Likelihood Based Hierarchical Clustering

R. M. Castro*, *Student Member, IEEE,* M. J. Coates, *Member, IEEE*
R. D. Nowak, *Member, IEEE,*

Abstract

This paper develops a new method for hierarchical clustering. Unlike other existing clustering schemes, our method is based on a generative, tree-structured model that represents relationships between the objects to be clustered, rather than directly modeling properties of objects themselves. In certain problems, this generative model naturally captures the physical mechanisms responsible for relationships among objects, for example, in certain evolutionary tree problems in genetics and communication network topology identification. The paper examines the networking problem in some detail, to illustrate the new clustering method. More broadly, the generative model may not reflect actual physical mechanisms, but it nonetheless provides a means for dealing with errors in the similarity matrix, simultaneously promoting two desirable features in clustering: intra-class similarity and inter-class dissimilarity.

EDICS Category: 2-LEAR

Corresponding author: R. M. Castro. R. M. Castro and R. D. Nowak are with the Department of Electrical and Computer Engineering, Rice University, MS366, Houston, TX 77251-1892 USA (e-mail: rcastro@rice.edu, phone: + 1 608 263 1557, fax: + 1 608 262 1267). M. J. Coates is with the Department of Electrical and Computer Engineering, McGill University, 3480 University Street, Montreal, QC, Canada H3A-2A7. This work was partially supported by the National Science Foundation, grant nos. CCR-0310889 and ANI-0099148, the Office of Naval Research, grant no. N00014-00-1-0390, the Army Research Office, grant no. DAAD19-99-1-0290, and the State of Texas ATP, grant no. 003604-0064-2001.

Likelihood Based Hierarchical Clustering

I. Introduction

A clustering algorithm is a process designed to organize a set of objects into various classes, such that objects within the same class share certain characteristics. In many cases it is desirable to perform this task without user supervision. Let S denote a set of objects, called input objects. While for some clustering problems, the goal is to partition the set of input objects into disjoint classes (*k*-clustering), for some other problems one desires to obtain a hierarchical structure, where each class of objects is also partitioned into sub-classes and so on. In this latter case, one can represent the clustering of objects as a tree, also called a *dendrogram*, in which the nodes represent subsets of the input set S. The leaf nodes correspond to the individual elements of S, and the root corresponds to the entire set. Each edge in the dendrogram represents an inclusion relationship (see Figure 1 for illustration). This paper develops a new method for hierarchical clustering based on a generative dendritic cluster model. The objects are viewed as being generated by a tree-structured refinement process. This process models the similarities between each pair of objects, rather than the properties of the individual objects. In certain problems, this generative model naturally captures the physical mechanisms responsible for relationships among objects, for example, in evolutionary trees and network topology identification. The latter problem was the main motivation for the framework introduced by this paper, and is examined in detail in Section VI. In broader settings, the generative model is not reflective of actual physical mechanisms, but it nonetheless provides a means for dealing with measurement errors and simultaneously promotes two desirable features in clustering: intra-class similarity and inter-class dissimilarity. For example, in protein classification, it is difficult to discern an underlying generative model, but the presented technique provides a method for addressing errors in similarity measurements.

The new clustering method is based on a maximum likelihood framework. Associated with the set of objects S is an $|S| \times |S|$ matrix $\mathbf{X}$ of estimated pairwise similarity measures between objects. We assume that the empirical similarity matrix $\mathbf{X}$ is related to an ideal (or "true") similarity matrix $\boldsymbol{\gamma}$ through a probability density function that describes the possible errors or distortions in $\mathbf{X}$. That is, $\mathbf{X} \sim p(\mathbf{x}|\boldsymbol{\gamma})$. As a simple example, $p(\mathbf{x}|\boldsymbol{\gamma})$ might be a Gaussian density with mean $\boldsymbol{\gamma}$ and diagonal covariance σ^2I, in which case $\mathbf{X}$ can be regarded as a "noisy" version of $\boldsymbol{\gamma}$. In general, $\mathbf{X}$ may be related to $\boldsymbol{\gamma}$ in a much more complicated manner. The similarity matrix $\boldsymbol{\gamma}$ must correspond to a dendritic structure, but is otherwise unknown and unconstrained. The likelihood is also a function of the unknown dendritic tree T, governing the structure of γ, and it is our main subject of interest. We present deterministic and Monte Carlo approaches to estimate the underlying dendritic tree T. We consider also maximum penalized likelihood approaches that control the complexity of the tree estimate, preventing the selection of a tree that fits the particular realization of the measurements instead of the true underlying tree structure.

Hierarchical clustering is a widely used approach, which has a long history [1]–[5] and is especially popular for document clustering [6]–[9]. Most approaches to hierarchical clustering are agglomerative algorithms that follow a simple methodology [10], and proceed by repeatedly applying four steps: (i) choose the pair of nodes with the highest similarity; (ii) merge the pair into a new node/cluster; (iii) update the similarities between the new node and the former existing nodes; and (iv) repeat the procedure until only one node is left. The crucial step is the update of the similarity values. The nature of the update is determined through the specification of *linkage metrics*, which embody the closeness (or connectedness) of subsets of nodes. Much of the work in hierarchical clustering research revolves around the derivation of effective linkage metrics from the similarity matrix **X**. The choice of linkage metric has a very strong influence on the resultant clustering, so it significantly impacts clustering performance. The suitability of a specific linkage metric depends on the problem at hand. In our model-based approach this issue is less arbitrary since the linkage metrics are induced from the assigned probability density $p(\mathbf{x}|\gamma, T)$. In Section V-A we present an algorithm following the agglomerative framework, where the linkage metric arises from the probability model. For the simple case of the Gaussian model mentioned above (with a diagonal covariance matrix σ^2I) this leads essentially to the well known Unweighted Pair-Group Average Method (UPGMA) [10].

Other model-based approaches have been used for hierarchical clustering in the past [11]–[14]. Such methods usually model the clusters directly, as Gaussian components, for example. The cluster models induce a distance measure. The work of Banfield and Raftery [5] is representative of this strategy. In contrast, our approach is based on a generative model of the similarity matrix **X** rather than a model of the objects in the clusters. Therefore, the objects to be clustered do not need to be directly embedded in a metric space. This allows us to express possible errors or uncertainties in our similarity measures in a simple manner. We are not aware of previous work that has adopted this approach. Moreover, in certain problems such a formulation is physically motivated and more natural than devising models for the individual classes. This is particularly true if there is an underlying hierarchical structure.

This paper is organized as follows: Sections II and III introduce the problem and the likelihood framework. Section IV provides some important characterizations and key properties of our framework. In Section V two algorithms seeking the solution of our problem are introduced; one more deterministic and greedy in nature, and the other based on a random search technique. In Section VI we present a practical and relevant application of the concept and algorithms introduced previously. The section includes some simulation and experimental results. Finally, in Section VII, we make some concluding remarks. The proofs of various results in the paper are presented in Section VIII.

II. Problem Statement

A hierarchical clustering of a set of objects can be described as a tree, in which the leaves are precisely the objects to be clustered. An example is depicted in Figure 1 and it is used to clarify the concepts introduced below. In our formulation we focus primarily on the tree representation.

Let $T = (V, L)$ denote a rooted tree with nodes V and directed links L (we consider a strongly acyclic graph, that is, if we disregard the direction of the edges the graph is a tree). Denote the root node by `Root`. Denote by S the leaf nodes. Each leaf node corresponds to one object in the input set S, and the root node corresponds to a cluster encompassing all input objects. For example, in Figure 1, $S = \{4, 6, 7, 8, 9\}$.

Every node has at least two descendants, apart from the leaf nodes, which have none. If all internal nodes have exactly two descendants then the tree is called binary. For each node $i \in V$ let $f(i)$ denote the parent of i, *e.g.*, $f(8) = 5$. We can identify each link with the corresponding end node, *i.e.*, $(f(i), i) \sim i,\ (f(i), i) \in L$. Let $a(i, j),\ i, j \in V$, denote the nearest ancestor of the pair of nodes (i, j), *e.g.*, $a(4, 9) = 2$. We define also $c(i),\ i \in V$, as the set of children nodes of i, *e.g.*, $c(2) = \{4, 5\}$. For a given node k we denote by $S(k)$ the subset of leaves S with k as an ancestor. This set corresponds to the elements in the cluster represented by node k, *e.g.*, $S(2) = \{4, 8, 9\}$.

For each pair of input objects we can consider a similarity metric value. This value reflects how similar (with respect to certain characteristics) the two objects are (the larger the value, the more similar they are). Let γ_{ij} denote the similarity between two input objects $i, j \in S$. The goal of clustering is to group inputs together that have the large pairwise similarities and to simultaneously separate inputs with lesser similarities. We refer to each such group as a cluster.

Two key notions in clustering are the inter-cluster and intra-cluster similarities. We define the inter-cluster similarities of two disjoint clusters as the similarities of pairs of objects where one element in the pair belongs to one cluster and the other element belongs to the other cluster. For a single cluster, we define the intra-cluster similarities as the pairwise similarities of objects within the cluster. A good clustering is one in which the inter-cluster similarities between disjoint clusters are small, compared to the intra-cluster similarities of each individual cluster. Ideally, the inter-cluster similarities between two disjoint clusters are identical; *i.e.*, two clusters share the same similarity value between each other, for example, in the clustering depicted in Figure 1, γ_{46} and γ_{47} are assumed to be identical (this corresponds to the inter-cluster similarities of clusters $\{4\}$ and $\{6, 7\}$). However, in practice, this notion of identical similarity between clusters is not generally met by the empirical similarities $\{x_{ij}\}$ for several reasons. First and foremost, the hierarchical cluster structure is typically an approximation of the physical relationships between objects, and the actual relationships may not strictly adhere to the hierarchical structure. Secondly, the measured similarities may be contaminated with errors or may be biased due to limitations of the measurement system. For example, in Figure 1, the measured similarities x_{46} and x_{47} (defined formally in the next section) may not be equal to one another.

In our model, enforcing identical inter-cluster similarities between disjoint clusters is accomplished as follows. Consider an arbitrary tree $T = (V, L)$. With each node in the tree we associate a metric value $\gamma_k \in \mathbb{R},\ k \in V$. This metric value provides a measure of the similarity of the elements in $S(k)$. For each pair of objects $i, j \in S$ we require the pairwise metric value γ_{ij} to be the metric value of the nearest ancestor, that is $\gamma_{ij} \equiv \gamma_{a(i,j)}$. Note that this enforces symmetry of the pairwise metrics values (*i.e.*, $\gamma_{ij} = \gamma_{ji}$). The value γ_{ij} can be regarded as a characterization of the shared portion of the paths from the root to i and j. The shared path for a pair of nodes (i, j) is the path from the root to node $a(i, j)$. Notice that the tree structure imposes restrictions on γ_{ij}; for example, if two different pairs of nodes have the same shared paths then their similarity values must be the same, *e.g.*, consider the tree in Figure 1; The pairs of nodes $(6, 8)$ and $(4, 7)$ have the same shared paths, thus $\gamma_{68} = \gamma_{47}$.

Define the matrix $\boldsymbol{\gamma} = (\gamma_{ij} : i, j \in S)$. According to the discussion above, in order for $\boldsymbol{\gamma}$ to be compatible with the tree structure it must belong to the set

$$\boldsymbol{\Gamma}(T) \equiv \{\boldsymbol{\gamma} : \gamma_{ij} = \gamma_{kl} \text{ if } a(i,j) = a(k,l)\} \ . \tag{1}$$

Clearly matrices in this set are not only symmetric, but have a more constrained structure. We disregard node self-similarities γ_{ii}, because these do not provide relevant information for the clustering task.

To ensure identifiability of the tree T (that is, to ensure we can fully recover the tree given the set of pairwise metrics) we require the metrics $\boldsymbol{\gamma}$ to satisfy the *monotonicity property*, which requires that $\gamma_{f(k)} < \gamma_k$ for any internal node k in the tree ($k \in V \setminus \{S, \texttt{Root}\}$). This property has a very simple graphical interpretation: deeper nodes (the leaves are the deepest nodes) have greater similarity values. From a clustering perspective this enforces that the inter-cluster similarities between two disjoint clusters are small, compared to the intra-cluster similarities of each one of the two clusters. We will see shortly that knowledge of the metric values for each pair of elements in S and the enforcement of the monotonicity property are sufficient for identification of the underlying tree topology.

For a given tree T, the set of all metrics satisfying the monotonicity property, denoted the *monotonic metric subset*, is defined as

$$\mathcal{G}(T) \equiv \left\{ \boldsymbol{\gamma} \in \boldsymbol{\Gamma}(T) : \gamma_{f(k)} < \gamma_k, \ \forall k \in V \setminus \{\texttt{Root}, S\} \right\} \ . \tag{2}$$

Given the pairwise similarities $\boldsymbol{\gamma}$ characterizing the input set, it is possible to reconstruct the dendritic tree. For example, referring to Figure 1, the metric γ_{89} will be greater than γ_{i9} for all $i \in S \setminus \{8, 9\}$, revealing that nodes 8 and 9 have a common parent in the tree. Using a simple agglomerative bottom-up procedure, following the same conceptual framework as many hierarchical clustering methods [1], [4], one can recover the underlying tree. The following result, appearing in a networking context in [15], ensures that for a similarity matrix $\boldsymbol{\gamma}$ satisfying the monotonicity property, the set of pairwise similarities completely determines the tree.

Proposition 1: Let T be a tree topology with object set S. Let $\{\gamma_{ij}\}$ be the set of pairwise similarities, corresponding to a monotonic metric on T. Then T is the only tree with pairwise similarities $\{\gamma_{ij}\}$ satisfying the monotonicity property. That is $\boldsymbol{\gamma} \in \mathcal{G}(T)$, and $\boldsymbol{\gamma} \not\subseteq \mathcal{G}(T'), \ \forall \ T' \neq T$.

The proof of the result follows by constructing the agglomerative bottom-up algorithm suggested above, and can be found in [15].

The goal of our work is to determine an optimal hierarchical clustering tree with respect to the probability $p(\mathbf{x}|\boldsymbol{\gamma})$. Our optimality criterion will be specified in the next section, after introducing some basic notation, terminology and concepts.

III. Likelihood formulation

Recall that **X** denotes the empirical similarity matrix. These are the accessible measurements, which convey information about $\boldsymbol{\gamma}$ and T. We consider the empirical similarities to be imperfect, possibly contaminated with errors, or only crude reflections of the true similarities between objects. We regard **X** as a random process parameterized by $\boldsymbol{\gamma}$ and T.

For a given unknown tree T let $\mathbf{X} \equiv \{X_{ij} : i, j \in S, \ i \neq j\}$, where each X_{ij} is a random variable parameterized by $\boldsymbol{\gamma} \equiv \{\gamma_{ij}\} \in \mathcal{G}(T)$. Let $p(\mathbf{x}|\boldsymbol{\gamma})$ denote the probability density function of **X**, parameterized by $\boldsymbol{\gamma}$. A sample $\mathbf{x} \equiv \{x_{ij} : i, j \in S, \ i \neq j\}$ of **X** is observed. Notice that, in general, $x_{ij} \neq x_{ji}$. When $p(\mathbf{x}|\boldsymbol{\gamma})$ is viewed as a function of T and $\boldsymbol{\gamma} \in \mathcal{G}(T)$ it is called the likelihood of T and $\boldsymbol{\gamma}$. The maximum likelihood tree estimate is given by

$$T^* \ = \arg\max_{T \in \mathcal{F}} \sup_{\boldsymbol{\gamma} \in \mathcal{G}(T)} p(\mathbf{x}|\boldsymbol{\gamma}) \ , \tag{3}$$

where $\mathcal{F}$ denotes the *forest* of all possible trees with leaves S. If the maximizer of the above expression is not unique define T^* as one of the possible maximizers. In many situations we are not primarily interested in $\widehat{\boldsymbol{\gamma}}(\mathbf{x})$, an estimate of $\boldsymbol{\gamma}$ from the measurements, hence we can regard $\boldsymbol{\gamma}$ as a nuisance parameter. In that case (3) can be interpreted as a maximization of the profile likelihood [16]

$$\mathcal{L}(\mathbf{x}|T) \equiv \sup_{\boldsymbol{\gamma} \in \mathcal{G}(T)} p(\mathbf{x}|\boldsymbol{\gamma}) \; . \tag{4}$$

The solution of (3) is referred to as the Maximum Likelihood Tree (MLT). Searching for this tree is our attempt to find the tree associated with the unknown pairwise metrics $\boldsymbol{\gamma}$. The remainder of the paper presents various techniques and characterizations of the likelihood structure that lead us in that direction.

In our model, we also assume that the probability density has a diagonal structure. Specifically, the random variables X_{ij} are independent and have densities $p(x_{ij}|\gamma_{ij})$, $i,j \in S$, $i \neq j$. This assumption is somewhat restrictive (e.g., the observed similarities cannot have correlated errors), but we feel it is not unreasonable in many applications, such as the networking problem we consider later. Moreover, without a diagonal structure it may not be possible to base the clustering process on local algorithms, like the common agglomerative, bottom-up procedure mentioned in the previous section.

To simplify our presentation, denote the log-likelihood by $h_{ij}(x_{ij}|\gamma_{ij}) = \log p(x_{ij}|\gamma_{ij})$. We will also assume that $h_{ij}(x_{ij}|\gamma_{ij})$ is a strictly concave functional of γ_{ij} having a maximizer in $\mathbb{R}$ (note that the maximizer is unique since the function is strictly concave). The log-likelihood is hence

$$\log p(\mathbf{x}|\boldsymbol{\gamma}) = \sum_{i \in S} \sum_{j \in S \setminus \{i\}} h_{ij}(x_{ij}|\gamma_{ij}) \; . \tag{5}$$

The assumption of concavity is strong, but not uncommon in most familiar probability models. Furthermore, note that, although parameterized by a common parameter γ_{ij}, the log-densities h_{ij} and h_{ji} are not necessarily the same (as in the example in Section VI), so the two measurements x_{ij} and x_{ji} are different in nature, and convey different information.

IV. Characterization of the Maximum Likelihood Structure

The optimization problem in (3) is quite formidable. We are not aware of any method for computation of the global maximum except by a brute force examination of each tree in the forest. Consider a tree with N leaves. A very loose lower bound on the size of the forest $\mathcal{F}$ is $N!/2$. For example, if $N = 10$ then there are more than 1.8×10^6 trees in the forest. This explosion of the search space precludes the brute force approach in all but very small forests. Moreover, the computation of the profile likelihood (4) is non-trivial because it involves a constrained optimization over $\mathcal{G}(T)$.

Using the model (5) we have that $p(\mathbf{x}|\boldsymbol{\gamma})$ is a continuous and bounded function of $\boldsymbol{\gamma}$ and hence we can rewrite the profile likelihood as

$$\mathcal{L}(\mathbf{x}|T) \equiv \max_{\boldsymbol{\gamma}\in\overline{\mathcal{G}(T)}} p(\mathbf{x}|\boldsymbol{\gamma}) \;, \tag{6}$$

where $\overline{\mathcal{G}(T)}$ denotes the closure of the set $\mathcal{G}(T)$. The profile likelihood can be computed by solving this constrained optimization problem, although the solution is still fairly involved. The following results establish some key properties for this problem. The next lemma characterizes a modified version of the profile likelihood (6) (the version of the profile likelihood in the lemma does not involve the monotonicity constraint, that is, the optimization is over $\boldsymbol{\Gamma}(T)$ instead of $\overline{\mathcal{G}(T)}$, the monotonic metric subset).

Lemma 1: Let T be an arbitrary tree. The solution of

$$\widehat{\boldsymbol{\gamma}} = \arg\max_{\boldsymbol{\gamma}\in\boldsymbol{\Gamma}(T)} p(\mathbf{x}|\boldsymbol{\gamma})$$

is unique and given by

$$\widehat{\gamma}_{ij} = \arg\max_{\gamma\in\mathbb{R}} \sum_{k,l\in S:a(k,l)=a(i,j)} h_{kl}(x_{kl}|\gamma) \;. \tag{7}$$

The proof of the lemma is elementary and it is presented in Section VIII-A. The evaluation of this version of the profile likelihood for a given tree is very simple, following from equation (7). This is a one-dimensional concave maximization problem that can be solved efficiently using simple numerical methods. For certain classes of models (like the one in Section VI) it can also be solved analytically.

The unconstrained optimization in the lemma above is easy to solve, and the following theorem (proved in Section VIII-B) shows that it is sufficient for the purposes of determining the MLT.

Theorem 1: Consider a fixed realization $\mathbf{x}$ of $\mathbf{X}$. Let $\log p(\mathbf{x}|\boldsymbol{\gamma})$ be given by (5) and $\widetilde{T}$ be a tree such that

$$\max_{\boldsymbol{\gamma}'\in\boldsymbol{\Gamma}(\widetilde{T})} p(\mathbf{x}|\boldsymbol{\gamma}') \;>\; \max_{\boldsymbol{\gamma}'\in\overline{\mathcal{G}(\widetilde{T})}} p(\mathbf{x}|\boldsymbol{\gamma}') \;. \tag{8}$$

Then there exists another tree $(T,\boldsymbol{\gamma})$, $\boldsymbol{\gamma}\in\mathcal{G}(T)$, satisfying the monotonicity property, such that

$$p(\mathbf{x}|\boldsymbol{\gamma}) \;>\; \max_{\boldsymbol{\gamma}'\in\overline{\mathcal{G}(\widetilde{T})}} p(\mathbf{x}|\boldsymbol{\gamma}') \tag{9}$$

In particular, if T^* is the solution to (3), *i.e.*, the MLT, we have

$$\arg\max_{\boldsymbol{\gamma}'\in\boldsymbol{\Gamma}(T^*)} p(\mathbf{x}|\boldsymbol{\gamma}') \;=\; \arg\max_{\boldsymbol{\gamma}'\in\overline{\mathcal{G}(T^*)}} p(\mathbf{x}|\boldsymbol{\gamma}') \tag{10}$$

Remark 1: Consider an arbitrary tree $\widetilde{T}$. Suppose that the maximum of $\log p(\mathbf{x}|\boldsymbol{\gamma})$ is attained for $\boldsymbol{\gamma} \in \boldsymbol{\Gamma}(\widetilde{T}) \setminus \overline{\mathcal{G}(\widetilde{T})}$, that is, expression (8) holds. The theorem implies that in that case there exists another tree $(T, \boldsymbol{\gamma})$, $\boldsymbol{\gamma} \in \overline{\mathcal{G}(T)}$, with a higher likelihood. Consequently the tree $\widetilde{T}$ cannot be the maximum likelihood tree (10).

Remark 2: The second part of the theorem (10) shows that it is unnecessary to perform the constrained optimization over $\mathcal{G}(T)$. For each tree, we can compute the much simpler optimization (over $\boldsymbol{\Gamma}(T)$), using Lemma 1, and check if the resulting maximizer lies in the set $\overline{\mathcal{G}(T)}$.

Define the set of trees

$$\mathcal{F}' = \left\{ T \in \mathcal{F} : \arg \max_{\boldsymbol{\gamma} \in \boldsymbol{\Gamma}(T)} p\left(\mathbf{x}|\boldsymbol{\gamma}\right) \in \overline{\mathcal{G}(T)} \right\} .$$

Note that the maximum likelihood tree belongs to this set, *i.e.*, $T^* \in \mathcal{F}'$. We call $\mathcal{F}'$ the *feasible forest*. Despite this characterization, a search over $\mathcal{F}'$ might still be too demanding.

Remark 3: Note that we can restrict our attention to binary trees, because the maximum of the likelihood function can always be achieved by a binary tree. To see this, simply notice that for any non-binary branching point (e.g., a node with three or more branches) there exists a sequence of binary branchings with the same (or larger) likelihood, since the extra branches provide extra degrees of freedom in fitting to the empirical similarities. Of course, it may be desirable to obtain a clustering with as few branches as possible, and this issue will be examined in the next section.

V. Hierarchical Clustering Algorithms

In this section we present two algorithms intended to solve the problem (3). Hierarchical clustering algorithms generally belong to one of two types: bottom-up, agglomerative constructions or top-down, divisive methods. Our probabilistic model for the similarity matrix allows us to develop a novel agglomerative algorithm based on the likelihood function. Furthermore, by viewing the (profile) likelihood function as a discrete probability mass function, we also devise a Metropolis-Hastings search strategy that circumvents the greedy search strategy associated with agglomerative and divisive algorithms.

A. Bottom-up Agglomerative Approach

At the end of Section II we noticed that the pairwise similarity values can be used to recover the underlying tree structure through a bottom-up merging procedure. In most practical scenarios, we only have access to the measurements $\mathbf{x}$, conveying information about $\boldsymbol{\gamma}$ (and hence about T). In this case we can still develop a bottom-up agglomerative clustering algorithm to estimate the true dendrogram.

As pointed out in *Remark 3*, we can restrict our attention to binary trees, and this restriction leads to a particularly simple algorithm for clustering. The algorithm proceeds by forming estimates of the pairwise similarities for each pair of leaf nodes:

$$\widehat{\gamma}_{ij} = \arg\max_{\gamma \in \mathbb{R}} \left(h_{ij}(x_{ij}|\gamma) + h_{ji}(x_{ji}|\gamma) \right) , \ i, j \in S, \ i \neq j \ .$$

One expects the above estimated pairwise similarities to be reasonably close to the true similarities $\boldsymbol{\gamma}$, with the differences being due to measurement noise and limitations of the measurement procedure.

Let i and j be the indices of the leaf nodes with the highest similarity, that is

$$\widehat{\gamma}_{ij} \geq \widehat{\gamma}_{lm}, \ \forall l, m \in S \ .$$

We infer that i and j are the most similar leaf nodes, implying that they have a common parent in the dendrogram. Denote their parent node by k. In other words, we are constructing the cluster $S(k) = \{i, j\}$, and node k in the dendrogram represents that cluster. Assuming that our decision is correct then the tree structure imposes the condition $a(i, l) = a(j, l)$ for all $l \notin \{i, j\}$. Hence we update our pairwise similarity estimates for pairs involving i and j, using Lemma 1. Furthermore, since $\widehat{\gamma}_{il} = \widehat{\gamma}_{jl}$ for any $l \notin \{i, j\}$, we can just add node k as a new leaf node, and remove i and j from the leaf set. Define the new leaf set $S' = S \bigcup \{k\} \setminus \{i, j\}$. We need to define pairwise similarity estimates for pairs involving the new node k:

$$\widehat{\gamma}_{kl} = \widehat{\gamma}_{lk} \equiv \arg\max_{\gamma \in \mathbb{R}} \sum_{r \in S(k)} h_{rl}(x_{rl}|\gamma) + h_{lr}(x_{lr}|\gamma) \ , \quad \text{where } l \in S' \setminus \{k\} \ .$$

This procedure is iterated until there is only one element left in S'. The algorithm, denoted Agglomerative Likelihood Tree (ALT), is detailed in Figure 2.

Notice that the number of elements of S' is decreased by one at each step of the algorithm, guaranteeing that the algorithm stops. The algorithm structure ensures similarity estimates $\widehat{\boldsymbol{\gamma}}$ satisfy the monotonicity property, with respect to the final tree (this follows from the algorithm structure and Lemma 2 in the Appendix). Notice also that the algorithm always yields a binary tree.

Before moving on, let us comment briefly on the binary tree restriction. Suppose that the underlying tree were non-binary, consisting of only the root node and three leaf nodes. Let the leaf nodes be denoted by 1, 2 and 3. Then $\gamma_{12} = \gamma_{13} = \gamma_{23}$. The estimates of the metric values are almost surely not going to be identical (since the density is continuous). Therefore, in this case we notice that the maximum likelihood tree must have one internal node (this should be clear from the proof of Theorem 1, as the estimated binary tree cannot match the underlying, non-binary tree. In conclusion, when the underlying tree is non-binary, the MLT will have extra links (not in the underlying tree), used to fit the data more accurately.

B. Asymptotic Performance of ALT

As pointed out before, one expects the estimated pairwise similarity values $\widehat{\boldsymbol{\gamma}}$ to be close to the true similarities $\boldsymbol{\gamma}$. In this section, we characterize the limiting behavior of ALT as the estimated similarities tend to the true similarities. Define

$$\widehat{\gamma}_{ij}(\mathbf{x}) \equiv \arg\max_{\gamma \in \mathbb{R}} h_{ij}(x_{ij}|\gamma) + h_{ji}(x_{ji}|\gamma) \ ,$$

where $i, j \in S$. Given the measurements **x**, $\widehat{\gamma}_{ij}(\mathbf{x})$ is the maximum likelihood estimate of the pairwise similarities γ_{ij} (without further knowledge of the tree topology). In various scenarios we can increase the accuracy of $\widehat{\gamma}_{ij}(\mathbf{x})$ by considering more elaborate similarity measurements, or considering multiple (roughly independent) measurements.

Let ρ "index" the accuracy of the estimates. For example, $\rho \in \mathbb{R}^+$ can represent a signal-to-noise ratio, governing the measurements, or $\rho \in \mathbb{N}$ can denote the number of independent measurements made for each input pair and averaged to compute the estimate. Suppose that $\widehat{\gamma}_{ij}(\mathbf{x})$ converges in probability to γ_{ij} as ρ tends to infinity.

Proposition 2: For a binary tree T the ALT algorithm is consistent, that is, if $\widehat{T}$ is the tree obtained by the ALT algorithm then

$$\lim_{\rho \to \infty} \Pr(T = \widehat{T}) = 1 \ .$$

Also, as ρ grows, the probability that this is the only binary tree in feasible forest $\mathcal{F}'$ converges to 1.
The proposition (proved in Section VIII-C) indicates that the ALT algorithm perfectly reconstructs the original binary tree, provided that one can estimate the similarity values with enough accuracy. The second part of the result characterizes the feasible forest $\mathcal{F}'$, and has the following important implication.

Proposition 3: For binary trees, the MLT is a consistent estimator of the true dendritic tree. That is, if T^* denotes the MLT then $\lim_{\rho\to\infty} \Pr(T = T^*) = 1$.

Proof: The proof of this result is elementary, since the maximum likelihood tree is binary and belongs to $\mathcal{F}'$, therefore, as ρ increases, the probability that the MLT is the only binary tree in $\mathcal{F}'$ converges to one, by Proposition 2. ■

For non-binary trees, neither the MLT nor the ALT are consistent, but nevertheless the inferred tree is "compatible" with the true underlying tree, in the sense that it has extra, superfluous links, that are used to fit the data more closely. The incremental differences, $\hat{\gamma}_k - \hat{\gamma}_{f(k)}$, in such cases will be typically small, and these extra links can be eliminated by using a simple thresholding procedure, collapsing the links with incremental difference smaller than a certain threshold. Of course, selecting a reasonable threshold is a tricky issue. Another possibility is to consider cliques of more than two nodes at each agglomeration step, which allows for non-binary trees. Such cliques are selected if all nodes in a clique have similarities within a small range, indicating that they are nearly equally similar to one another. This possibility requires the specification a tolerance or threshold to define the notion of a "small range". However, it is not clear which cliques should be agglomerated first. With either approach, deleting seemingly superfluous links or merging larger groups, the extension of ALT beyond binary models thus requires the specification of extra tuning parameters, and therefore we do not investigate this issue further. Instead, we advocate a Monte Carlo method that is capable of handling non-binary trees, and circumvents the greedy nature of ALT.

C. Markov Chain Monte Carlo Method

Although the ALT algorithm is a consistent estimator of the true dendrogram (when it is a binary tree), it is a greedy strategy, based on local decisions over the pairwise metric values. Unlike the ALT, the MLT estimator takes a global approach, seeking for the best (in a maximum likelihood sense) tree. The price to pay is that we now must search over the entire forest $\mathcal{F}$. We can simplify this search using Theorem 1, and make the search only over the set $\mathcal{F}'$, but this is still a very computationally demanding task. In this section we propose a random search technique to efficiently search the forest of trees.

Consider the profile likelihood $\mathcal{L}(\mathbf{x}|T)$, as defined in (6). The maximum likelihood tree was defined as

$$T^* \; = \arg\max_{T\in\mathcal{F}} \mathcal{L}(\mathbf{x}|T) \; . \tag{11}$$

For a fixed measurement **x** we can regard the profile likelihood $\mathcal{L}(\mathbf{x}|T)$ as a discrete distribution over $\mathcal{F}$ (up to a normalizing factor). Then, one way of searching the set $\mathcal{F}$ is to sample it according to this distribution. More likely trees are then sampled more often than the less likely trees, making the search more efficient. This approach can be viewed in a Bayesian framework, with the adoption of a prior on the forest of trees that renders each tree equally likely *a priori*.

Evaluation of the profile likelihood (6) is complicated, as pointed out before, since it involves a constrained optimization over the monotonic metric set $\mathcal{G}(T)$. As we observed in Section IV the MLT belongs to the set of feasible trees $\mathcal{F}'$. For trees in $\mathcal{F}'$ one can compute the profile likelihood very easily using Lemma 1. Also, given an arbitrary tree, one can easily verify if that tree belongs to $\mathcal{F}'$

by performing the less-constrained optimization (over $\mathbf{\Gamma}(T)$), using Lemma 1, and then checking if the resulting maximizer lies in the set $\overline{\mathcal{G}(T)}$. The main idea is then to perform the search over $\mathcal{F}'$, instead of $\mathcal{F}$. Define

$$\mathcal{L}'(\mathbf{x}|T) = \begin{cases} \mathcal{L}(\mathbf{x}|T) & \text{if } T \in \mathcal{F}' \\ 0 & \text{otherwise} \end{cases} . \tag{12}$$

The above expression can be evaluated much faster than (6). We can regard (12) as a distribution over $\mathcal{F}$ (up to a normalizing function) and proceed as before. Again, we can fit this approach in a Bayesian framework, where we now consider a prior such that any tree in the feasible set $\mathcal{F}'$ is equally likely, and trees outside this set have probability zero.

There are numerous ways of drawing samples from a distribution that is known up to a normalizing factor [17]. Here we outline an MCMC approach. The algorithm we present is relatively simple and serves mainly to illustrate the basic methodology. There are many strategies that can be applied to improve the performance, such as smart restarts and annealing [18]. We do not pursue such enhancements in this paper, but believe that it is a fertile avenue for future work.

One possible way to perform the sampling is to use the Metropolis-Hastings algorithm [19], [20]. For this we need to construct a irreducible Markov chain with state space $\mathcal{F}$ (note that in this case the state space is finite), so that each state corresponds to a tree. We allow only certain transitions (or, equivalently, allocate certain transitions probability 0). For a given state (a tree) $s_i \in \mathcal{F}$ we can move to another state (tree) using two possible moves: (i) a “death” move, in which we collapse an internal edge (not a leaf edge), identifying the endpoints of the collapsed edge; and (ii) a “birth” move, in which we first choose a pair of children of a node k with three or more children, then introduce a new node k^* and make it the parent node of the chosen pair and a child of k. The nature of the moves is illustrated on Figure 3.

We build our Markov chain using a step-by-step approach. For a given state $s_i \in \mathcal{F}$ there are n_{s_i} allowed transitions (both deaths and births). This number can be easily computed. At any state s_i, we choose among the possible transitions uniformly, that is, we choose a particular transition with probability $1/n_{s_i}$. The transition matrix for this chain is

$$q_{T,T'} \quad = \quad \Pr(s_{i+1} = T' | s_i = T) = \begin{cases} 0 & \text{if } T' \text{ is not within one move from } T \\ \frac{1}{n_T} & \text{otherwise} \end{cases} . \tag{13}$$

Let $\{s_i : i \in \mathbb{N}\}$ be the chain satisfying the Markov condition (13).

Proposition 4: The chain $\{s_i\}$ is irreducible, that is

$$\forall T, T' \in \mathcal{F} \quad \exists n \in \mathbb{N} : q^{(n)}_{T,T'} > 0 \; ,$$

where $q^{(n)}_{T,T'}$ is the probability of reaching state T' starting from state T using n moves.
The proof of the proposition is presented in Section VIII-D.

Using a generalization of the Metropolis algorithm [19] due to Hastings [20], we construct another Markov Chain in order to obtain a chain whose unique limit distribution (and also unique stationary distribution) is precisely $\mathcal{L}'(\mathbf{x}|T)$. Thus if we sample from this chain for a sufficiently large period of time, the observed states will be samples of the distribution $\mathcal{L}'(\mathbf{x}|T)$, regardless of the initial state.

The new chain can be constructed in a two-stage fashion. Define the probability of acceptance $\alpha_{T,T'}$

$$\alpha_{T,T'} = \begin{cases} \min\left\{\frac{\mathcal{L}'(\mathbf{x}|T')\cdot q_{T',T}}{\mathcal{L}'(\mathbf{x}|T)\cdot q_{T,T'}}, 1\right\} & \text{if } \mathcal{L}'(\mathbf{x}|T)\cdot q_{T,T'} > 0 \\ 1 & \text{otherwise} \end{cases} .$$

For a given state $T \in \mathcal{F}'$ we randomly choose the possible next state according to $q_{T,T'}$, and accept that transition with probability $\alpha_{T,T'}$. Hence the transition matrix of the new chain is $\{r_{T,T'}\}$ with

$$r_{T,T'} = \begin{cases} \alpha_{T,T'} q_{T,T'} & \text{if } T' \neq T \\ 1 - \sum_{T''\neq T} \alpha_{T,T''} q_{T,T''} & \text{if } T' = T \end{cases} . \quad (14)$$

We need to ensure now that the above chain has a unique limit distribution, proportional to $\mathcal{L}'(\mathbf{x}|T)$. Clearly, if we start in a state in $\mathcal{F}'$ the next state is going to be in $\mathcal{F}'$ (with probability one). One needs to be sure that the chain with transition matrix $\{r_{i,j}\}$ is irreducible and aperiodic, and thus has a unique limit distribution (12) [21]. The following proposition, proved in Section VIII-E, ensures these properties.

Proposition 5: The chain with state space $\mathcal{F}'$ and transition matrix $\{r_{T,T'}\}$ is irreducible and aperiodic.

To get our (approximate) solution of (11) we simulate the above chain and consider the MAP estimator, that is, the state with largest value $\widehat{\mathcal{L}}(\mathbf{x}|T)$ visited. The longer the chain is simulated, the higher is the chance of visiting the MLT at least one time. Theoretically, the initial state of the chain is not important, provided that the chain is simulated for a sufficient period. In practical applications, on the other hand, it is of crucial importance, since the state space is very large, and reaching the region where the MLT lies may take a large number of transitions. Starting the chain simulation from the tree obtained using the ALT algorithm is a reasonable choice, since this is a consistent estimator in the binary case, and one expects the ALT to be "close" (in terms of the number of moves) to the actual MLT tree. If that is the case, the MCMC starts by searching trees that are near the ALT tree, and therefore close to the MLT. This is particularly relevant for clusterings involving a large number of input objects, yielding a very large state-space $\mathcal{F}'$, difficult to explore completely in reasonable time. Starting the chain from the ALT ensures that the search will focus on a promising region of the forest. Other techniques can be used to enhance the performance of this method, for example restarting the chain at various different trees [18].

It is important to point out that, although the states visited by a simulation of the chain constructed above may be regarded as samples from the (non-normalized) discrete distribution $\mathcal{L}'(\mathbf{x}|T)$, these samples are not independent, and so, if the distribution is multimodal, we can get "stuck" on one of the modes of the distribution, although eventually (*i.e.*, with positive probability) we will jump out. This further illustrates that the initial state of the chain is of utmost importance in a practice. At the end of the following section we propose a way of directing the search, to prevent some of these problems.

D. Maximum Penalized Likelihood Estimation

One drawback of the maximum likelihood approaches to our problem is that, in general, trees with more links have higher likelihood values (since the extra degrees of freedom they possess allow them to fit the data more closely). If the true tree T is not binary then it is going to yield a lower likelihood value than a certain binary tree. This is an example of the "overfitting" problem in model selection. The extra links in the MLT are there only to fit the particular set of measurements more accurately, and do not reflect the underlying model we wish to identify. The usual strategy for tackling this issues is to weight the complexity of the models involved, penalizing models that are more complex [22], [23]. The basic idea behind penalized estimators is to find the "best" *simple* model.

In this approach, instead of maximizing the profile likelihood we will maximize the functional

$$\mathcal{L}_\lambda(\mathbf{x}|T) = \mathcal{L}(\mathbf{x}|T)\exp(-\lambda n(T)) \ , \tag{15}$$

where $n(T)$ is the number of internal links in the tree T, and $\lambda \geq 0$ is parameter that controls the trade-off between the fit of the data and the tree complexity (number of links). Our estimate is then

$$T^*_\lambda \ = \arg\max_{T\in\mathcal{F}} \log \mathcal{L}_\lambda(\mathbf{x}|T) \ . \tag{16}$$

We denote T^*_λ by the Maximum Penalized Likelihood Tree (MPLT). Notice that if $\lambda = 0$ this is precisely the MLT estimate. The higher the value of λ, the more effect the penalization has, relative to the likelihood (data-fitting term). Large values of λ lead to trees with fewer links. Note that the complexity penalization criterion above takes a global approach, and it is not based on local decisions, unlike the pruning techniques mentioned in the end of Section V-B in the context of ALT algorithm.

The following result, similar in spirit to Theorem 1, provides a partial characterization of the solution of (16) and alleviates the search for the optimal tree.

Proposition 6: The MPLT T^*_λ is in the feasible tree set $\mathcal{F}'$.

This result (proved in Section VIII-F) indicates that we can use our MCMC approach as before, but now instead of $\mathcal{L}'(\mathbf{x}|T)$ we use $\mathcal{L}'_\lambda(\mathbf{x}|T) = \mathcal{L}'(\mathbf{x}|T)\exp(-\lambda n(T))$. The chain obtained in this fashion is still irreducible and aperiodic and hence has a unique limit distribution.

The choice of the complexity penalty parameter λ is a crucial step in the above approach. There is a vast literature addressing this issue; a classic technique is the Minimum Description Length principle [22]. In this paper the choice of λ follows from the analysis of a simple, but relevant model (see Section VI). Suppose that x_{ij} is normally distributed with mean γ_{ij} and variance σ^2 (that is $x_{ij} \sim \mathcal{N}(\gamma_{ij}, \sigma^2)$). Consider now a tree with three leaf nodes (which we refer to as the true tree). We have essentially two possibilities: (i) either all the three leaves have a common parent; (ii) the tree is binary. In both cases we can ask when does the penalized approach fails, producing a tree with a different number of nodes than the true tree. The two failure probabilities can be easily computed; for case (i) the failure probability is a decreasing function of λ, for case (ii) it is an increasing function of λ. Ideally we want the two probabilities to be small. Balancing these two quantities provides a tradeoff that can be used to choose the appropriate value of λ. Despite the simplicity of the model, it provides useful guidelines for the choice of penalty parameter for larger numbers of objects, under the Gaussian model.

As a closing remark on this Section, notice that a simple procedure, inspired by Simulated Annealing techniques [24], can be used in the context of the MCMC method developed. The idea is to start simulating the Markov chain with a large penalty parameter λ and reduce it gradually (according to some “cooling schedule”). This technique is different than the traditional simulated annealing procedure, but follows the main conceptual ideas. This approach forces the search to focus first on simpler trees (those with less links) and then gradually extend the search to explore increasingly complex trees. This reduces the possibility of the search being stuck in a undesirable region (a local maximum). There are several parameters that need to be tuned, and we do not address this issue here. Discussions on general simulated annealing techniques can be found in [24].

VI. Network Topology Identification

In this Section we present a practical example that illustrates the use of the above framework. We discuss some simulation results, as well as practical experimental results. Consider a communication network, in which a sender node transmits information packets to a set of receiver nodes, denoted by S. The receivers are, in this case, the usual “objects” to be clustered. Assume that the routes from the sender to the receivers are fixed. The physical network topology is essentially a graph, where each node corresponds to a physical device (*e.g.*, router, switch, terminal, etc.) and the links correspond to the connections between these.

The problem we address is the identification of the network topology based on end-to-end measurements, which is a practical problem relevant in the networking community [25]–[28]. Knowledge of the network topology is essential for tasks like monitoring and provisioning a network. There are tools, such as `traceroute`, that rely on close cooperation from the network internal devices, such as routers. These tools are effective only when the devices are responsive and working properly. These conditions are becoming less common as the network grows, and with the increasing concerns with malicious activities (denial-of-service attacks, worms, etc.). Therefore it is important to be able to infer information from the network without cooperation from the internal devices. In the following, we consider only end-to-end measurements, preventing us from utilizing internal network device information. This forces us to rely solely on the traffic and queueing characteristics. With this limited information, it is only possible to identify the so-called "logical topology", defined by the branching points between paths to different receivers. This corresponds to a tree-structured topology with the sender at the root and the receivers at the leaves, as depicted in Figure 1. The tree topology is effectively a hierarchical clustering of the receivers, where each cluster corresponds to the maximal set of receivers that share a common device in the network.

We define a similarity metric γ_{ij}, associated with each pair of receivers $i, j \in R$. The value of γ_{ij} is related to the extent of the shared portion of routes from the sender to receivers i and j (*i.e.*, the shared path for pair (i, j), as defined before). We cannot measure the pairwise metrics directly, so we estimate them, performing measurements across the network. In most cases, the measurement is based on active probing, involving the transmission of probe packets from the source to the receivers. In earlier work, we proposed a metric based on delay difference measurements [27]. The delay difference measurements provide (noisy) versions of a metric related to the number of shared queues in the paths to two receivers. More precisely, if the constituent links of the shared path have bandwidths B_m, the pairwise metric value is approximately proportional to $\sum 1/B_m$, the sum of the inverse bandwidths of the constituent links. Thus each link in the shared path contributes an additional positive measure that is inversely proportional to its bandwidth, and therefore the monotonicity property follows naturally.

The measurements $\{x_{ij}\}$ are the empirical means of repeated delay difference measurements. Under reasonable assumptions, the measurements are statistically independent, and, according to the Central Limit Theorem, the distribution of each empirical mean tends to a Gaussian. This motivates the following (approximate) model:

$$x_{ij} \;\sim\; \mathcal{N}(\gamma_{ij}, \sigma_{ij}^2) \;, \tag{17}$$

where σ_{ij}^2 is sample variance of the n_{ij} measurements associated with empirical mean x_{ij}, divided by n_{ij}, and $\mathcal{N}(\gamma, \sigma^2)$ denotes the Gaussian density with mean γ and variance σ^2. The measurements for different pairs of receivers are also independent.

Although model (17) is just an approximation of the true underlying statistics of x_{ij} it allows us to cope with practical modeling difficulties in a relatively easy way. Note that we are in the scenario described in Section III:

$$h_{ij}(x_{ij}, \sigma_{ij} | \gamma_{ij}) = -\frac{(x_{ij} - \gamma_{ij})^2}{2\sigma_{ij}^2} + C_{ij} \ , \tag{18}$$

where C_{ij} is a normalizing constant. We can then apply the algorithms developed before to estimate the network topology. Furthermore the model (18) has some desirable properties, namely, it is closed under summation:

$$h_{ij}(x_{ij}, \sigma_{ij}^2 | \gamma) + h_{kl}(x_{kl}, \sigma_{kl}^2 | \gamma) = -\frac{\left(\left(\frac{x_{ij}}{\sigma_{ij}^2} + \frac{x_{kl}}{\sigma_{kl}^2} \right) / \left(\frac{1}{\sigma_{ij}^2} + \frac{1}{\sigma_{kl}^2} \right) - \gamma \right)^2}{2 \left(\frac{1}{\sigma_{ij}^2} + \frac{1}{\sigma_{kl}^2} \right)^{-1}} + C \ .$$

This makes the computations arising from Lemma 1 very simple.

We performed some simple simulation experiments according to the model above, to assess the performance of the algorithms developed. In these simulations we randomly generate tree topologies (clusterings), that we call the true generative trees (clusterings). We compare the true and estimated clusterings using two criterions: (i) the percentage of clusters of the true clustering that were correctly identified in the clustering estimate; and (ii) the percentage of clusters of the estimate that are not present in the true underlying clustering.

In the first set of simulations we randomly generated 1000 binary topologies with 10 leaves, where the link-level metric values (*i.e.*, $\gamma_k - \gamma_{f(k)}$) were generated randomly (independent and identically distributed like $1 + E$, where E denotes a standard exponential random variable). The variance σ_{ij} for each pair of leaves was chosen uniformly in the range $[1, 4]$. We observed that in general the ALT algorithm and the MCMC method ($\lambda = 0$, stopped after 2500 iterations) had similar performances under this scenario. On average the ALT algorithm identified 93.8% of the clusters in the true clustering and failed to identify 6.2% of them. On the other hand the MCMC method identified on average 92.2% of the true clusters and failed to identify 4.1%. We experimented with different variance ranges, but the behavior of the two algorithms exhibited the same trends. The MCMC approach yielded in general a smaller number of misclassified clusters, although the difference in performance between the two algorithms was not very significant. In the second set of simulations we considered also non-binary trees, obtained by randomly pruning binary trees. In this case we used the penalized approach in the MCMC algorithm, with a penalty parameter λ chosen with the aid of the guidelines described above. The number of correctly identified clusters was still comparable between the two algorithms (93.1% for the ALT and 91.0% for the MCMC), but, as expected, the number of misclassified clusters was significantly higher for the ALT (19.2% opposed to 11.3% for the MCMC). This indicates that the ALT is still performing well, except that it is introducing more clusters than necessary. In a third experiment we considered larger binary topologies (100 objects), in this case the initial tree estimate (obtained from the ALT algorithm) was generally the tree with the largest likelihood visited by the MCMC method.

The above experiments indicate that, under the scenarios considered, the ALT is a robust estimator, although it tends to overfit the data. It is possible to devise scenarios where the ALT estimator fails, due to its greedy nature. We conducted some simple simulations in order to contrast the characteristics of the MCMC, the ALT, and a classic algorithm, the unweighted pair-group average method (UPGMA) [10]. These simulations are not an exhaustive performance comparison of the algorithms, but are intended to indicate the benefits and drawbacks of the various methods.

We consider a randomly chosen six receiver binary topology, such that the link-level parameters $\gamma_k - \gamma_{f(k)}$ are the same for all links ($\gamma_k - \gamma_{f(k)} = 1$). For each pair of leaves $(i, j) \in S$ we generate 100 independent measurements with variance $100\sigma_{ij}^2$. In the first experiment we let $\sigma_{ij}^2 = \sigma^2/100$ for all $i \neq 1$, and $\sigma_{1j}^2 = \alpha^2\sigma^2/100$, where $\sigma^2 = 25$ and $\alpha > 1$. This corresponds to a practical networking scenario: since the delay differences are measured at each receiver, we mimic the case where measurements x_{1i} taken at receiver 1 (identified with leaf 1) are less reliable than the measurements collected elsewhere. For each value of α we evaluate the three methods on 1000 randomly generated trees.

In Figure 4(a) we plot the performance of the three methods as a function of the standard deviation ratio α. As we can observe, the MCMC and ALT performance are very similar, and remain almost constant for the range of α considered. On the other hand, the performance of the UPGMA algorithm degrades significantly as α increases. The reason for this is that the metrics $\hat{\gamma}_{1j}$ in the UPGMA are strongly affected by the highly variable empirical means x_{1j}, for $j \in R$. In the case of the ALT, those empirical means do not affect the performance because the measurements are weighted according to their variance. Hence, for high values of α, the measurements x_{1j} have little impact on the estimated tree.

In Figure 4(b) we plot the results of an experiment similar to the previous one, but now $\sigma_{ij}^2 = \sigma^2/100$ for $i \neq \{1, 2\}$ and $\sigma_{1j}^2 = \sigma_{2j}^2 = \alpha^2\sigma^2$. In this case we observe that the performance of the three methods degrades as α increases. The UPGMA algorithm exhibits the same trends as before, degrading considerably as α increases, but we note also that the ALT performance degrades considerably as compared to that of the MCMC, although it is still much better than that of the UPGMA. The reason for this is that the estimate of γ_{12} has a high variance, since both σ_{12}^2 and σ_{21}^2 are large, for large α. There is a higher probability of mispairing leaves 1 and 2, since there is a higher chance of choosing that particular pair in the greedy decision step. Unlike the ALT, the MCMC algorithm is able to account for the higher variance of those measurements, with respect to all the other measurements, consequently improving its performance.

We conducted Internet experiments that demonstrate the use of the techniques described in a practical scenario. In this case we had also access to a partial topology map of the network, obtained using a tool (called `traceroute`) relying on network device information. While in this case the topology can be partially identified using `traceroute`, in many other cases routers may not cooperate. The point of this particular experiment is that we can use the `traceroute` topology as a “ground-truth” to verify that our method produces an accurate topology. The detailed experimental setup can be found in [27]. The measurements collected for each pair span a wide range of values: The maximum empirical average ($3464\mu s$) was observed for the measurements involving the two machines in Portugal, which is not surprising since those receivers share a common bottleneck link with small capacity. The variability associated with the measurements taken at those machines is also large, especially for the ones taken at the I.S.T. machine (standard deviation as large as $746\mu s$, and all larger than the standard deviation of measurements taken at any other receiver). On the other hand, for other pairs of receivers we observed empirical averages as low as $307\mu s$ and corresponding standard deviations as small as $23.5\mu s$.

In Figure 5 we depict the topology obtained using `traceroute` and the topology obtained solely from end-to-end measurements, using the algorithms described in Section V. Using the ALT algorithm we are always going to obtain a binary tree, and thus some of the links might be artifacts due to data “overfitting” (see [27] for details). In the ALT solution we notice that three links have link-parameter values $\gamma_k - \gamma_{f(k)}$ one order of magnitude smaller than all the other links (see Figure 5(b)). This suggests that the four nodes inside the ellipse might be an artifact of the binary tree, and possibly correspond to a single node in the logical topology. Using the MPLT technique we obtain a simpler tree, with less artifacts than the ALT tree, as depicted in Figure 5(c). Notice that the MPLT is very close to the `traceroute` topology, but fails to detect the backbone connection between Texas and Indianapolis. We know that the latter connection has a very large bandwidth and the queuing effects on the constituent links are too minor to influence measurements. The estimated topologies also place an extra element shared between the Rice computers. Although that element is not a router, hence it is not shown in the topology estimate using `traceroute`, it corresponds to a existing physical device. To the best of our knowledge the detected element is a bandwidth limitation device.

VII. Final Remarks

In this paper we develop a new framework for hierarchical clustering based on a generative dendritic cluster model. We pursue a maximum likelihood approach and present two clustering algorithms based on this framework. The ALT algorithm has low complexity, and is an agglomerative hierarchical clustering algorithm, hence greedy. The second algorithm presented overcomes the problems of the greedy nature of the ALT by performing a random (but informed) search on the space of possible clusterings, using Markov Chain Monte Carlo techniques, aiming to find the maximum likelihood clustering tree.

For the case of underlying binary trees, consistency of ALT and the maximum likelihood approach can be shown. In the case of non-binary trees the maximum likelihood approach tends to fit the data too closely, and does not convey the description of the underlying model properly, having more clusters than the underlying generating model. To overcome those problems we propose a maximum penalized likelihood approach that enforces the choice of simpler models whenever the decrease in likelihood is small (as compared to complex models). Both random search approaches benefit greatly from a characterization of the maximum likelihood tree, given by Theorem 1.

Although the MCMC approach is very appealing, the fact that the number of trees in the forest grows combinatorially with the number of objects to cluster renders the search of the entire forest difficult. On the other hand the ALT algorithm has low complexity, and appears to perform extremely well in a variety of scenarios, although overfitting the data. As seen in Section V-C, in many cases the ALT is likely to be “close” to the MLT, therefore, when the MCMC is started using the ALT, it improves on the ALT simply by conducting a search over the trees in the vicinity of the ALT. This is particularly important when the number of input objects is large.

The framework presented here is very well suited to problems like the network topology identification, wherein the generative model reflects the physical mechanisms involved. Ongoing research is aimed at assessing the advantages and disadvantages of the framework in more general settings, as well as deriving performance guarantees under general models.

Acknowledgements: The authors would like to thank the anonymous reviewers for the numerous helpful comments that allowed us to greatly improve the presentation of the topic.

VIII. Proofs of the Results

For various proofs in this Section we need the following lemma, characterizing the sum of strictly concave functions.

Lemma 2: Let $i \in \{1, \ldots, n\}$ and $f_i(\cdot)$ be strictly concave functions with (unique) maximizers $x_i \in \mathbb{R}$. Let $g(x) = \sum_{i=1}^{n} f_i(x)$. Define $x_{\max} = \max_i x_i$ and $x_{\min} = \min_i x_i$. The function $g(x)$ is strictly concave, has a unique maximizer $\widehat{x} = \arg\max_{x \in \mathbb{R}} g(x)$ and, if $x_{\max} \neq x_{\min}$ then $x_{\min} < \widehat{x} < x_{\max}$.

A. Proof of Lemma 1

The proof of (7) is elementary, and relies on reordering the terms in (5). Begin by noting that

$$\arg \max_{\boldsymbol{\gamma} \in \boldsymbol{\Gamma}(T)} \log p(x|\boldsymbol{\gamma}) = \arg \max_{\boldsymbol{\gamma} \in \boldsymbol{\Gamma}(T)} \sum_{v \in V \setminus \{S\}} \sum_{i,j \in S : a(i,j) = v} h_{ij}(x_{ij}|\gamma_{ij}) \ .$$

Each of the terms in the first summation corresponds to a single metric value. Since there are no further restrictions on those metric values we can maximize each term separately, that is,

$$\hat{\gamma}_{ij} = \hat{\gamma}_v = \arg\max_{\gamma \in \mathbb{R}} \sum_{k,l \in S : a(k,l) = v} h_{kl}(x_{kl}|\gamma), \quad \text{where } v = a(i,j) \ .$$

The uniqueness and existence of the above solution follows directly from Lemma 2. ■

B. Proof of Theorem 1

The proof is done by construction. Given a tree $\widetilde{T}$ satisfying (8) we construct a tree T satisfying (9). Let $\widetilde{T}$ be a tree satisfying (8). Let

$$\widetilde{\boldsymbol{\gamma}} = \arg \max_{\boldsymbol{\gamma}' \in \overline{\mathcal{G}(\widetilde{T})}} \log p(\mathbf{x}|\boldsymbol{\gamma}') \quad \text{and} \quad \widetilde{\boldsymbol{\gamma}}_{\text{unconst}} = \arg \max_{\boldsymbol{\gamma}' \in \boldsymbol{\Gamma}(\widetilde{T})} \log p(\mathbf{x}|\boldsymbol{\gamma}') \ .$$

Note that $\overline{\mathcal{G}(\widetilde{T})} \subseteq \boldsymbol{\Gamma}(\widetilde{T})$ and that $\mathcal{G}(\widetilde{T})$ is an open set. By concavity of the log-likelihood we have

$$\begin{aligned} \log p(\mathbf{x}|(1-\lambda)\widetilde{\boldsymbol{\gamma}} + \lambda \widetilde{\boldsymbol{\gamma}}_{\text{unconst}}) &> (1-\lambda) \log p(\mathbf{x}|\widetilde{\boldsymbol{\gamma}}) + \lambda \log p(\mathbf{x}|\widetilde{\boldsymbol{\gamma}}_{\text{unconst}}) \\ &\geq \log p(\mathbf{x}|\widetilde{\boldsymbol{\gamma}}), \quad \lambda \in (0,1) \ . \end{aligned}$$

Thus, if $\widetilde{\boldsymbol{\gamma}} \in \mathcal{G}(\widetilde{T})$ then choosing a small $\lambda > 0$ we get another point in $\mathcal{G}(\widetilde{T})$ yielding a higher log-likelihood, a contradiction. Hence $\widetilde{\boldsymbol{\gamma}} \in \partial\mathcal{G}(\widetilde{T})$ where ∂ denotes the boundary of a set.

The fact that $\widetilde{\boldsymbol{\gamma}} \in \partial\mathcal{G}(\widetilde{T})$ holds, indicates that there are links l such that $\widetilde{\gamma}_l = \widetilde{\gamma}_{f(l)}$. Consider now the tree obtained by collapsing all such links and keeping the value of the remaining link-level parameters unchanged. Denote the tree and corresponding metric values by T' and $\boldsymbol{\gamma}'$ respectively (see Figure 6 (a)(b)).

Note that the parameters $\boldsymbol{\gamma}'$ satisfy the constrains (2), and that $(T', \boldsymbol{\gamma}')$ yields the same log-likelihood value as $(\widetilde{T}, \widetilde{\boldsymbol{\gamma}})$, that is $\log p(\mathbf{x}|\widetilde{\boldsymbol{\gamma}}, \widetilde{T}) = \log p(\mathbf{x}|\boldsymbol{\gamma}', T')$. By our construction we see that $\boldsymbol{\gamma}' \in \mathcal{G}(T')$. We conclude that

$$\boldsymbol{\gamma}' = \arg \max_{\boldsymbol{\gamma} \in \boldsymbol{\Gamma}(T')} \log p(\mathbf{x}|\boldsymbol{\gamma}) \ . \tag{19}$$

To see this suppose that (19) does not hold. Then, by the same argument used in the beginning of the proof (but now applied to T' instead of $\widetilde{T}$) we must have $\boldsymbol{\gamma}' \in \partial\mathcal{G}(T')$, a contradiction. Note that, according to Lemma 1, $\boldsymbol{\gamma}'$ satisfies (7).

The rest of the proof ensues by constructing another tree by adding links to T', such that (9) holds.

Consider a node k of T' such that k has more than two descendants (such a node must exist because we pruned at least one link from $\widetilde{T}$). Define

$$g_{ij}(\gamma) = \sum_{m\in S(i)} \sum_{n\in S(j)} h_{mn}(x_{mn}|\gamma) + h_{nm}(x_{nm}|\gamma), \quad i,j \in c(k),\ i \neq j\ ,$$

and

$$\beta_{ij} = \arg\max_{\gamma\in\mathbb{R}} g_{ij}(\gamma), \quad i,j \in c(k),\ i \neq j\ .$$

From Lemma 1 we know that

$$\gamma'_k = \arg\max_{\gamma\in\mathbb{R}} \sum_{i,j\in c(k),\ i\neq j} g_{ij}(\gamma)\ .$$

Case 1: Suppose that not all the values β_{ij} are the same. Then there exists a pair of nodes $o, p \in c(k)$ such that $\beta_{op} \geq \beta_{ij}$, for all $i, j \in c(k)$, and from Lemma 2 we conclude that $\beta_{op} \neq \gamma'_k$ (since $\min_{i,j} \beta_{ij} < \gamma'_k < \max_{i,j} \beta_{ij}$). Using the chosen pair (o, p), we construct a new tree T (refer to Figure 6(c)) adding an extra node k^* descending from k and with children $\{o, p\}$. Loosely speaking we are pulling the pair of nodes o and p down, adding a new node k^*. The parameter values for this new tree, denoted by $\boldsymbol{\gamma}$, are adjusted such that $\gamma_{k^*} = \gamma'_k + \delta$, $\delta > 0$. All the other metric values remain the same. Note that $\delta > 0$, but small enough so that the tree $(T, \boldsymbol{\gamma})$ still satisfies the constraints (2).

The log-likelihood of $(T, \boldsymbol{\gamma})$ is identical of the one from $(\widetilde{T}, \widetilde{\boldsymbol{\gamma}})$, except for the term involving γ_{k^*}. Thus

$$\begin{aligned}
\log p(\mathbf{x}|\gamma) - \log p(\mathbf{x}|\gamma') &= g_{op}(\gamma^*_k) - g_{op}(\gamma'_k) \\
&= g_{op}(\gamma'_k + \delta) - g_{op}(\gamma'_k) \\
&= g_{op}((1-\lambda)\gamma'_k + \lambda\beta_{op}) - g_{op}(\gamma'_k) \\
&> (1-\lambda)g_{op}(\gamma'_k) + \lambda g_{op}(\beta_{op}) - g_{op}(\gamma'_k) \\
&= \lambda(g_{op}(\beta_{op}) - g_{op}(\gamma'_k)) > 0\ ,
\end{aligned}$$

where we take $\lambda = \delta/(\beta_{op} - \gamma'_k)$. The last inequality follows from uniqueness of the maximizer of g_{op} and $\beta_{op} \neq \gamma'_k$.

In conclusion, for δ small enough, we have $\log p(\mathbf{x}|\boldsymbol{\gamma}) > \log p(\mathbf{x}|\boldsymbol{\gamma}')$, and hence

$$\log p(\mathbf{x}|\boldsymbol{\gamma}) > \log p(\mathbf{x}|\widetilde{\boldsymbol{\gamma}})\ . \tag{20}$$

Case 2: In this case all the values β_{ij}, $i, j \in c(k)$ are the same. Thus $\gamma'_k = \beta_{ij}$ for all $i, j \in c(k)$. As in Case 1 we construct another tree T by adding an extra node k^* descending from k, such that node k^* has children o and p. The parameter values for this new tree, denoted by $\boldsymbol{\gamma}$, are adjusted such that $\gamma_{k^*} = \gamma'_k$. All the other metric values remain the same. From Lemma 1 we observe that $\boldsymbol{\gamma}$ is the maximizer over $\boldsymbol{\Gamma}(T)$ of $\log p(\mathbf{x}|\boldsymbol{\gamma})$. Hence $\log p(\mathbf{x}|\boldsymbol{\gamma}, T) = \log p(\mathbf{x}|\widetilde{\boldsymbol{\gamma}}, \widetilde{T})$.

Suppose that all nodes of T' with more than two descendants correspond to case 2. We could then add to T' all the links we removed (when we constructed T' from $\widetilde{T}$), and obtain the original tree $\widetilde{T}$, but in this case $\max_{\boldsymbol{\gamma}' \in \boldsymbol{\Gamma}(\widetilde{T})} \log p(\mathbf{x}|\boldsymbol{\gamma}') = \max_{\boldsymbol{\gamma}' \in \overline{\mathcal{G}(\widetilde{T})}} \log p(\mathbf{x}|\boldsymbol{\gamma}')$, a contradiction. Hence at least one node of T' corresponds to case 1, and so (20) necessarily holds, proving the result. The second part of the theorem follows from *Remark 1*. ■

C. *Proof of Proposition 2*

The result is a consequence of the following lemma.

Lemma 3: Let X_ρ be a collection of real random variables converging in probability (when $\rho \to \infty$) to constant a. Let Y_ρ be another collection of random variables converging to constant b, and $a < b$. Then for any $\delta > 0$ we have that $\Pr(X_\rho > Y_\rho) < \delta$ for large enough ρ.

We observed before, in Proposition 1, that the relative ordering of the metric values $\{\gamma_{ij}\}$ uniquely determines the tree. Observing that our estimates of the metric values $\{\widehat{\gamma}_{ij}\}$ converge in probability to the real metric values, one concludes that the probability of the relative ordering of the estimated and true metric values being different becomes smaller as ρ increases, and so the result follows. ■

D. *Proof of Proposition 4*

Let T be the initial state, with $n(T)$ internal links. If we apply a sequence of $n(T)$ death moves, each one removing an internal link, we obtain a tree T'' with a single internal node. Since each of the death moves has non-zero probability we are able to reach T'' within $n(T)$ moves with non-zero probability. The same reasoning applies if we considered T' to be the initial state. In this case we would be able to reach T'' with non-zero probability using $n(T')$ moves. Considering now that for each death move there is a corresponding birth move, we have a sequence of birth moves carrying tree T'' to T'. Thus $q_{T,T'}^{(n(T)+n(T'))} > 0$ and hence the chain is irreducible. ■

E. *Proof of Proposition 5*

To prove this result it suffices to show that one can move from any tree $T \in \mathcal{F}'$ to any other tree $T' \in \mathcal{F}'$ using the birth and death moves. From this and the fact that $\mathcal{L}'(\mathbf{x}|T) > 0$ for all $T \in \mathcal{F}'$ it follows that the chain is irreducible. The aperiodicity follows from the chain construction in (14).

The proof strategy is the same as use for Proposition 4, that is, we begin by applying a sequence of death moves to get from T to a tree with no internal links, and then a sequence of birth moves taking us to T'. We only need to check that each transition tree is still in $\mathcal{F}'$. Without loss of generality suppose that T has at least one internal node. It suffices to show that there is a death move that can be applied to T, yielding a tree that is still in the feasible forest $\mathcal{F}'$.

Let $\widehat{\gamma}$ be given by (7) for the tree T. Consider an internal node k such that all its descendants are leaf nodes. Furthermore choose node k, such that it is the one with smallest metric value $\hat{\gamma}_k$. When we apply a death move, "killing" the internal link $(f(k), k)$ we obtain another tree $\widetilde{T}$. It can be verified, with the aid of Lemma 2, that this tree is still in the feasible forest, that is $\widetilde{T} \in \mathcal{F}'$. The rest of the proof continues as the proof of Proposition 4. ■

F. *Proof of Proposition 6*

The proof follows from a key observation: In Theorem 1, the constructed tree (T, γ) has no more links than the initial tree $\widetilde{T}$. We proceed by contradiction. Let $\widetilde{T}$ be a candidate MPLT and suppose

$$\max_{\boldsymbol{\gamma}' \in \mathbf{\Gamma}(\widetilde{T})} p(\mathbf{x}|\boldsymbol{\gamma}') \exp(-\lambda n(\widetilde{T})) \;>\; \max_{\boldsymbol{\gamma}' \in \overline{\mathcal{G}(\widetilde{T})}} p(\mathbf{x}|\boldsymbol{\gamma}') \exp(-\lambda n(\widetilde{T})) \; .$$

Then, from Theorem 1, there exists another tree $(T, \boldsymbol{\gamma})$, $\boldsymbol{\gamma} \in \mathcal{G}(T)$, satisfying the monotonicity property, such that $p(\mathbf{x}|\boldsymbol{\gamma}) \; > \max_{\boldsymbol{\gamma}' \in \overline{\mathcal{G}(\widetilde{T})}} p(\mathbf{x}|\boldsymbol{\gamma}')$. Furthermore $n(\widetilde{T}) \geq n(T)$, thus

$$p(\mathbf{x}|\boldsymbol{\gamma}) \exp(-\lambda n(T)) \;>\; \max_{\boldsymbol{\gamma}' \in \overline{\mathcal{G}(\widetilde{T})}} p(\mathbf{x}|\boldsymbol{\gamma}') \exp(-\lambda n(\widetilde{T})) = \mathcal{L}_\lambda(\mathbf{x}|\widetilde{T}) \; ,$$

and so $\widetilde{T}$ cannot be the MPLT. ■

REFERENCES

[1] D. Fasulo, "An analysis of recent work on clustering algorithms," Department of Computer Science and Engineering, University of Washington, Tech. Rep. # 01-03-02, 1999. [Online]. Available: citeseer.nj.nec.com/fasulo99analysi.html

[2] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999. [Online]. Available: citeseer.nj.nec.com/jain99data.html

[3] D. Fisher, "Iterative optimization and simplification of hierarchical clusterings," *Journal of Artificial Intelligence Research*, vol. 4, pp. 147–180, 1996. [Online]. Available: citeseer.nj.nec.com/fisher95iterative.html

[4] F. Murtagh, "A survey of recent advances in hierarchical clustering algorithms," *The Computer Journal*, vol. 26, no. 4, 1983.

[5] J. D. Banfield and A. E. Raftery, "Model-based Gaussian and non-Gaussian clustering," *Biometrics*, vol. 49, pp. 803–821, 1993.

[6] P. Willet, "Recent trends in hierarchical document clustering: a critical review," *Information Processing and Management*, vol. 24, pp. 577–597, 1988.

[7] E. M. Voorhees, “Implementing agglomerative hierarchical clustering algorithms for use in document retrieval,” *Information Processing and Management*, vol. 22, pp. 465–476, 1986.

[8] A. El-Hamdouchi and P. Willet, “Hierarchic document clustering using ward’s method,” in *proceedings of the Ninth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1986.

[9] T. Zhang, R. Ramakrishnan, and M. Livny, “BIRCH: an efficient data clustering method for very large databases,” in *ACM SIGMOD International Conference on Management of Data*, Montreal, Canada, June 1996, pp. 103–114. [Online]. Available: citeseer.nj.nec.com/article/zhang96birch.html

[10] L. Kaufman and P. Rousseeuw, *Finding Groups in Data*. Wiley, 1990.

[11] T. Hofmann, “Unsupervised learning by probabilistic latent semantic analysis,” *Machine Learning*, vol. 42, pp. 177–196, 2001.

[12] C. Fraley, “Algorithms for model-based Gaussian hierarchical clustering,” *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 270–281, 1998.

[13] S. Vaithyanathan and B. Dom, “Model-based hierarchical clustering,” in *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, Stanford University; Stanford, CA., June 2000.

[14] S. D. Kamvar, D. Klein, and C. D. Manning, “Interpreting and extending classical agglomerative clustering algorithms using a model-based approach,” in *International Conference on Machine Learning (ICML)*, 2002.

[15] N. Duffield, J. Horowitz, F. L. Presti, and D. Towsley, “Multicast topology inference from measured end-to-end loss,” *IEEE Trans. Info. Theory*, vol. 48, no. 1, pp. 26–45, January 2002.

[16] J. O. Berger, B. Liseo, and R. L. Wolpert, “Integrated likelihood methods for eliminating nuisance parameters,” *Statistical Science*, vol. 14, pp. 1–28, 1999.

[17] M. A. Tanner, *Tools for Statistical Inference - Methods for the Exploration of Posterior Distributions and Likelihood Functions*. Springer Verlag Series in Statistics, 1996.

[18] W. Gilks, S. Richardson, and D. Spiegelhalter, *Markov Chain Monte Carlo in Practice*. Chapman & Hall, 1996.

[19] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, “Equations of state calculations by fast computing machines,” *J. Chem. Phys.*, vol. 21, pp. 1087–1091, 1953.

[20] W. Hastings, “Monte Carlo sampling methods using Markov chains and their applications,” *Biometrika*, vol. 57, pp. 97–109, 1970.

[21] S. I. Resnick, *Adventures in Stochastic Processes*. Birkhauser, 1992.

[22] J. Rissanen, *Stochastic Complexity in Statistical Inquiry*. Singapore: World Scientific, 1989.

[23] G. Schwarz, “Estimating the dimension of a model,” *Annals of Statistics*, vol. 6, no. 2, pp. 461–464, 1978.

[24] M. I. Jordan, *Learning in Graphical Models*, ser. NATO ASI Series. Kluwer, 1998, vol. D89.

[25] S. Ratnasamy and S. McCanne, “Inference of multicast routing trees and bottleneck bandwidths using end-to-end measurements,” in *Proceedings of IEEE INFOCOM 1999*, New York, NY, March 1999.

[26] N. Duffield, J. Horowitz, F. L. Presti, and D. Towsley, “Multicast topology inference from end-to-end measurements,” in *ITC Seminar on IP Traffic, Measurement and Modelling*, Monterey, CA, Sep. 2000.

[27] M. Coates, R. Castro, R. Nowak, and Y. Tsang, “Maximum likelihood network topology identification from edge-based unicast measurements,” in *Proc. ACM SIGMETRICS 2002*, Marina del Rey, CA, June 2002.

[28] R. Castro, M. Coates, G. Liang, R. Nowak, and B. Yu, “Network tomography: Recent developments,” 2004, accepted for publication.

LIST OF FIGURES

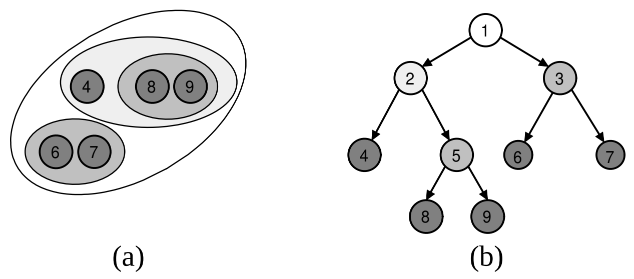


Fig. 1. Dual representation of an hierarchical clustering: (a) Example of an hierarchical clustering, (b) corresponding tree representation.

```
1) **Input:** Set of input nodes $S$, likelihood functions $\{f_{ij}\}$.
2) **Initialization:** $S' := S$, $V := S$, $S(r) = \{r\}, \forall r \in S$, and $\widehat{\gamma}_{ij} = \arg\max_{\gamma \in \mathbb{R}} (f_{ij}(x_{ij}|\gamma) + f_{ji}(x_{ji}|\gamma))$.
3) Find the a pair of nodes $i, j \in S'$ such that
       $$\widehat{\gamma}_{ij} \geq \widehat{\gamma}_{kl},\ \forall k, l \in S' \ .$$
4) Denote by $k$ the new node, the inferred parent of the nodes $i, j$. Set $V := V \cup \{k\}$, $S' := S' \cup \{k\} \setminus \{i, j\}$
   and $S(k) = S(i) \cup S(j)$. Set also $f(i) := k$; $f(j) := k$. Define
       $$\widehat{\gamma}_{kl} = \widehat{\gamma}_{lk} \equiv \arg\max_{\gamma \in \mathbb{R}} \sum_{r \in S(k)} f_{rl}(x_{rl}|\gamma) + f_{lr}(x_{lr}|\gamma) \ , \quad \text{where } l \in S' \setminus \{k\} \ .$$
5) If $|S'| > 1$ go back to 3. Otherwise set Root $\equiv k$.
6) **Output:** The node set $V$ and the parent function $f : V \setminus \{\text{Root}\} \to V$, defining a unique tree. Also the
   set of similarity estimates $\widehat{\gamma}$, if desired.
```

Fig. 2. Agglomerative Likelihood Tree (ALT) algorithm.

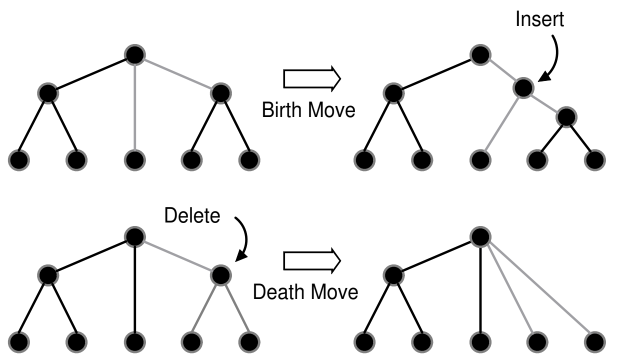
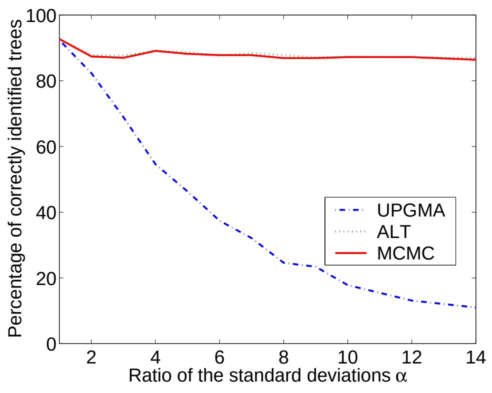
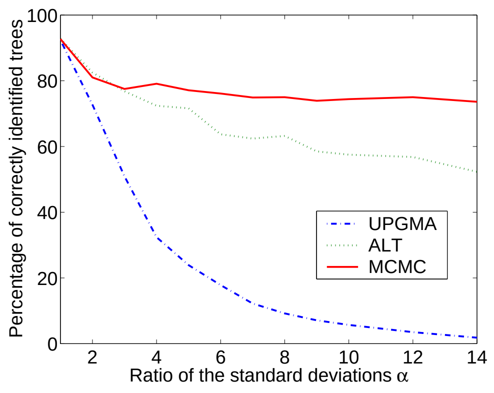


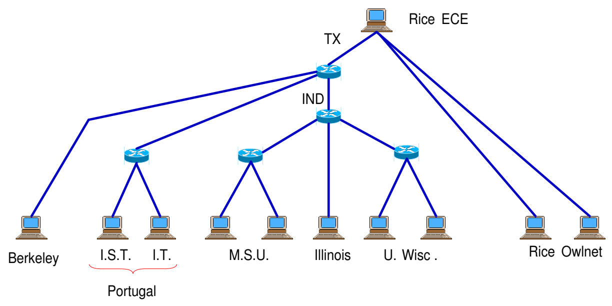
Fig. 3. The birth-step and death-step moves illustrated: The birth-step selects a node with more than two children, chooses two of these children, and inserts an extra node as the new parent of these children. The death step selects and deletes a internal node.

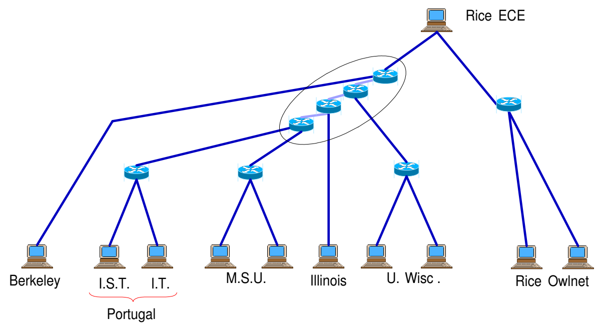




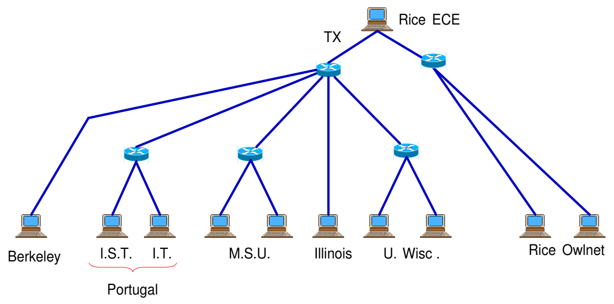
(a) (b)

Fig. 4. Simulation results contrasting the performance of the ALT, MCMC and UPGMA algorithms: (a) Performance results considering different measurement variance for one receiver, (b) Performance results considering different measurement variance for two receivers.





(a) (b)



(c)

Fig. 5. Comparison between the `traceroute` topology and the estimated topologies for the Internet experiment described in Section VI: (a) The topology of the network used for Internet experiments, obtained using `traceroute`. (b) Estimated topology using the ALT algorithm. The three links inside the ellipse have link-parameter values $\gamma_k - \gamma_{f(k)}$ one order of magnitude smaller than all the other links. (c) MPLT obtained using MCMC techniques.

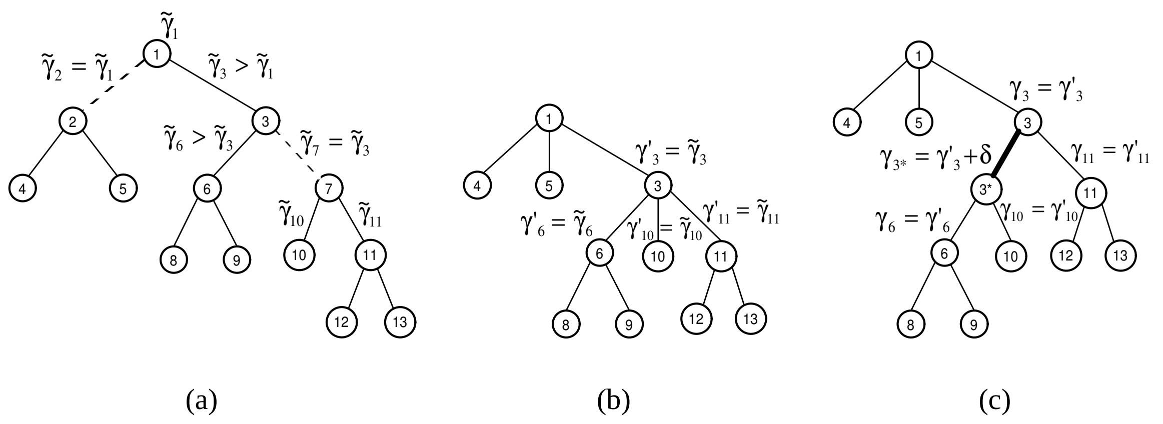

$\tilde{\gamma}_1$
$\tilde{\gamma}_2 = \tilde{\gamma}_1$
$\tilde{\gamma}_3 > \tilde{\gamma}_1$
$\tilde{\gamma}_6 > \tilde{\gamma}_3$
$\tilde{\gamma}_7 = \tilde{\gamma}_3$
$\tilde{\gamma}_{10}$
$\tilde{\gamma}_{11}$
(a)
$\gamma'_3 = \tilde{\gamma}_3$
$\gamma'_6 = \tilde{\gamma}_6$
$\gamma'_{10} = \tilde{\gamma}_{10}$
$\gamma'_{11} = \tilde{\gamma}_{11}$
(b)
$\gamma_3 = \gamma'_3$
$\gamma_{3*} = \gamma'_3+\delta$
$\gamma_{11} = \gamma'_{11}$
$\gamma_6 = \gamma'_6$
$\gamma_{10} = \gamma'_{10}$
3*
(c)

Fig. 6. Trees illustrating the proof of Theorem 1: (a) Original tree $(\widetilde{T}, \widetilde{\gamma})$; (b) Collapsed tree (T', γ'); (c) Constructed tree (T, γ).