Guaranteed Scalable Learning of Latent Tree Models

Furong Huang *
University of Maryland

Niranjan Uma Naresh
Microsoft

Ioakeim Perros
HEALTH[at]SCALE

Robert Chen
Flatiron Health

Jimeng Sun
Georgia Institute of Technology

Anima Anandkumar
California Institute of Technology

Abstract

We present an integrated approach for structure and parameter estimation in latent tree graphical models. Our overall approach follows a "divide-and-conquer" strategy that learns models over small groups of variables and iteratively merges onto a global solution. The structure learning involves combinatorial operations such as minimum spanning tree construction and local recursive grouping; the parameter learning is based on the method of moments and on tensor decompositions. Our method is guaranteed to correctly recover the unknown tree structure and the model parameters with low sample complexity for the class of linear multivariate latent tree models which includes discrete and Gaussian distributions, and Gaussian mixtures. Our bulk asynchronous parallel algorithm is implemented in parallel and the parallel computation complexity increases only logarithmically with the number of variables and linearly with dimensionality of each variable.

1 INTRODUCTION

Latent tree graphical models are a popular class of latent variable models, where a probability distribution involving observed and hidden variables are Markovian on a tree. Since the structure of (observable and hidden) variable interactions is approximated as a tree, inference on latent trees can be carried out exactly through a simple belief propagation [Pearl, 1988]. Therefore, latent tree graphical models present a good trade-off between model accuracy and computational complexity. They are applicable in many domains [Durbin et al., 1999, Choi et al., 2012a,b, Wang and Li, 2013], where it is natural to expect hierarchical or sequential relationships among the variables through a hidden-Markov model.

The task of learning a latent tree model consists of two parts: *learning the tree structure* and *learning the parameters of the tree*. We list the challenges in learning a latent tree model as follows:

1. **Challenge 1: Consistent structure learning.** The location and the number of latent variables are hidden and the marginalized graph over the observable variables no longer conforms to a tree structure.
2. **Challenge 2: Consistent parameter estimation.** Parameter estimation in latent tree model is typically carried out through Expectation Maximization (EM) or other local search heuristics [Choi et al., 2011]. These methods have no consistency guarantees, suffer from the problem of local optima and are not easily parallelizable.
3. **Challenge 3: Computational complexity of structure learning.** Complexity of existing algorithms are typically polynomial with the number of variables p (i.e., observed nodes) as discussed in Anandkumar et al. [2011], Choi et al. [2011]. These methods are sequential in nature and therefore are not scalable for large p.
4. **Challenge 4: Efficient structure and parameter estimation in parallel.** Existing methods treat structure learning and parameter estimation sequentially – the parameter estimation can only be done after completion of structure learning. Therefore it is highly inefficient if the goal is to estimate only a small subset of variables/nodes.

Choi et al. [2011] addressed **challenge 1** using recursive grouping algorithms for Gaussian and discrete variables only. It remains unclear how to extend to variables in high-dimensions. As for **challenge 2**, there is no work for guaranteed parameter estimation of latent tree models: although Anandkumar et al. [2012b] proposed ten-

*Email: furongh@cs.umd.edu.

sor decomposition mechanisms for simple latent variable models such as multi-view and mixture of Gaussian models, extending those tensor decomposition mechanisms to hierarchical models such as latent trees are non-trivial, and involves alignment of locally estimated parameters. No existing work addresses **challenge 3** or implements a structure learning of latent tree in time less than polynomial with p. Lastly for **challenge 4**, there is no obvious way to (and no prior work did) directly parallelize these sequential methods without losing global consistency guarantees.

We close the loop of consistent learning of latent tree model in high-dimensions via an integrated parallel approach to simultaneous structure and parameter estimation. Our method overcomes all above challenges.

Benefits of integrated structure and parameter estimation. The locally implemented and yet globally consistent simultaneous recovery of structure and parameter is the key to our algorithm's efficiency. Without this integration, parameter estimation has to wait until the completion of structure recovery, making the algorithm intrinsically sequential and thus less efficient. Another attractive feature of our method is that it is amenable for user interaction allowing the user to provide feedback and change the course of various stages of the algorithm in a smooth manner: a quality not found in other graphical model learning methods. More precisely, the user can select neighborhoods for adding hidden variables, using scores such as BIC. This is suggested in Choi et al. [2011], but the re-estimation of parameters through EM and sequential execution makes it expensive. In our approach, no re-estimation of parameters are needed since the structure and parameter estimation under our framework go hand-in-hand, i.e., as the structure of the tree is obtained, parameters are dynamically estimated.

Summary of Contributions

Theoretical contributions. **(1)** Given enough computational resources, our method achieves consistent latent tree structure learning with $\log(p)$ computational complexity in a "divide-and-conquer" manner, improving the state-of-the-art poly(p) complexity. We present a rigorous proof on the global consistency of the structure and parameter estimation under the "divide-and-conquer" framework. Our consistency guarantees are applicable to a broad class of linear multivariate latent tree models including discrete distributions, continuous multivariate distributions (e.g. Gaussian), and mixed distributions such as Gaussian mixtures. This model class is much more general than discrete models prevalent in most previous works on latent tree models [Mossel and Roch, 2005, Mossel, 2007, Erdos et al., 1999, Anandkumar et al., 2013]. **(2)** Our algorithm guarantees consistent latent tree parameter estimation using inverse method of moments and tensor decomposition, the first guarantee for consistent parameter estimation in latent tree whose sample complexity is $\log(k)$. In contrast, the previous state-of-the-art is the EM algorithm [Choi et al., 2011]. In addition, we extend tensor decomposition [Anandkumar et al., 2012b] in models with simple structure to hierarchical tensor decomposition for more complex models. **(3)** Moreover, we carefully integrate structure learning with parameter estimation, based on tensor spectral decompositions [Anandkumar et al., 2012b]. The locally implemented and yet globally consistent recovery of structure and parameter simultaneously is the key for the efficiency of our algorithm. Without this integration, parameter estimation has to wait until the completion of structure recovery, making the algorithm intrinsically sequential and thus less efficient. Finally, our approach has a high degree of parallelism, and is *bulk asynchronous* parallel [Gerbessiotis and Valiant, 1994]. Thus, we propose a parallel and an integrated method for structure and parameter estimation without sacrificing on global correctness guarantees.

Empirical justification. We demonstrate that our algorithm is fast and scalable up to thousands of nodes and hundreds of thousands of data dimensions – for example, it takes about 1 minute to run our method on nine nodes each with a dimensionality of 100,000, and about 70 minutes when the number of nodes is 729 on a single workstation. Conceivably our method can be scalable to even larger dimensions by employing a cloud based implementation of the method. In our experiments, we obtain a high level of accuracy, for both structure and parameter recovery, even as the problem dimensions increase. In contrast, the EM method is stuck in local optima and has bad accuracy in parameter estimation, even for an extremely small example with nine nodes. Thus, we demonstrate a scalable and guaranteed approach for learning latent tree graphical models.

Application contribution. In this work, we use latent tree model for discovering a hierarchy among diseases based on co-morbidities exhibited in patients' health records, i.e. co-occurrences of diseases in patients. In particular, two large healthcare datasets of 30K and 1.6M patients are used to build the latent disease trees. Our algorithm is used to discover hidden patterns, or concepts reflecting co-occurrences of particular diagnoses in patients in outpatient and intensive care settings. While such a task is currently done through manual analysis of the data, our method provides an automated way to discover novel clinical concepts from high dimensional, multi-modal data. Clinically meaningful disease clusters are identified as shown in fig 7.

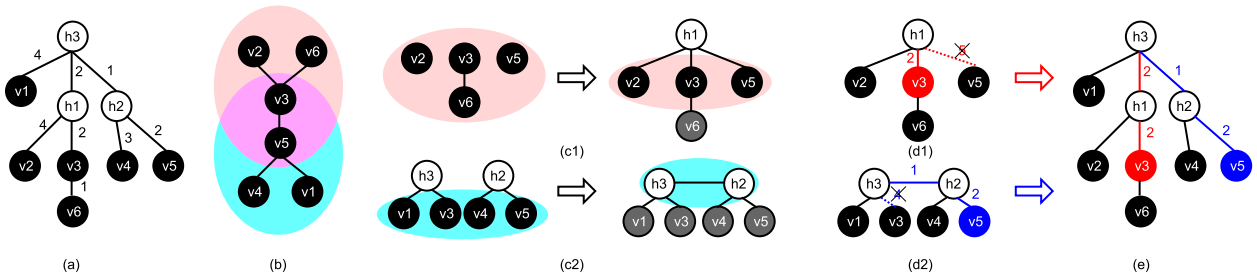

Figure 1: **(a)** Ground truth latent tree to be estimated, numbers on edges are *multivariate information distances*. Filled nodes are observed and blank nodes are hidden. **(b)** MST constructed using the *multivariate information distances*. v_3 and v_5 are internal nodes (leaders). Note that *multivariate information distances* are additive on latent tree, not on MST. **(c1)** LRG on nbd$[v_3, \text{MST}]$ to get local structure $\mathbf{N}_3$. Pink shadow denotes the active set. Local parameter estimation is carried out over triplets with joint node, such as (v_2, v_3, v_5) with joint node h_1. **(c2)** LRG on nbd$[v_5, \text{MST}]$ to get local structure $\mathbf{N}_5$. Cyan shadow denotes the active set. **(d1)(d2)** Merging local sub-trees. Path$(v_3,v_5;\ \mathbf{N}_3)$ and path$(v_3,v_5;\ \mathbf{N}_5)$ conflict. **(e)** Final recovery.

2 LATENT TREE GRAPHICAL MODEL

We denote $[n] := \{1, \ldots, n\}$. Let $\mathcal{T} := (\mathcal{V}, \mathcal{E})$ denote the ground-truth undirected tree with vertex set $\mathcal{V}$ and edge set $\mathcal{E}$. The *neighborhood* of a node v_i on tree $\mathcal{T}$, nbd$[v_i, \mathcal{T}]$, is the set of nodes to which v_i is directly connected on the tree. Leaves which have a common neighboring node are defined as *siblings*, and the common node is referred to as their *parent*. Let N denote the number of samples. An example of latent tree is depicted in Figure 1(a).

There are two types of variables on the nodes, namely, the observable variables, denoted by $\mathcal{X} := \{x_1, \ldots, x_p\}$ ($p := |\mathcal{X}|$), and hidden variables, denoted by $\mathcal{H} := \{h_1, \ldots, h_m\}$ ($m := |\mathcal{H}|$). Let $\mathcal{Y} := \mathcal{X} \cup \mathcal{H}$ denote the complete set of variables and let y_i denote the random variable at node $v_i \in \mathcal{V}$, and similarly let y_A denote the set of random variables in set A. A *graphical model* is defined as follows: given the neighborhood nbd$[v_i, \mathcal{T}]$ of any node $v_i \in \mathcal{V}$, the variable y_i is conditionally independent of the rest of the variables in $\mathcal{V}$, i.e., $y_i \perp\!\!\!\perp y_j | y_{\text{nbd}[v_i,\mathcal{T}]},\ \forall v_j \in \mathcal{V} \backslash \{v_i \cup \text{nbd}[v_i, \mathcal{T}]\}$.

Linear models. We consider the class of linear latent tree models. The observed variables x_i are random vectors of length d_i, i.e., $x_i \in \mathbb{R}^{d_i}$, $\forall i \in [p]$ while the latent nodes are k-state categorical variables, i.e., $h_i \in \{e_1, \ldots, e_k\}$, where $e_j \in \mathbb{R}^k$ is the j^{th} standard basis vector. Although d_i can vary across variables, we use d for notation simplicity. In other words, for notation simplicity, $x_i \in \mathbb{R}^d$, $\forall i \in [p]$ is equivalent to $x_i \in \mathbb{R}^{d_i}$, $\forall i \in [p]$. For any variable y_i with neighboring hidden variable h_j, we assume a linear relationship: $\mathbb{E}[y_i|h_j] = A_{y_i|h_j} h_j$, where transition matrix $A_{y_i|h_j} \in \mathbb{R}^{d \times k}$ is assumed to have full column rank, $\forall y_i, h_j \in \mathcal{V}$. This implies that $k \leq d$, which is natural if we want to enforce a parsimonious model for fitting the observed data. If two observable variables interact through at least a hidden variable (i.e., there is at least a hidden variable along the path between the two nodes), we have $\mathbb{E}[y_a y_b^\top] = \sum_{e_i} \mathbb{E}[h_j = e_i] A_{y_a|h_j=e_i} A_{y_b|h_j=e_i}^\top$. We see that $\mathbb{E}[y_a y_b^\top]$ is of rank k since $A_{y_a|h_j=e_i}$ or $A_{y_b|h_j=e_i}$ is of rank k. We consider the class of tree models where it is possible to recover the latent tree model uniquely.

Assumption 2.1 (Structure Identifiable Condition [Choi et al., 2011])**.** *Each hidden variable has at least three neighbors (which can be either hidden or observed).*

Remark: Our structure identifiable condition ensures a minimal latent tree with no redundant nodes. Therefore our goal is to consistently learn the set of identifiable latent tree model.

Assumption 2.2 (Parameter Identifiable Condition)**.** *(1) The pairwise correlation matrix $\mathbb{E}[x_a x_b^\top]$, between neighboring observable variables x_a and x_b, is of rank k. (2) Any two variables connected by an edge in the tree model are neither perfectly dependent nor independent.*

Remark: (1) When two observed nodes are directly connected according to the structure learned, the conditional probability decomposes into k factors. This assumption is mild and applies to various applications where observed nodes have intrinsic memberships (low dimensional representation) through which they interact with other nodes. (2) This condition is necessary for the identifiability of parameters as stated in Choi et al. [2011].

Learning objective. Our goal is to learn the ground-truth structure $\mathcal{T} := (\mathcal{V}, \mathcal{E})$ and parameter $A_{y_i|h_j}$ given N examples of $\left\{\left(x_1^{(j)}, \ldots, x_p^{(j)}\right)\right\}_{j=1}^N$.

3 APPROACH OVERVIEW

The overall approach is depicted in Figure 1, where (a) and (b) show the data preprocessing step, (c) - (e) illustrate the divide-and-conquer step for structure and parameter learning. We will describe each step in details in the later sections.

We start with the parallel computation of pairwise *multivariate information distances* (defined in Definition 4.1) between all pairs of observed variables. Information distance roughly measures the correlation between different pairs of observed variables and requires SVD computations. For this example in Figure 1(a), the multivariate information distances between pairs of $\{v_1, \ldots, v_6\}$ will be estimated empirically via Equation (2). Note that the tree structure is hidden and unknown.

Then, as depicted in Figure 1(b), a Minimum Spanning Tree (MST) over the estimated pairwise multivariate information distances is constructed over observable variables in parallel [Bader and Cong, 2006]. The local groups (pink and cyan) are also obtained through MST so that they are available for the structure and parameter learning step that follows.

The structure and parameter learning is done jointly through a divide-and-conquer strategy. Figure 1(c) illustrates the divide step (or local learning), where local structure (LRG) and parameter estimation (tensor decomposition) is performed (Procedure 1).

Our algorithm also performs the local merge to obtain group level structure and parameter estimates. As shown in Figure 1(d1) and (d2), after the local structure and parameter learning is finished within the groups, we perform merge operations among groups, again guided by the Minimum Spanning Tree structure. For the structure estimation it consists of a union operation of sub-trees (Procedure 2); for the parameter estimation, it consists of linear algebraic operations (Procedure 3) – Since our method is unsupervised, an alignment procedure of the hidden states is carried out which finalizes the global estimates of the tree structure and the parameters.

4 STRUCTURE LEARNING

Structure learning in graphical models involves finding the underlying Markov graph, given the observed samples. For latent tree models, structure can be estimated via distance based methods. This involves computing certain *information* distances between any pair of observed variables, and then finding a tree which fits the computed distances.

Multivariate information distances: We propose a distance metric, that is additive on the ground truth tree, for multivariate linear latent tree models. For a pair of (observed or hidden) variables y_a and y_b, consider the pairwise correlation matrix $\mathbb{E}\left[y_a y_b^\top\right]$ (the expectation is over samples). Note that its rank is k, dimension of the hidden variables.

Definition 4.1. *The multivariate information distance between nodes i and j is defined as*

$$dist(v_a, v_b) := -\log \frac{\prod_{i=1}^{k} \sigma_i\left(\mathbb{E}(y_a y_b^\top)\right)}{\sqrt{\det(\mathbb{E}(y_a y_a^\top))\det(\mathbb{E}(y_b y_b^\top))}} \tag{1}$$

where $\{\sigma_1(\cdot), \ldots, \sigma_k(\cdot)\}$ are the top k singular values.

Remark: This is an extension of the distance measure to multivariate variables, which is not introduced in Choi et al. [2011]. Note that definition 4.1 suggests that this multivariate information distance allows heterogeneous settings where the dimensions of y_a and y_b are different (and $\geq k$). For finite number (N) of samples the empirical estimation of the multivariate information distance is estimated as

$$\widehat{\text{dist}}(v_a, v_b) = -\log \frac{\prod_{i=1}^{k} \sigma_i\left(\sum_{j=1}^{N} (y_a^{(j)}(y_b^{(j)})^\top)\right)}{\sqrt{\det(\sum_{j=1}^{N} (y_a^{(j)}(y_a^{(j)})^\top))\det(\sum_{j=1}^{N} (y_b^{(j)}(y_b^{(j)})^\top))}} \tag{2}$$

For latent tree models, we can find information distances which are provably *additive* on the underlying tree in expectation, i.e. the expected distance between any two nodes in the tree is the sum of distances along the path between them.

Lemma 4.2. *The multivariate information distance is additive on the tree $\mathcal{T}$, i.e., dist(v_a, v_c) = dist(v_a, v_b) + dist(v_b, v_c), where v_b is a node in the path from v_a to v_c and v_a,v_b,$v_c \in \mathcal{V}$.*

Refer to Appendix E for proof. The empirical distances can be computed via rank-k SVD of the empirical pairwise moment matrix $\hat{\mathbb{E}}[y_a y_b^\top]$. Note that the distances for all the pairs can be computed in parallel.

To estimate the structure, we could do reverse engineering to figure out where to introduce hidden nodes or how nodes are connected since the multivariate information distances should be additive on the ground-truth tree. Below we extend LRG in Choi et al. [2011] to be in parallel to consistently and efficieintly estimate the structure.

Formation of local groups via MST: Once the empirical distances are computed, we construct a Minimum Spanning Tree (MST), based on those distances. Note that the MST can be computed efficiently in parallel [Vineet et al., 2009, Michael, 2012]. We now form groups of observed variables over which we carry out

learning independently, without any coordination. These groups are obtained by the (closed) neigborhoods in the MST, i.e. an internal node and its one-hop neighbors form a group. The corresponding internal node is referred to as the *group leader*. See Figure 1(b).

Local recursive grouping (LRG): Once the groups are constructed via neighborhoods of MST, we construct a sub-tree with hidden variables in each group (in parallel) using the recursive grouping introduced in Choi et al. [2011], depicted in 1(c1) and (c2). The recursive grouping uses the multivariate information distances and decides the locations and numbers of hidden nodes. It proceeds by deciding which nodes are "siblings" or "parent and child", using the following property that determines a pair of siblings or a parent and child pair.

Property 4.3 (Siblings)**.** *Define a potential function as* $\Phi(v_i, v_j; v_a) := dist(v_i, v_a) - dist(v_j, v_a)$*. A pair of observed nodes* (v_i, v_j) *are siblings with parent* v_l*, if the potential function is fixed* $\forall v_a, v_b$ *in the active set*

$$\Phi(v_i, v_j; v_a) \equiv \Phi(v_i, v_j; v_b) = dist(v_i, v_l) - dist(v_j, v_l).$$

From additivity of the (expected) information distances, we have $\text{dist}(v_i, v_a) = \text{dist}(v_i, v_l) + \text{dist}(v_l, v_a)$ and similarly for $\text{dist}(v_j, v_a)$. Thus, we have $\Phi(v_i, v_j; v_a) := \text{dist}(v_i, v_a) - \text{dist}(v_j, v_a) = \text{dist}(v_i, v_l) - \text{dist}(v_j, v_l)$, which is independent of node v_a.

Property 4.4 (Parent-Child)**.** *A pair of observed nodes* (v_l, v_i) *is a parent* (v_l) *and child* (v_i) *pair , if* $\forall v_a, v_b$ *in the active set*

$$\Phi(v_i, v_l; v_a) = dist(v_i, v_a) - dist(v_l, v_a) = dist(v_i, v_l)$$

Thus, comparing the quantity $\Phi(v_i, v_j; v_a)$ for all nodes v_a allows us to determine whether v_i and v_j are siblings or parent-child. For instance, in (c1), firstly the active set is $\{v_2, v_3, v_5, v_6\}$, v_3 and v_6 are detected as parent and child because for all other nodes in the active set, i.e., v_2 and v_5, we have $\Phi(v_6, v_3; v_2) = \text{dist}(v_6, v_2) - \text{dist}(v_3, v_2) = \Phi(v_6, v_3; v_5) = \text{dist}(v_6, v_5) - \text{dist}(v_3, v_5) = \text{dist}(v_3, v_6)$; secondly the active set is updated to $\{v_2, v_3, v_5\}$ (children are deleted), v_2 and v_3 are detected as siblings because for all other nodes in the active set, i.e., v_5, we have $\Phi(v_2, v_3; v_5) \equiv \Phi(v_2, v_3; v_5)$. Similarly v_2 and v_5 are detected as siblings and therefore v_2, v_3, v_5 are detected as siblings in the second step.

Once the siblings are inferred, the hidden nodes are introduced, and the same procedure repeats to construct the higher layers. Note that whenever we introduce a new hidden node h_{new} as a parent, we need to estimate multivariate information distance between h_{new} and nodes in active set Ω. This is discussed in Choi et al. [2011] in detail.

Finite samples Our algorithm produces consistent results even when there is finite number of samples and the estimation of $\text{dist}(v_a, v_b)$ is noisy (see Equation (2)).

Parent-child: In the noiseless case, if $\Phi(v_a, v_b; v_c) = \text{dist}(v_a, v_b),\ \forall\ v_c \in \Omega \backslash \{v_a, v_b\}$, then v_a is a leaf node and v_b is its parent. However, in the noisy case, we modify it to if $|\widehat{\Phi}(v_a, v_b; v_c) - \widehat{\text{dist}}(v_a, v_b)| \leq \epsilon,\ \forall\ v_c \in \Omega \backslash \{v_a, v_b\}$, then v_a is a leaf node and v_b is its parent.

Siblings: In the noiseless case, if $-\text{dist}(v_a, v_b) < \Phi(v_a, v_b; v_c) = \Phi(v_a, v_b; v_c') < \text{dist}(v_a, v_b), \forall v_c, v_c' \in \Omega \backslash \{v_a, v_b\}$, then v_a and v_b are siblings. However, in the noisy case, we modify it to if $|\widehat{\Phi}(v_a, v_b; v_c) - \widehat{\Phi}(v_a, v_b; v_c')| \leq \epsilon$, and $|\widehat{\Phi}(v_a, v_b; v_c)| < \widehat{\text{dist}}(v_a, v_b) + \epsilon$, $\forall v_c, v_c' \in \Omega \backslash \{v_a, v_b\}$, then v_a and v_b are siblings.

The threshold in the procedure is not a simple heuristic but is supported by theory: Lemma 7.2 provides a consistency guarantee on the learning of latent tree structure using the noisy pairwise distance. For a given precision ϵ and a given number of variables p, we derive the number of samples N needed for our algorithm to be consistent.

We describe the LRG in details with integrated parameters estimation in Procedure 1 in Section 6. In the end, we obtain a sub-tree over the local group of variables. After this *local recursive grouping test*, we store the neighborhood relationship for the leader v_i using an adjacency list $\mathcal{N}_i$. We call the resultant local structure the *latent sub-tree*.

5 PARAMETER ESTIMATION

Along with the structure learning, we use a moment-based spectral learning technique for parameter estimation. This is a guaranteed and fast approach to recover parameters via moment matching for third order moments of the observed data. In contrast, traditional approaches such as Expectation Maximization (EM) suffer from spurious local optima and cannot provably recover the parameters.

A latent tree with three leaves: We first consider an example of three observable leaves x_1, x_2, x_3 (i.e., a triplet) with a common hidden parent h. We then clarify how this can be generalized to learn the parameters of the latent tree model. Let $\otimes$ denote for the tensor product. For example, if $x_1, x_2, x_3 \in \mathbb{R}^d$, we have $x_1 \otimes x_2 \otimes x_3 \in \mathbb{R}^{d \times d \times d}$ and $[x_1 \otimes x_2 \otimes x_3]_{ijk} = x_1(i) x_2(j) x_3(k)$.

Property 5.1 (Tensor decomposition for triplets)**.** *For a linear latent tree model with three observed nodes* v_1, v_2, v_3 *with a common hidden parent* h*, we have* $\mathbb{E}(x_1 \otimes x_2 \otimes x_3) = \sum_{r=1}^{k} \mathbb{P}[h = e_r] A^r_{x_1|h} \otimes A^r_{x_2|h} \otimes$

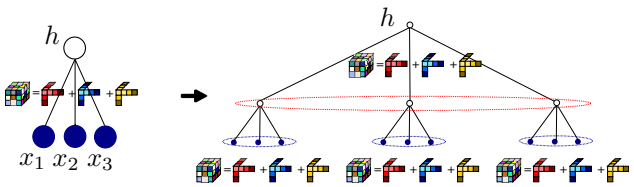

Figure 2: Parameter estimation in latent tree using hierarchical tensor decomposition.

$A^r_{x_3|h}$, where $A^r_{x_i|h} = \mathbb{E}(x_i|h = e_r)$, i.e., r^{th} column of the transition matrices from h to x_i. The hidden variable separates the observed variables on the latent tree, therefore the observed variables are conditionally independent given the hidden variable, according to the definition of graphical model. The tensor decomposition method of Anandkumar et al. [2012b] provably recovers the parameters $A_{x_i|h}$, $\forall i \in [3]$, and $\mathbb{P}[h]$.

Remark: To recover the model parameter $A_{x_i|h}$, we need an empirical estimation of $\mathbb{E}(x_1 \otimes x_2 \otimes x_3)$, computed as $\widehat{\mathbb{E}}(x_1 \otimes x_2 \otimes x_3) = \frac{1}{N} \sum_{j=1}^{N} x_1^{(j)} \otimes x_2^{(j)} \otimes x_3^{(j)}$. We then use tensor decomposition on $\widehat{\mathbb{E}}(x_1 \otimes x_2 \otimes x_3)$ to estimate $A_{x_i|h}$. Once we obtain the conditional distribution of observed variable given the hidden variable $\mathbb{P}(x_i|h)$ and the marginal distribution of the $\mathbb{P}(h)$ after tensor decomposition, we sample from the posterior distribution $\mathbb{P}(h|X)$ for each corresponding observed example in parameter estimation procedure.

Tensor decomposition for learning latent tree models: We employ the above approach for learning latent tree model parameters as follows: for every triplet of variables y_a, y_b, and y_c (hidden or observed), we consider the hidden variable h_i which is the *joining point* of y_a, y_b and y_c on the tree, i.e., the node that path(y_a, y_b), path(y_b, y_c) and path(y_a, y_c) go through. They form a *triplet* model, for which we employ the tensor decomposition procedure. However, it is wasteful to do it over all the triplets in the latent tree. In the next section, we demonstrate how we efficiently estimate the parameters simultaneously as we learn the structure, and minimize the tensor decompositions required for estimation.

The samples from the posterior distribution $\mathbb{P}(h|X)$ are required for parameter estimation (tensor decomposition), although it is not required for the computation of additive distance. The tensor decomposition for each triplets on the tree requires sample estimation of moments on variables. When estimating parameters related to the observed variables, no sampling is needed as samples are observed. However estimating parameters for the internal hidden nodes requires sampling from the hidden variables to form the moments of the triplets.

6 INTEGRATED STRUCTURE AND PARAMETER ESTIMATION

So far, we have described, in a high-level, procedures of structure estimation through local recursive grouping (LRG) and parameter estimation through tensor decomposition over triplets of variables, respectively. We now introduce an integrated and efficient approach which brings all these ingredients together. We provide merging steps to obtain a global model that is consistent, using the sub-trees and parameters learned over local groups.

6.1 Structure with Parameter Estimation

Intuitively, we find efficient groups of triplets to carry out tensor decomposition simultaneously, as we estimate the structure through recursive grouping. In LRG, pairs of nodes are recursively grouped as siblings or as parent-child. As this process continues, we carry out tensor decompositions whenever there are siblings presented as triplets. If there are only a pair of siblings, we find an observed node with closest distance to the pair. Once the tensor decompositions are carried out on the observed nodes, we proceed to structure and parameter estimation of the added hidden variables. The samples of the hidden variables can be obtained via the posterior distribution, which is learnt earlier through tensor decomposition. This allows us to predict information distances and third order moments among the hidden variables as process continues. See algorithm in Procedure 1.

The divide-and-conquer local spectral parameter estimation is superior compared to popular EM-based method [Choi et al., 2011], which is slow and prone to local optima. More importantly, EM can only be applied on a stable structure since it is a global update procedure. Our proposed spectral learning method, in contrast, is applied locally over small groups of variables, and is a guaranteed learning with sufficient number of samples [Anandkumar et al., 2012b]. Moreover, since we integrate structure and parameter learning, we avoid recomputing the same quantities, e.g. SVD computations are required both for structure estimation (for computing distances) and parameter estimation (for whitening the tensor). Combining these operations results in huge computational savings (see Section 7 for the exact computational complexity of our method).

6.2 Merging and Alignment Correction

We have so far learnt sub-trees and parameters over local groups of variables, where the groups are determined by the neighborhoods of the MST. The challenge now is to combine them to obtain a globally consistent estimate.

Procedure 1 LRG with Parameter Estimation

Input: Internal nodes $\mathcal{X}_{\text{int}}$ on MST, for each $v_i \in \mathcal{X}_{\text{int}}$, active set $\Omega := \text{nbd}[v_i; \text{MST}]$, precision ϵ

Output: for each $v_i \in \mathcal{X}_{\text{int}}$, local sub-tree adjacency matrix $\mathcal{N}_i$, and $\widehat{\mathbb{E}}[y_a|y_b]$ for all $(v_a, v_b) \in \mathcal{N}_i$.

Active set $\Omega \leftarrow \text{nbd}[v_i; \text{MST}]$
while $|\Omega| > 2$ **do**
  for all $v_a, v_b \in \Omega$ **do**
    if $|\widehat{\Phi}(v_a, v_b; v_c) - \widehat{\text{dist}}(v_a, v_b)| \leq \epsilon, \ \forall\ v_c \in \Omega \backslash \{v_a, v_b\}$ **then**
      v_a is a leaf node and v_b is its parent,
      Eliminate v_a from Ω.
    if $|\widehat{\Phi}(v_a, v_b; v_c) - \widehat{\Phi}(v_a, v_b; v_c')| \leq \epsilon$, and $|\widehat{\Phi}(v_a, v_b; v_c)| < \widehat{\text{dist}}(v_a, v_b) + \epsilon$, $\forall v_c, v_c' \in \Omega \backslash \{v_a, v_b\}$ **then**
      v_a and v_b are siblings,eliminate v_a and v_b from Ω, add h_{new} to Ω.
      Introduce new hidden node h_{new} as parent of v_a and v_b.
      if more than 3 siblings under h_{new} **then**
        find v_c in siblings,
      else
        find $v_c = \arg\min_{v_c \in \Omega} \text{dist}(v_a, v_c)$.
      Estimate empirical third order moments $\widehat{\mathbb{E}}(y_a \otimes y_b \otimes y_c)$
      Decompose $\widehat{\mathbb{E}}(y_a \otimes y_b \otimes y_c)$ to get $\widehat{\Pr}[h_{\text{new}}]$ and $\widehat{\mathbb{E}}(y_r|h_{\text{new}}), \forall r = \{a, b, c\}$.

There are non-trivial obstacles to achieving this: first, the constructed local sub-trees span overlapping groups of observed nodes, and possess conflicting paths. Second, local parameters need to be re-aligned as we merge the subtrees to obtain globally consistent estimates. To be precise, different tensor decompositions lead to permutations of the hidden labels (columns permutations of the transition matrices) across triplets. Thus, we need to find the permutation matrix best correcting the alignment of hidden states of the transition matrices, so as to guarantee global consistency.

Structure Union: We now describe the procedure to merge the local structures. We merge them in pairs to obtain the final global latent tree. Recall that $\mathcal{N}_i$ denotes a sub-tree constructed locally over a group, whose leader is node v_i. Consider a pair of subtrees $\mathcal{N}_i$ and $\mathcal{N}_j$, whose group leaders v_i and v_j are neighbors on the MST. Since v_i and v_j are neighbors, both the sub-trees contain them, and have different paths between them (with hidden variables added). Moreover, note that this is the only conflicting path in the two subtrees. We now describe how we can resolve this: in $\mathcal{N}_i$, let h_1^i be the neighboring hidden node for v_i and h_2^i be the neighbor of v_j. There could be more hidden nodes between h_1^i and h_2^i. Similarly, in $\mathcal{N}_i$, let h_1^j and h_2^j be the corresponding nodes in $\mathcal{N}_j$. The shortest path between v_i and v_j in the two sub-trees are given as follows:

$$\text{path}(v_i, v_j; \mathcal{N}_i) := [v_i - h_1^i - \ldots - h_2^i - v_j] \quad (3)$$

$$\text{path}(v_i, v_j; \mathcal{N}_j) := [v_i - h_1^j - \ldots - h_2^j - v_j] \quad (4)$$

Then the union path is formed as follows:

$$\text{merge}(\text{path}(v_i, v_j; \mathcal{N}_i), \text{path}(v_i, v_j; \mathcal{N}_j)) := [v_i - h_1^i - \ldots - h_2^i - h_1^j \ldots h_2^j - v_j] \quad (5)$$

In other words, we retain the immediate hidden neighbor of each group leader, and break the paths on the other end. For example in Figure 1(d1,d2), we have the path $v_3 - h_1 - v_5$ in $\mathcal{N}_3$ and path $v_3 - h_3 - h_2 - v_5$ in $\mathcal{N}_5$. The resulting path is $v_3 - h_1 - h_3 - h_2 - v_5$, as see in Figure 1(e). After the union of the conflicting paths, the other nodes are attached to the resultant latent tree. We present the pseudo code in Procedure 2 in Appendix I.

Procedure 2 Merging and Alignment Correction (MAC)

Input: *Latent sub-trees* $\mathcal{N}_i$ for all internal nodes i.

Output: Global latent tree T structure and parameters.

for $\mathcal{N}_i$ and $\mathcal{N}_j$ in all the sub-trees **do**
  if there are common nodes between $\mathcal{N}_i$ and $\mathcal{N}_j$ **then**
    Find the shortest path$(v_i, v_j; \mathcal{N}_i)$ between v_i and v_j on $\mathcal{N}_i$ and path$(v_i, v_j; \mathcal{N}_j)$ in $\mathcal{N}_j$;
    Union the only conflicting path$(v_i, v_j; \mathcal{N}_i)$ & path$(v_i, v_j; \mathcal{N}_j)$ according to equation (5);
    Attach other nodes in $\mathcal{N}_i$ and $\mathcal{N}_j$ to the union path;
    Perform alignment correction as described in Procedure 3.

Parameter Alignment Correction: As mentioned before, our parameter estimation is unsupervised, and therefore, columns of the estimated transition matrices may be permuted across different triplets over which tensor decomposition is carried out. The parameter estimation within the triplet is automatically acquired through the tensor decomposition technique, so that the alignment issue only arises across triplets. We refer to this as the alignment issue and it appears at various levels.

There are two types of triplets, namely, *in-group* and *out-group* triplets. A triplet of nodes Trip(y_i, y_j, y_l) is said to be *in-group* (denoted by $\text{Trip}_{\text{in}}(y_i, y_j, y_l)$) if its containing nodes share a joint node h_k and there are no other hidden nodes in path(y_i, h_k), path(y_j, h_k) or path(y_l, h_k). Otherwise, this triplet is *out-group* denoted by $\text{Trip}_{\text{out}}(y_i, y_j, y_l)$. We define a group as *sufficient children* group if it contains at least three *in-group* nodes.

Procedure 3 Parameter Alignment Correction
($\mathbb{G}_r$ denotes reference group, $\mathbb{G}_o$ denotes the list of other groups, each group has a reference node denoted as $\mathcal{R}_l$, and the reference node in $\mathbb{G}_r$ is $\mathcal{R}_g$. The details on alignment at line 8 is in Appendix I.)

Input: Triplets and unaligned parameters estimated for these triplets, denoted as $\text{Trip}(y_i, y_j, y_k)$.

Output: Aligned parameters for the entire latent tree T.

1: Select $\mathbb{G}_r$ which has *sufficient children*;
2: Select refer node $\mathcal{R}_g$ in $\mathbb{G}_r$;
3: **for all** a, b in $\mathbb{G}_r$ **do**
4: Align $\text{Trip}_{\text{in}}(y_a, y_b, \mathcal{R}_g)$;
5: **for all** i_g in $\mathbb{G}_o$ **do**
6: Select refer node $\mathcal{R}_l$ in $\mathbb{G}_o[i_g]$;
7: Align $\text{Trip}_{\text{out}}(\mathcal{R}_g, y_a, \mathcal{R}_l)$ and $\text{Trip}_{\text{out}}(\mathcal{R}_l, y_i, \mathcal{R}_g)$;
8: **for all** i, j in $\mathbb{G}_o[i_g]$ **do**
9: Align $\text{Trip}(y_i, y_j, \mathcal{R}_l)$;

Designing *in-group* alignment correction with *sufficient children* is relatively simple: we achieve this by including a local reference node for all the *in-group* triplets. Thus, all the triplets are aligned with the reference node. The alignment correction is more challenging if lacking *sufficient children*. We propose *out-group* alignment to solve this problem. We first assign one group as a *reference group*, and the *local reference node* in that *reference group* becomes the *global reference node*. In this way, we align all recovered transition matrices in the same order of hidden states as in the reference node. See Procedure 2 and 3 for merging the local structures and align the parameters from LRG local sub-trees.

7 THEORETICAL GUARANTEES

Correctness of Proposed Parallel Algorithm: We now provide the main result of this paper on global consistency for our method.

Theorem 7.1 (Sample Complexity)**.** *Given samples from an identifiable latent tree model, the proposed method consistently recovers the structure with $O(\log p)$ sample complexity and parameters with $O(\log k)$ sample complexity, where p is the number of nodes and k is the dimension of hidden variable which is usually small.*

The proof sketch is in Appendix G. Lemma 7.2 and Lemma K.1 are used to prove Theorem 7.1. Note that the sample complexity of our algorithm is dimension independent, therefore easily scalable to large d.

In order to understand the correctness of the structure learning part of our algorithm, the following Lemma 7.2 states a guaranteed consistency of the structure learning.

Lemma 7.2 (Consistency of Structure Learning)**.** *Let $\widehat{\mathcal{T}}^N$ be our estimated tree using N number of examples, then for every $\eta > 0$, if $N > C\log(p/\eta^{1/3})$ for some constant $C > 0$, the error probability for structure reconstruction is upper bounded by η, i.e.,*

$$\Pr(h(\widehat{\mathcal{T}}^N \neq \mathcal{T})) \leq \eta \tag{6}$$

where h is a graph homomorphism – a mapping between graphs that respects their structure.

The multivariate information distance introduced in our paper enjoys the statistical efficiency in Anandkumar et al. [2011] as both methods require "enough" samples to get a confident estimation of the additive distance, which involves spectral decomposition of the empirical covariance. Although there seems to be a log difference, their algorithm requires "multiplications" of the metrics when testing quarterts, whereas we use "summations" of the metrics. Both analyses result in the same sample complexity.

Lemma K.1 guarantees consistency of the parameter learning part of our algorithm. Tensor decomposition guarantees consistent estimation of the parameters with $O(\log k)$ examples, one of the key contributions compared to Choi et al. [2011] which uses EM.

Computational Complexity: Recall d is the observable node dimension, k is the hidden node dimension ($k \ll d$), N is the number of samples, p is the number of observable nodes, and z is the number of non-zero elements in each sample. Let Γ denote the maximum size of the groups, over which we operate the local recursive grouping procedure. Thus, Γ affects the degree of parallelism for our method. Recall that it is given by the neighborhoods on MST, i.e., $\Gamma := \max_i |\text{nbd}[i; \text{MST}]|$. Below, we provide a bound on Γ.

Lemma 7.3. *The maximum size of neighborhoods on MST, denoted as Γ, satisfies $\Gamma \leq \Delta^{1+\frac{u_d}{l_d}\delta}$, where $\delta := \max_i\{\min_j\{path(v_i, v_j; \mathcal{T})\}\}$ is the effective depth, Δ is the maximum degree of $\mathcal{T}$, and the u_d and l_d are the upper and lower bound of information distances between neighbors on $\mathcal{T}$.*

Thus, we see that for many natural cases, where the degree and the depth in the latent tree are bounded (e.g. the hidden Markov model), and the parameters are mostly homogeneous (i.e., u_d/l_d is small), the group sizes are bounded, leading to a high degree of parallelism. We summarize the computational complexity in Table 1. Details can be found in Appendix J.

The reason we emphasize parallel complexity is to show the high "degree of parallelism" of our algorithm, one of

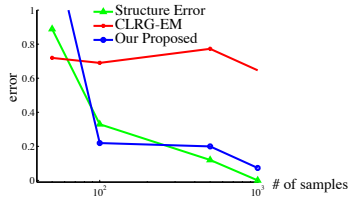

Figure 3: Comparison of structure and parameters recovery error between CLRG structure learning with EM parameter estimation and our algorithm. Structure errors (green) are the same for the two approaches. Proposed (blue) parameter estimation performs better than EM (red) ($d = 2, p = 9$).

the main advantages of our algorithm over a sequential implementation. Given two algorithms with the same serial complexity, we prefer the one with higher degree of parallelism as it could be made more efficient.

Algorithm Step	Time per worker	Degree of parallelism
Distance Est.	$O(Nz + d + k^3)$	$O(p^2)$
MST	$O(\log p)$	$O(p^2)$
LRG	$O(\Gamma^3)$	$O(p/\Gamma)$
Tensor Decomp.	$O(\Gamma k^3 + \Gamma dk^2)$	$O(p/\Gamma)$
Merging step	$O(dk^2)$	$O(p/\Gamma)$

Table 1: Worst-case computational complexity of our algorithm. The total complexity is the product of the time per work and degree of parallelism.

8 EXPERIMENTS

Synthetic experiments. We compare our method with CLRG-EM [Choi et al., 2011], where a sequential learning procedure is carried out for binary variables ($d = 2$) and parameter learning is carried out via EM. We consider a latent tree model over nine observed variables and four hidden nodes. We restart EM 20 times, and select the best result. The results are in Figure 3. We measure the structure recovery error via $\left\{ \sum_{i=1}^{|\widehat{G}|} \min_j |\widehat{G}_i \notin G_j| / |\widehat{G}_i| \right\} / |\widehat{G}|$, where G and $\widehat{G}$ are the ground-truth and recovered categories. We measure the parameter recovery error via $\mathcal{E} = \|A - \hat{A}\Pi\|_F$, where A is the true parameter and $\hat{A}$ is the estimated parameter and Π is a suitable permutation matrix that aligns the columns of $\hat{A}$ with A so that they have minimum distance. Π is greedily calculated. It is shown that as number of samples increases, both methods recover the structure correctly, as predicted by the theory. However, EM is stuck in local optima and fails to recover the true parameters, while the tensor decomposition correctly recovers the true parameters. We present the high performance of our algorithm on the large p and d regime where CLRG-EM is slow and easily stuck in local optima, in Table 2. We achieve efficient running times with good accuracy for structure and parameter estimation.

d	p	N	Struct Error	Param Error	Running Time(s)
10	9	50K	0	0.0104	3.8
100	9	50K	0	0.0967	4.4
1000	9	50K	0	0.1014	5.1
10,000	9	50K	0	0.0917	29.9
100,000	9	50k	0	0.0812	56.5
100	9	50K	0	0.0967	10.9
100	81	50K	0.06	0.1814	323.7
100	729	50K	0.16	0.1913	4220.1

Table 2: Algorithm performance in large d and p regime where CLRG-EM is not amendable and in large p regime where CLRG-EM is slow and easily stuck in local optima.

Real data experiments on NIPS and NY Times. Real datasets experiments which estimate a hierarchical structure over words are implemented and presented in Appendix C Figure 4, 5 and 6. The relationships among the words discovered by our algorithm match intuition. For example in Figure 6, govern and secur are grouped together whereas movi, studio and produc are clustered.

Healthcare data analysis. We demonstrate that our algorithm works for challenging tasks such as healthcare analytics in Appendix D. Our goal is to discover a disease hierarchy based on their co-occurring relationships in the patient records. In general, longitudinal patient records store the diagnosed diseases on patients over time, where the diseases are encoded with International Classification of Diseases (ICD) code. We use two large patient datasets (**MIMIC2** and **CMS**) of different sizes with respect to the number of samples, variables and dimensionality. The details of the datasets are discussed in Appendix D.1. The goal is to learn the latent nodes and the disease hierarchy and associated parameters from data. We validate the resulting disease hierarchy both quantitatively and qualitatively, and verify the scalability of our algorithm in Appendix D.

Acknowledgement

Huang is supported by startup fund from Department of Computer Science, University of Maryland, National Science Foundation IIS-1850220 CRII Award 030742-00001, and Adobe, Capital One and JP Morgan faculty fellowships. Sun is supported by the National Science Foundation award IIS-1418511, CCF-1533768 and IIS-1838042, the National Institute of Health award 1R01MD011682-01 and R56HL138415. Anandkumar is supported in part by Bren endowed chair, Darpa PAI, Raytheon, and Microsoft, Google and Adobe faculty fellowships.

References

A. Anandkumar, K. Chaudhuri, D. Hsu, S. M. Kakade, L. Song, and T. Zhang. Spectral methods for learning multivariate latent tree structure. *arXiv preprint arXiv:1107.1283*, 2011.

A. Anandkumar, D. P. Foster, D. Hsu, S. M. Kakade, and Y.-K. Liu. Two svds suffice: Spectral decompositions for probabilistic topic modeling and latent dirichlet allocation. *CoRR, abs/1204.6703*, 1, 2012a.

A. Anandkumar, R. Ge, D. Hsu, S. M. Kakade, and M. Telgarsky. Tensor decompositions for learning latent variable models. *arXiv preprint arXiv:1210.7559*, 2012b.

A. Anandkumar, V. Y. F. Tan, F. Huang, and A. S. Willsky. High-dimensional structure learning of Ising models: local separation criterion. *The Annals of Statistics*, 40(3):1346–1375, 2012c.

A. Anandkumar, R. Valluvan, et al. Learning loopy graphical models with latent variables: Efficient methods and guarantees. *The Annals of Statistics*, 41(2): 401–435, 2013.

D. A. Bader and G. Cong. Fast shared-memory algorithms for computing the minimum spanning forest of sparse graphs. *Journal of Parallel and Distributed Computing*, 66(11):1366–1378, 2006.

J. T. Chang. Full reconstruction of markov models on evolutionary trees: identifiability and consistency. *Mathematical biosciences*, 137(1):51–73, 1996.

J. T. Chang and J. A. Hartigan. Reconstruction of evolutionary trees from pairwise distributions on current species. In *Computing science and statistics: Proceedings of the 23rd symposium on the interface*, pages 254–257. Interface Foundation Fairfax Station, VA, 1991.

M. Choi, A. Torralba, and A. Willsky. Context models and out-of-context objects. *Pattern Recognition Letters*, 2012a.

M. J. Choi, V. Y. Tan, A. Anandkumar, and A. S. Willsky. Learning latent tree graphical models. *The Journal of Machine Learning Research*, 12:1771–1812, 2011.

M. J. Choi, A. Torralba, and A. S. Willsky. Context models and out-of-context objects. *Pattern Recognition Letters*, 33(7):853–862, 2012b.

K. L. Clarkson and D. P. Woodruff. Low rank approximation and regression in input sparsity time. In *Proceedings of the 45th annual ACM symposium on Symposium on theory of computing*, pages 81–90. ACM, 2013.

L. Dagum and R. Menon. Openmp: an industry standard api for shared-memory programming. *Computational Science & Engineering, IEEE*, 5(1):46–55, 1998.

J. Denny, M. Ritchie, M. Basford, J. Pulley, L. Bastarache, K. Brown-Gentry, D. Wang, D. Masys, R. DM, and D. Crawford. Phewas: demonstrating the feasibility of a phenome-wide scan to discover gene–disease associations. *Bioinformatics*, 26 (9):1205–1210, 2010.

R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge Univ. Press, 1999.

P. L. Erdos, M. A. Steel, L. A. Székely, and T. J. Warnow. A few logs suffice to build (almost) all trees (i). *Random Structures and Algorithms*, 14(2):153–184, 1999.

A. Falanga, M. Marchetti, A. Vignoli, and D. Balducci. Clotting mechanisms and cancer: implications in thrombus formation and tumor progression. *Clinical advances in hematology & oncology: H&O*, 1(11): 673–678, 2003.

A. V. Gerbessiotis and L. G. Valiant. Direct bulk-synchronous parallel algorithms. *Journal of parallel and distributed computing*, 22(2):251–267, 1994.

A. Gittens and M. W. Mahoney. Revisiting the nystrom method for improved large-scale machine learning. *CoRR*, abs/1303.1849, 2013a.

A. Gittens and M. W. Mahoney. Revisiting the nystrom method for improved large-scale machine learning. *arXiv preprint arXiv:1303.1849*, 2013b.

U. C. S. W. Group et al. United states cancer statistics: 1999–2010 incidence and mortality web-based report. *Atlanta (GA): Department of Health and Human Services, Centers for Disease Control and Prevention, and National Cancer Institute*, 2014.

F. Huang, N. U. N, M. U. Hakeem, P. Verma, and A. Anandkumar. Fast detection of overlapping communities via online tensor methods on gpus. *CoRR*, abs/1309.0787, 2013.

A. Krishnamurthy, S. Balakrishnan, M. Xu, and A. Singh. Efficient active algorithms for hierarchical clustering. *arXiv preprint arXiv:1206.4672*, 2012.

N. Meinshausen and P. Bühlmann. High dimensional graphs and variable selection with the lasso. *Annals of Statistics*, 34(3):1436–1462, 2006.

Michael. Boruvka algorithm parallel implementation cuda, December 2012. URL `http://qnundrum.com/answer.php?q=340303`.

E. Mossel. Distorted metrics on trees and phylogenetic forests. *IEEE/ACM Transactions on Computa-*

tional Biology and Bioinformatics (TCBB), 4(1):108–116, 2007.

E. Mossel and S. Roch. Learning nonsingular phylogenies and hidden markov models. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 366–375. ACM, 2005.

A. P Parikh, L. Song, and E. P Xing. A spectral algorithm for latent tree graphical models. 2011.

J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann, 1988.

D. Robinson and L. R. Foulds. Comparison of phylogenetic trees. *Mathematical Biosciences*, 53(1):131–147, 1981.

A. Smola and S. Narayanamurthy. An architecture for parallel topic models. *Proceedings of the VLDB Endowment*, 3(1-2):703–710, 2010.

L. Song, E. P. Xing, and A. P. Parikh. Kernel embeddings of latent tree graphical models. In *Advances in Neural Information Processing Systems*, pages 2708–2716, 2011.

L. Song, H. Liu, A. Parikh, and E. Xing. Nonparametric latent tree graphical models: Inference, estimation, and structure learning. *arXiv preprint arXiv:1401.3940*, 2014.

V. Vineet, P. Harish, S. Patidar, and P. Narayanan. Fast minimum spanning tree for large graphs on the gpu. In *Proceedings of the Conference on High Performance Graphics 2009*, pages 167–171. ACM, 2009.

F. Wang and Y. Li. Beyond physical connections: Tree models in human pose estimation. In *Proc. of CVPR*, 2013.

J. Wei, W. Dai, A. Kumar, X. Zheng, Q. Ho, and E. P. Xing. Consistent Bounded-Asynchronous Parameter Servers for Distributed ML. *ArXiv e-prints*, Dec. 2013.