi_i = \nu_i \prod_{i'=1}^{i-1} (1 - \nu_{i'}) \qquad \theta_i \sim H$$

$$\nu_i \sim \text{Beta}(1, \alpha) \qquad \pi_1 = \nu_1$$

- Take a stick of unit length and break it at random location:

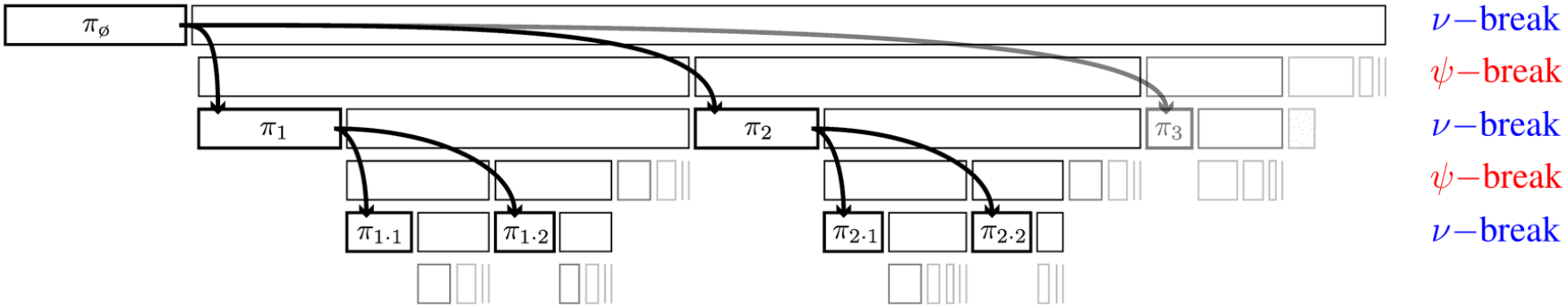


$$\pi_1 = \nu_1$$
$$\pi_2 = \nu_2(1 - \nu_1)$$
$$\pi_3 = \nu_2(1 - \nu_2)(1 - \nu_1)$$

(a) Dirichlet process stick breaking

Tree-structured stick breaking process (TSSBP) [GJA'10]

Interleaves two stick-breaking procedures to possess a hierarchical topology:



(b) Tree-structured stick breaking

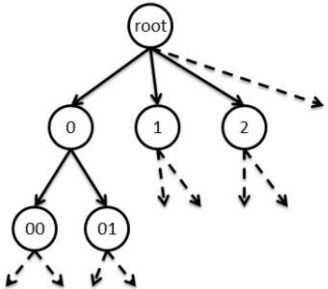
$$\psi_\epsilon \sim \text{Beta}(1,\gamma) \qquad \nu_\epsilon \sim \text{Beta}(1,\alpha)$$

$$\phi_{\epsilon\cdot i} = \psi_{\epsilon\cdot i}\prod_{j=1}^{i-1}(1-\psi_{\epsilon\cdot j}) \qquad \pi_\epsilon = \nu_\epsilon\phi_\epsilon\prod_{\epsilon'\prec\epsilon}(1-\nu_{\epsilon'})\phi_{\epsilon'}$$

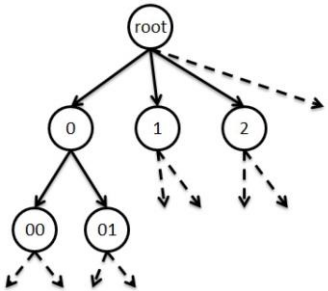
$$\phi_\emptyset = 1 \qquad \pi_\emptyset = \nu_\emptyset$$

$$\sum_\epsilon \pi_\epsilon = 1$$

- TSSBP mixture model: each node ϵ is associated with
  - a mass π_ϵ
    - data live at internal nodes in the tree
  - data-generating parameter θ_ϵ
    - images: Gaussian
    - documents: von-Mises Fisher (vMF)



- TSSBP mixture model: each node ϵ is associated with
  - a mass π_ϵ
    - data can live at internal nodes in the tree
  - data-generating parameter θ_ϵ



- *Dependent* nonparametric trees [DHWX'13]
  - model temporal variations of the infinite trees
  - cluster mass (stick-breaking)

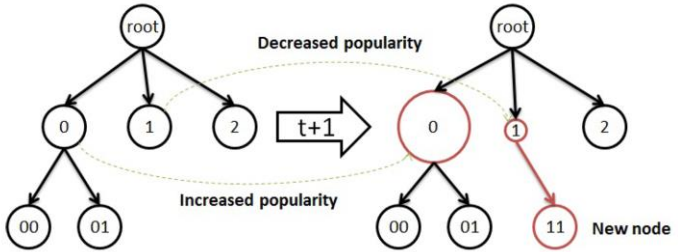
$$\nu_\epsilon^{(t)} \sim \text{Beta}(1 + s^{(t-1)}, \alpha + s^{(t-1)})$$
$$\psi_{\epsilon\cdot i}^{(t)} \sim \text{Beta}(1 + s^{(t-1)}, \gamma + s^{(t-1)})$$

  - θ (e.g. vMF to cluster text)

$$\theta_{\epsilon\cdot i}^{(t)} | \theta_\epsilon^{(t)}, \theta_{\epsilon\cdot i}^{(t-1)} \sim \text{vMF}(\tau_{\epsilon\cdot i}^{(t)}, \rho_{\epsilon\cdot i}^{(t)})$$

$$\tau_{\epsilon\cdot i}^{(t)} = \frac{\kappa_0\kappa_1\theta_\epsilon^{(t)} + \kappa_0\kappa_2\theta_{\epsilon\cdot i}^{(t-1)}}{\rho_{\epsilon\cdot i}^{(t)}}$$

$$\rho_{\epsilon\cdot i}^{(t)} = \kappa_0 \| \kappa_1\theta_\epsilon^{(t)} + \kappa_2\theta_{\epsilon\cdot i}^{(t-1)} \|$$



Parallel Inference

- **Basic idea:** split computation onto multiple machines, perform frequent local and once-in-a-while global computations
  - local step: assigns datum to components
  - global step: updates/synchronizes component parameters

Parallel Inference

- **Basic idea:** split computation onto multiple machines, perform frequent local and once-in-a-while global computations
  - local step: assigns datum to components
  - global step: updates/synchronizes component parameters

- **Challenges:**
  - conditional dependencies makes parallelization difficult
  - model alignment problem
    - a processor can add new component during local steps
    - global step need align components on different processors

Parallel Inference

- Previous BNP problem scales (Table 1)
- Mostly focused on DP/HDP

System	Smyth et al. 2009	Willia-mson et al. 2013	Chang & Fish-er III 2014	Bryant & Sud-derth 2012	Ours
Model	HDP	HDP	HDP	HDP	DNTs
Infer	MCMC	MCMC	MCMC	VI	VI
Sgl-mac parallel	√	√	√	·	√
Mul-mac parallel	√	·	·	·	√
Data size #doc	0.4M	2.5K	0.3M	1.2M	**8.4M**
Model size #topic #param	800 5.6M	80 1.2M	200 20M	600 4.8M	**10K 700M**
Train time	–	7.5hr	28hr	40hr	36hr

Table 1. Comparison of recent BNP training frameworks.

Parallel Inference

- Previous BNP problem scales (Table 1)
- Mostly focused on DP/HDP

This work:

- supports orders-of-magnitude larger datasets and model
- tackles DNTs which is tree-structured
  - additional dependencies
  - costly model alignment

System	Smyth et al. 2009	Williamson et al. 2013	Chang & Fisher III 2014	Bryant & Sudderth 2012	Ours
Model	HDP	HDP	HDP	HDP	DNTs
Infer	MCMC	MCMC	MCMC	VI	VI
Sgl-mac parallel	√	√	√	·	√
Mul-mac parallel	√	·	·	·	√
Data size #doc	0.4M	2.5K	0.3M	1.2M	**8.4M**
Model size #topic #param	800 5.6M	80 1.2M	200 20M	600 4.8M	**10K 700M**
Train time	–	7.5hr	28hr	40hr	36hr

Table 1. Comparison of recent BNP training frameworks.

Outline

- ❏ Background
  - ❏ Bayesian nonparametrics
  - ❏ Parallel inference
- ❏ Distributed learning of DNTs
  - ❏ Truncation-free varitional inference
  - ❏ Data- & model-parallelism

Overview

Algorihthm	System
Variational Inference:	Building on Petuum[1] parameter server:
parameter learning	data parallel
structure updating	model parallel

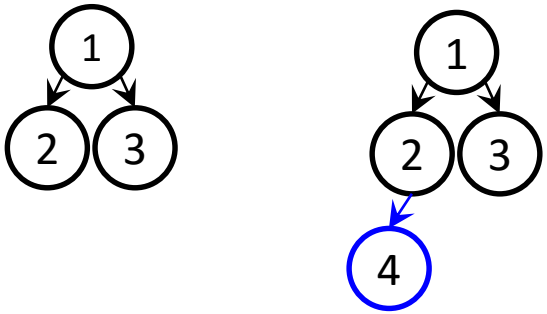
[1]petuum.org

Variational inference (VI)

- Approximates posterior $p(\theta \mid X)$ with a distribution $q(\theta)$
  - $q(\theta)$ assumes additional independencies that are appealing for parallelization

Truncation-free VI

- Parameter learning:
  - assume a truncation level K
  - force $q(z_n = k) = 0$ for $k > K$ [TKW'07; BS'12]
  - we can ignore components indexed $> K$, and learn parameters of components indexed $\leq K$.
- Structure updating:
  - adapt K through split-merge moves
  - explore unbounded tree space



K=3 K=4

Parameter Learning: local parameters

- Update cluster assignment $q(z_n)$ for each datum x_n

$$q(z_n = z) \propto \exp\{\mathbb{E}[\log\ p(z_n = z|\boldsymbol{\nu}, \boldsymbol{\psi})] + \mathbb{E}[\log\ p(x_n|\theta_z)]\}$$

(only need to consider $z\ =\ 1, 2, \ldots, K$)

Parameter Learning: global parameters

- stick-breaking parameters $\{\nu_\epsilon, \psi_\epsilon\}$,
  data-generatng parameters $\{\theta_\epsilon\}$
- Memoized VI [NH'98; HS'13]:
  - maintains sufficient statistics S of the whole dataset
  - pre-partition D into batches, each B has sufficient statistics S_B
  - when visiting B:
    - $S = S - S_B$, computes $S_B^{(new)}$, $S = S + S_B^{(new)}$
  - $\theta^{(l+1)} = f(S)$

- Efficiency: interleaving global and local updates frequently
- Robustness: avoiding learning rates (different from stochastic VI)

Parameter Learning: global parameters

- Sufficient statistics: $S = \{M_\epsilon, s_\epsilon\}_{\epsilon \in T}$

$$M_\epsilon \triangleq \mathbb{E}\left[\sum\nolimits_n z_{n\epsilon}\right] = \sum\nolimits_n q(z_n = \epsilon),$$

$$s_\epsilon \triangleq \mathbb{E}\left[\sum\nolimits_n z_{n\epsilon} x_n\right] = \sum\nolimits_n q(z_n = \epsilon) x_n.$$

- Update global parameters
  - approximations: 1) lower-bounding the Bessel function of vMF distribution; 2) Taylor expansion for non-conjugacy

Structure Updating: birth-merge moves

- Birth: creates a new child of an existing node
  - randomly selects a target node ϵ, $\propto \pi_\epsilon$
  - creates a child node $\epsilon \cdot k$ of ϵ
  - restricted VI: only update parameters of ϵ and $\epsilon \cdot k$

- Merge: merges two siblings, or a child with its parent
  - randomly select a node pair, and merge
  - if merge improves the objective, accept

Structure Updating: birth-merge moves

Birth/merge moves only access to local regions of the whole tree

- Birth: creates a new child of an existing node
  - randomly selects a target node ϵ, $\propto \pi_\epsilon$
  - creates a child node $\epsilon \cdot k$ of ϵ
  - restricted VI: only update parameters of ϵ and $\epsilon \cdot k$

Write access to one node

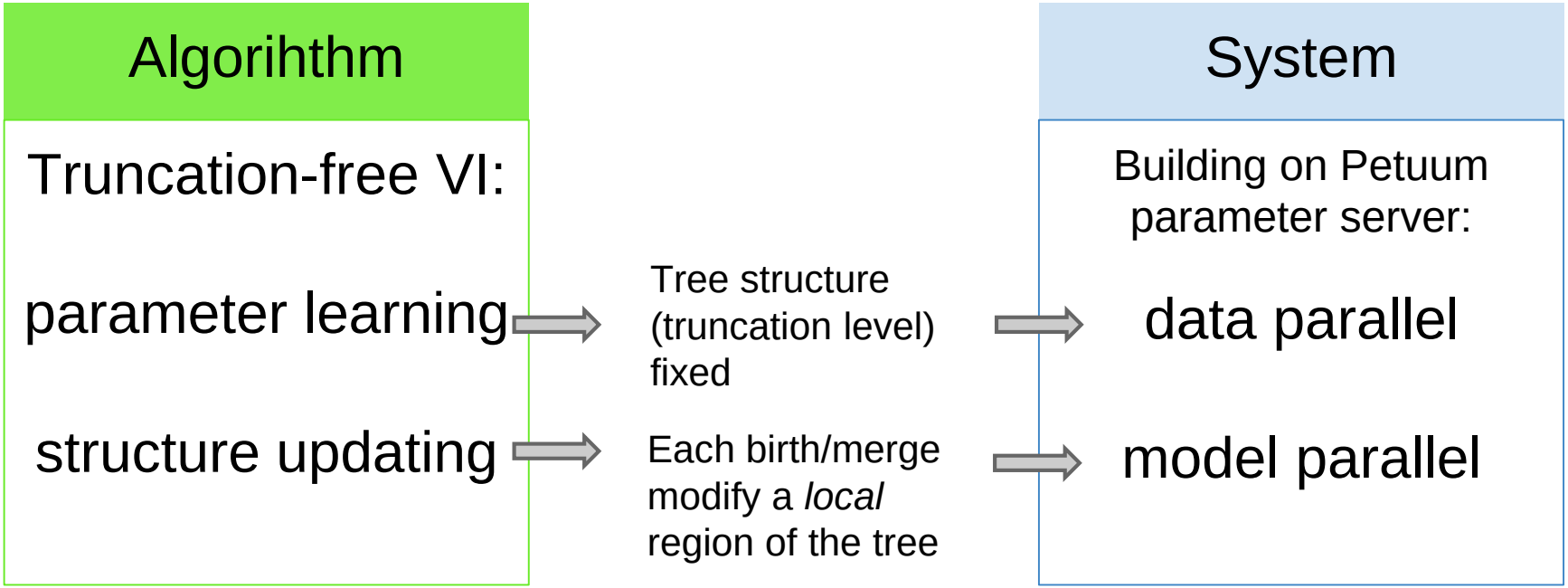
- Merge: merges two siblings, or a child with its parent
  - randomly select a node pair, and merge
  - if merge improves the objective, accept

Write access to two nodes to be merged

Summary so far

Algorihthm	
Truncation-free VI:	
parameter learning	Tree structure (truncation level) fixed
structure updating	Each birth/merge modify a *local* region of the tree

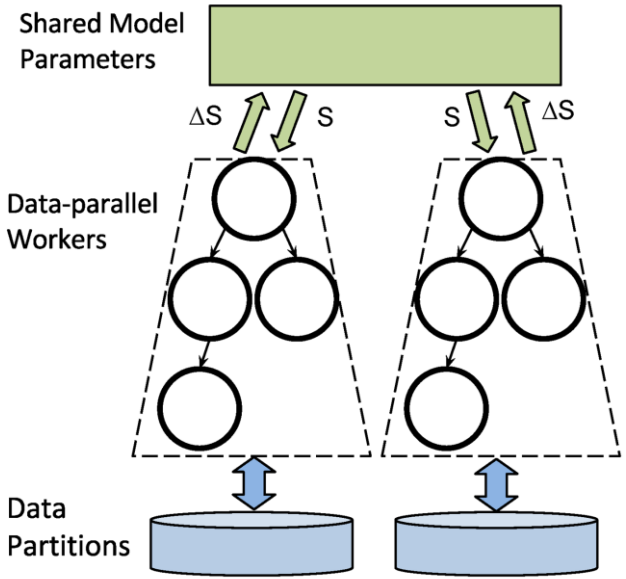
Distributed Implementation



Data parallel for parameter learning

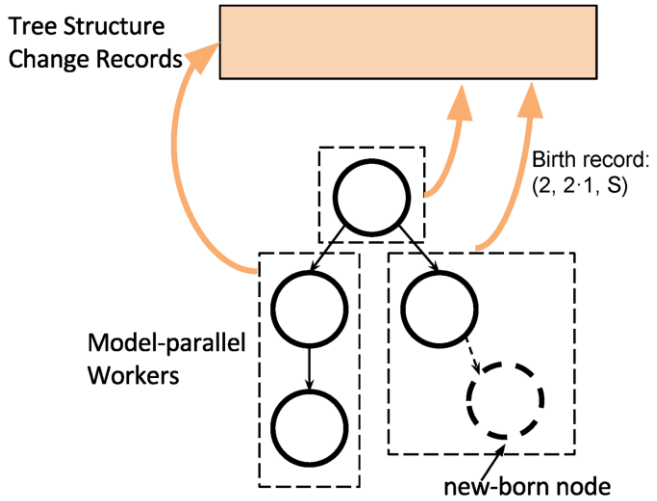
- Each worker maintains a model copy
- Full-data sufficient statistics S is stored on parameter server

- Additivity of the sufficient statistics
  - aggregate updates from different workers by simple accumulation
- Tree structure (truncation level) is fixed
  - no model alignment is required



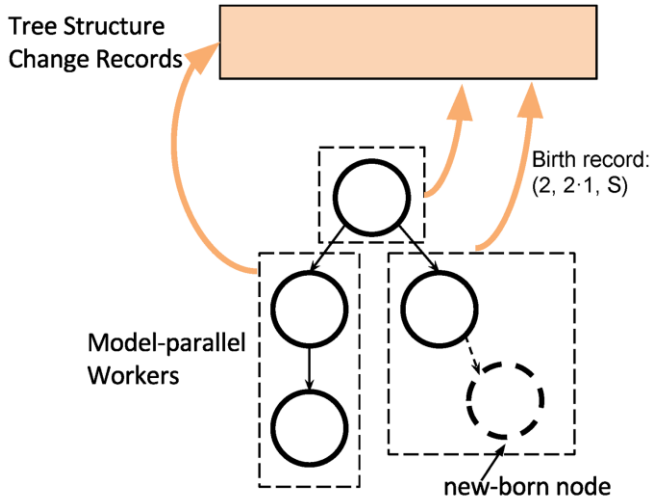
Model parallel for structure updating

- Birth/merge moves only access to local regions of the whole tree
  - (logically) partition the tree into multiple parts
  - processors do birth/merge independently



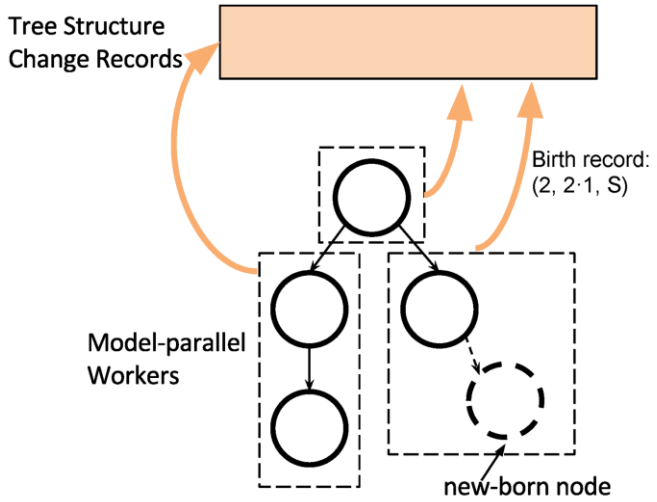
Model parallel for structure updating

- Birth/merge moves only access to local regions of the whole tree
  - (logically) partition the tree into multiple parts
  - processors do birth/merge independently
- Synchronize tree structure across processors
  - each processor broadcasts its assigned tree parts
    - high communication cost



Model parallel for structure updating

- Birth/merge moves only access to local regions of the whole tree
  - (logically) partition the tree into multiple parts
  - processors do birth/merge independently
- Synchronize tree structure across processors
  - each processor broadcasts its assigned tree parts
  - processors exchange lightweight *operation records*
    - birth: $(\epsilon,\ \epsilon \cdot k,\ S_{\epsilon},\ S_{\epsilon \cdot k})$
    - merge: $(\epsilon_a,\ \epsilon_b)$
  - each processor ``re-do'' all operations on its model copy
    - no need of restricted VI/accept-reject assessment



Experimental Results

Experimental Results

- Near-optimal (near-linear) speedup

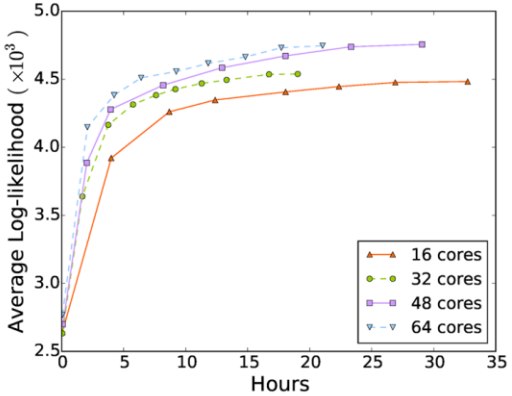
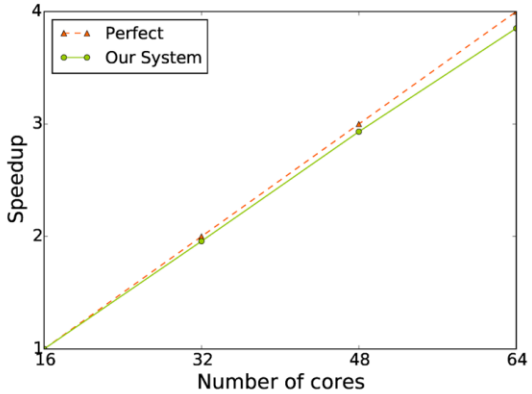


Figure 3. Converg on PubMed



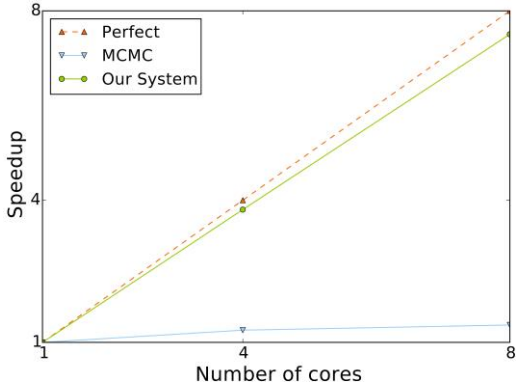


Figure 4. Speedup on PubMed

- PubMed dataset: 8.4M documents, 720M tokens
- Using 4 machines (64 cores), obtain a tree model with 10K topics in 36 hrs

Experimental Results

- Long-tail topics

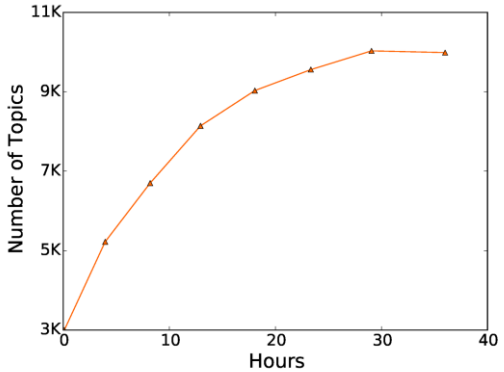


Figure 6. Growth of the tree

topic mass	top-6 words	topic label
27.9	at2r, at1r, receptor, type_2, angiotensin_ii, ang_ii	**Angiotensin II Type-2 Receptor**
5.9	fusion, thoracoscopic, spinal, treatment, level, surgery	**Thoracoscopic Spinal Fusion**
4.1	azt, nucleosides, pharmacokinetic, monkey, subcutaneous, methyl	**Pharmacokinetics with AZT in Monkeys**

Table 2. Long-tail topics from PubMed data. The topic labels are obtained by looking into the documents associated with the topics.

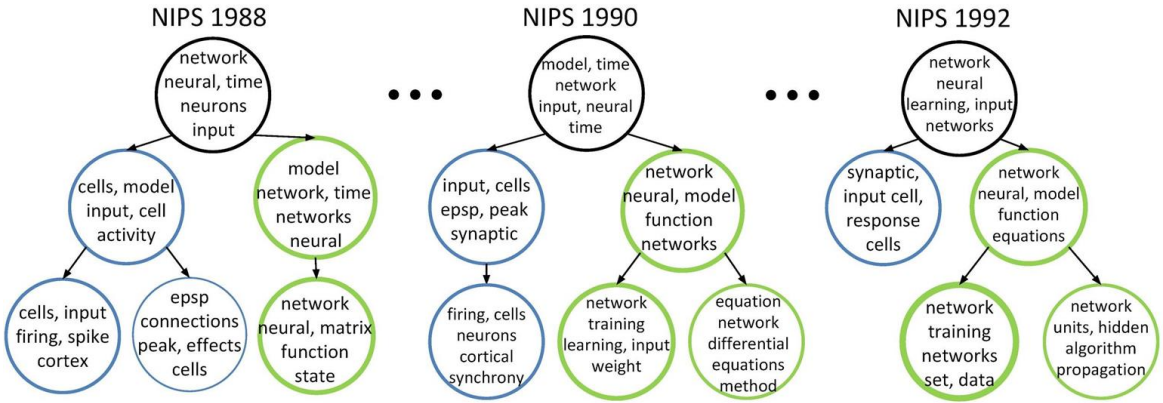
Experimental Results

- Competitive inference quality with MCMC
- Held-out average log-likelihood

Method	MCMC (Dubey et al., 2014)	Our algorithm
PNAS	4562±116	4479±103
NIPS	9811±232	9712±143

Table 3. Heldout average log-likelihood on two datasets.

- Topic variation of NIPS



Conclusion

- Truncation-free VI for dependent nonparametric trees
  - Memoized VI for parameter learning
  - Birth-merge moves for structure updating
- Distributed learning
  - Data parallel for parameter learning
  - Model parallel for structure updating
    - communication-efficient synchronization by exchanging operation records
- Supports orders-of-magnitude larger dataset and model

Thanks