Deep Tree Models for ‘Big’ Biological Data

(Invited Paper)

Lambros Mertzanis
Department of Informatics
Athens University of Economics & Business, Greece
la.mertzanis@gmail.com

Athina Panotopoulou
Department of Computer Science
Dartmouth College, USA
athina@cs.dartmouth.edu

Maria Skoularidou
MRC-Biostatistics Unit
University of Cambridge, UK
maria@mrc-bsu.cam.ac.uk

Ioannis Kontoyiannis
Department of Engineering
University of Cambridge, UK
i.kontoyiannis@eng.cam.ac.uk

***Abstract*—The identification of useful temporal dependence structure in discrete time series data is an important component of algorithms applied to many tasks in statistical inference and machine learning, and used in a wide variety of problems across the spectrum of biological studies. Most of the early statistical approaches were ineffective in practice, because the amount of data required for reliable modelling grew exponentially with memory length. On the other hand, many of the more modern methodological approaches that make use of more flexible and parsimonious models result in algorithms that do not scale well and are computationally ineffective for larger data sets. In this paper we describe a class of novel methodological tools for effective Bayesian inference for general discrete time series, motivated primarily by questions regarding data originating from studies in genetics and neuroscience.**

Our starting point is the development of a rich class of Bayesian hierarchical models for variable-memory Markov chains. The particular prior structure we adopt makes it possible to design effective, linear-time algorithms that can compute most of the important features of the relevant posterior and predictive distributions without resorting to Markov chain Monte Carlo simulation. The origin of some of these algorithms can be traced to the family of *Context Tree Weighting* (CTW) algorithms developed for data compression since the mid-1990s. We have used the resulting methodological tools in numerous application-specific tasks (including prediction, segmentation, classification, anomaly detection, entropy estimation, and causality testing) on data from different areas of application. The results obtained compare quite favourably with those obtained using earlier approaches, such as *Probabilistic Suffix Trees* (PST), *Variable-Length Markov Chains* (VLMC), and the class of *Markov Transition Distributions* (MTD).

I. Introduction

The starting point of many biological studies is the examination and analysis of discrete sequence data; and an important first step is often the identification of patterns of statistical structure in the data. Perhaps the most well studied examples of such problems arise from questions in bioinformatics, statistical genetics, and neuroscience.

Identifying and quantifying dependence is not only crucial in performing application-specific tasks (such as segmentation, compression, classification, and so on), but it is also of great intrinsic interest, as it reveals important features of the underlying biological data-generating mechanisms. It is in part for this purpose that the need for higher-memory Markov models in a variety of applications has been noted many times in the past; see, e.g., [18, 16, 5]. But higher-order Markov models are typically impossible to estimate and use in practice, as the number of parameters involved grows exponentially with memory length. In order to overcome this obstacle, numerous approaches have been developed, using more effective, lower-dimensional model classes; see, e.g., the broad discussions in [4, 2, 10, 7].

The class of variable-memory Markov chain models we consider here were first introduced in Rissanen’s celebrated work [22, 19], and have also been employed (in some cases with minor variations) in the Probabilistic Suffix Tree (PST) [23, 1, 7] and Variable-Length Markov Chain (VLMC) [4, 3, 17] literature. A different class of parsimonious models for higher-order memory modelling, the Markov Transition Distribution (MTD) model, was introduced by Raftery in 1985 [18] and explored further in [2].

The main contribution of this work is the development of a Bayesian framework for inference within the class of variable-memory Markov models. The main result we present is an algorithm which can be employed to determine exactly the maximum *a posteriori* probability (MAP) model. In fact, the k-MAPT algorithm allows us to find not only the model with the highest posterior probability, but also the $k \geq 1$ *a posteriori* most likely models. The complexity of the algorithm is only linear in the sample size, but grows rapidly with k.

The prior structure we use is motivated, in part, by considerations related to Rissanen’s *Minimum Description Length* (MDL) principle [21]. The starting point of our development is the family of *Context Tree Weighting* (CTW) [29] and *Context Tree Maximizing* (CTM) [30, 31, 27] algorithms.

Since its introduction and application to data compression, the CTW algorithm and its many variants have also been applied to numerous different statistical tasks, including prediction [32], segmentation [11]. reinforcement learning [25], network traffic analysis [12], turbo decoding [13], spam detection [24], and finance [6]. Biological applications can be found in several of the references cited above, as well as in [8, 9, 15].

In Section II we describe the hierarchical Bayesian model we consider, and Section III contains a description of the model selection algorithm k-MAPT. Finally, in Section IV we present brief experimental results illustrating the algorithm's performance on simulated data. Our results build on earlier work described in [14]. Theoretical results are stated without proofs; these will be given in a subsequent publication.

II. Tree Models and Prior Specification

Consider the class of dth order, homogeneous Markov chains, with values in the alphabet $A = \{0, 1, \dots, m-1\}$. The distribution of such a process $\{X_n\}$ will be described in terms of the conditional distribution of each X_i, $i \geq 1$, given the previous d symbols $X_{i-d}^{i-1} = (X_{i-d}, X_{i-d+1}, \dots, X_{i-1})$, where we write X_i^j for a vector of random variables $(X_i, X_{i+1}, \dots, X_j)$ and similarly $x_i^j \in A^{j-i+1}$ for a string $(x_i, x_{i+1}, \dots, x_j)$ representing a realization of the random variables X_i^j. The key element in specifying these distributions is a *context function* $C : A^d \to T$, which maps each length-d context x_{i-d}^{i-1} to a (typically strictly) shorter suffix $C(x_{i-d}^{i-1}) = x_{i-j}^{i-1}$ of itself, for some $0 \leq j \leq d$. Then the Markov property for $\{X_n\}$ takes the form:

$$P(x_1^n | x_{-d+1}^0) = \prod_{i=1}^n P(x_i | x_{i-d}^{i-1}) = \prod_{i=1}^n P(x_i | C(x_{i-d}^{i-1})).$$

The range T of C is a subset of $\cup_{i=0}^d A^i$, where we adopt the convention that the set A^0 contains only the empty string λ. We assume that the set T is *proper*, namely, that no element in T is a proper suffix of any other, and that if some $x_i^j = (x_i, x_{i+1}, \dots, x_j)$ is in the range of C, then so is every string of the form $(y, x_{i+1}, \dots, x_j)$, for all $y \in A$. Observe that, under these assumptions, the context function C is completely determined by its range T, since, for any string x_{i-d}^{i-1} there is exactly one element of T which is a suffix x_{i-j}^{i-1} of x_{i-d}^{i-1}.

To complete the specification of the distribution of $\{X_n\}$, in addition to the *context set* T, with every element $s \in T$ we associate a vector $\theta_s := (\theta_s(0), \theta_s(1), \dots, \theta_s(m-1))$, where the $\theta_s(j)$ are nonnegative and sum to one. Then,

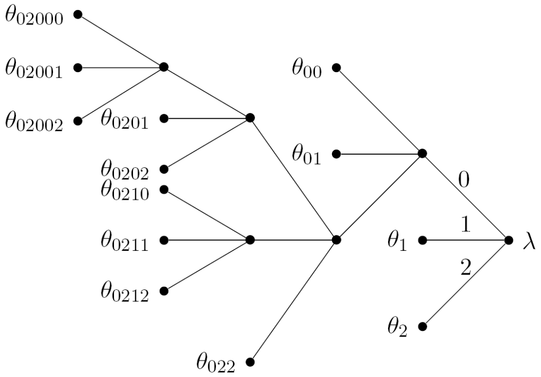
$$P(x_1^n | x_{-d+1}^0) = \prod_{i=1}^n \theta_{C(x_{i-d}^{i-1})}(x_i), = \prod_{s \in T} \prod_{j \in A} \theta_s(j)^{a_s(j)}, \quad (1)$$

where, in the last expression, instead of taking the product sequentially in time, we took the product over all possible contexts $s \in T$, and where the elements of each vector $a_s = (a_s(0), a_s(1), \dots, a_s(m-1))$ are,

$$a_s(j) = \text{\# times } j \in A \text{ follows context } s \text{ in } x_1^n. \quad (2)$$

We refer to the context set T as the *model* of $\{X_n\}$ and observe that it may be represented as a tree, where the context corresponding to the empty string λ is the root of the tree. We also refer to $\theta = \{\theta_s \; ; \; s \in T\}$ as the *parameters* of $\{X_n\}$.

Example. Consider a 5th order Markov chain on the alphabet $A = \{0, 1, 2\}$, defined by the context tree T shown below, and by a collection of (known) parameters $\theta = \{\theta_s \; ; \; s \in T\}$, where θ_s is a probability vector corresponding to leaf s in T.



The likelihood of an arbitrary string is easily computable explicitly via (1). For example, with $d = 5$ and $n = 12$, the string,

$$\underbrace{1, 0, 2, 1, 1}_{x_{-4}^0}, \underbrace{0, 0, 1, 2, 1, 0, 2, 0, 0, 2, 0, 1}_{x_1^{12}}$$

has probability given by (1) as:

$$\theta_1(0)^2 \cdot \theta_1(2) \cdot \theta_2(0)^2 \cdot \theta_2(1) \cdot \theta_{01}(0) \cdot \theta_{01}(2)$$
$$\cdot \theta_{00}(1) \cdot \theta_{00}(2) \cdot \theta_{0201}(0) \cdot \theta_{02002}(1).$$

Model prior. Given a fixed depth D and an arbitrary $\beta \in (0, 1)$, we define a prior distribution on models T (proper context sets, or the corresponding trees) of maximal depth no more than D, as,

$$\pi(T) := \pi_D(T) := \pi_D(T; \beta) := \alpha^{|T|-1} \beta^{|T| - L_D(T)},$$

where $\alpha := (1-\beta)^{1/(m-1)}$, $|T|$ denotes the number of leaves of T, and $L_D(T)$ denotes the number of leaves T has at depth D. It is not hard to show that $\pi_D(T, \beta)$ indeed defines a probability probability distribution on the set $\mathcal{T}(D)$ of all proper context trees of depth no greater than D.

Prior on θ. Given a model T, we define a prior distribution on the probability vectors $\theta = \{\theta_s \; ; \; s \in T\}$ on the leaves s of the context tree T: We place an independent Dirichlet$(1/2, \dots, 1/2)$ distribution on each θ_s so that, $\pi(\theta|T) = \prod_{s \in T} \pi(\theta_s)$, where,

$$\pi(\theta_s) = \frac{\Gamma(m/2)}{\pi^{m/2}} \prod_{j=0}^{m-1} \theta_s(j)^{-\frac{1}{2}}.$$

Finally, given T and the associated parameters θ, the likelihood of the observations is given as in (1),

$$P(x_1^n | x_{-d+1}^0, \theta, T) = \prod_{s \in T} \prod_{j=0}^{m-1} \theta_s(j)^{a_s(j)},$$

where, the $a_s(j)$ are defined in (2) and, by convention, when we write $\sum_{s \in T}$ or $\prod_{s \in T}$, we take the corresponding sum or product over all the *leaves* s of the tree, not all its nodes. Also, in order to avoid cumbersome notation, in what follows we often write x for the string x_1^n and suppress the dependence on its initial context x_{-d+1}^0, so that, for example, we denote,

$$P(x, \theta | T) = P(x_1^n, \theta | x_{-d+1}^0, T).$$

Choice of β. In order to maintain an "exponential penalization" of large models, the value of β needs to be adjusted for different alphabet sizes m. A simple calculation suggests the following practical rule: When m is significantly larger than 2, then β should be chosen to be close to $1-2^{-m+1}$ so that $\alpha \approx 1/2$; and when m is equal to 2 or is not much larger (so that $2^{-1/(m-1)}$ is not close to 1), then β should either be taken $\beta \approx 1-2^{-m+1}$ as before, or $\beta = 1/2$ for simplicity.

Marginal likelihood. An important and useful property of this prior specification is that the parameters θ can easily be integrated out: With the vectors a_s as in (2):

Lemma 2.1: The *marginal likelihood* $P(x|T)$ of the observations x given a model T is,

$$P(x|T) = \int P(x|\theta, T)\pi(\theta|T)d\theta = \prod_{s\in T} P_e(a_s),$$

with,

$$P_e(a_s) := \frac{\prod_{j=0}^{m-1}[(1/2)(3/2)\cdots(a_s(j)-1/2)]}{(m/2)(m/2+1)\cdots(m/2+M_s-1)}, \qquad (3)$$

where $M_s := a_s(0)+a_s(1)+\cdots+a_s(m-1)$.

In terms of inference, the more interesting quantity is the model posterior distribution,

$$\pi(T|x) = \frac{P(x|T)\pi(T)}{P(x)}.$$

As usual, the main obstacle in the computation of $\pi(T|x)$ is the appearance of $P(x)$, which can be expressed as the weighted mean of the marginal likelihoods $P(x|T)$. We refer to $P(x)$ as the *mean marginal likelihood* of x, and denote it:

$$P_D^*(x) := \sum_{T\in\mathcal{T}(D)} \pi_D(T)P(x|T).$$

The difficulty in computing the mean marginal likelihood $P_D^*(x)$ comes from the fact that the class of variable-memory models is enormously rich, even for moderate (or even small) alphabet sizes m and tree depths D: The number $|\mathcal{T}(D)|$ of models in the collection $\mathcal{T}(D)$ of all proper context trees of depth no greater that D grows *doubly exponentially* in D.

Nevertheless, a modification of the CTW algorithm (not discussed further here) makes it possible to compute the mean marginal likelihood $P_D^*(x)$ precisely, without resorting to simulation. Moreover, the k-Maximum A Posteriori Probability Tree (k-MAPT) algorithm (described next) allows us to compute the k *a posteriori* most likely tree models.

III. The k-MAPT Algorithm

The k-MAPT algorithm takes as input: Observations x_{-D+1}^n; the size of the alphabet m; the maximum context depth D; the value of the prior parameter β; and the number k of the k *a posteriori* most likely models to be determined.

For the sake of clarity of exposition, we describe the k-MAPT algorithm in three stages.

Stage I. Preliminary steps.

(a) Build an m-ary tree T_{MMLA}, whose leaves are all the contexts x_{i-D}^{i-1}, $1 \le i \le n$, that appear in the observations x_{-D+1}^n. If some node s of T_{MMLA} is at depth $d < D$ and some but not all of its children are in T_{MMLA}, then add all its remaining children as well, so that T_{MMLA} is a proper tree.

(b) Compute the count vector a_s as in (2), at *each* node s of the tree T_{MMLA} (not only at the leaves), and note that a_s will be the all-zero vector for the additional leaves included in the last step of (a).

(c) Compute the probability $P_{e,s} := P_e(a_s)$ given by (3), at each node s of the tree T_{MMLA}, with the convention that $P_e(a_s) = 1$ when a_s is the all-zero count vector.

Stage II. Idealized k-MAPT algorithm.

(i) Let T_{MMLA} be the *complete* m-ary tree at depth D; compute the count vectors a_s and probabilities $P_{e,s} = P_e(a_s)$ at all nodes s of T_{MMLA} as in the preliminary steps (b), (c).

(ii) Starting at the leaves and proceeding towards the root, at each node s we compute a list of k maximal probabilities $P_{m,s}^{(i)}$ and k position vectors $c_s^{(i)} = (c_s^{(i)}(0), c_s^{(i)}(1), \dots, c_s^{(i)}(m-1))$, for $i = 1, 2, \dots, k$, where each $c_s^{(i)}(j)$ is an integer between 0 and k, recursively as follows.

(iia) At each leaf s, we let $P_{m,s}^{(1)} = P_{e,s}$ and $c_s^{(1)} = (0, 0, \dots, 0)$, where the all-zero vector $c_s^{(i)}$ indicates that $P_{m,s}^{(1)}$ corresponds to the value of $P_{e,s}$ and does not depend on the children of s (since there are none). For $i = 2, 3, \dots, k$, we leave $P^{(i)}$ and $c_s^{(i)}$ undefined.

(iib) At each node s having only m descendants (which are necessarily leaves), we compute the probability-position vector pairs $\beta P_{e,s}$, $(0, 0, \dots, 0)$ and $(1-\beta)\prod_{j=0}^{m-1} P_{m,sj}^{(1)}$, $(1, 1, \dots, 1)$ (where the all-1 vector indicates that the latter probability only depends on the first maximal probability of each of the children), and sort them as $P_{m,s}^{(1)}$, $c_s^{(1)}$ and $P_{m,s}^{(2)}$, $c_s^{(2)}$ in order of decreasing probability. For $i = 3, 4, \dots, k$, we leave $P^{(i)}$ and $c_s^{(i)}$ undefined.

(iic) A general internal node s has m children, where each child sj has a list of k_j (for some $1 \le k_j \le k$) probability-vector pairs $P_{m,sj}^{(i)}, c_{sj}^{(i)}$, $1 \le i \le k_j$. We compute the probability $\beta P_{e,s}$ with associated position vector $(0, 0, \dots, 0)$, and all possible probability-position vector pairs,

$$(1-\beta)\prod_{j=0}^{m-1} P_{m,sj}^{(i_j)}, \quad (i_0, i_1, \dots, i_{m-1}),$$

for all possible combinations of indices $1 \le i_j \le k_j$ for $0 \le j \le m-1$. We then sort these $k' = 1 + k_0 \times k_1 \times \cdots \times k_{m-1}$ probabilities in order of decreasing probability, and rename the top k of them as $P_{m,s}^{(i)}$, for $i = 1, 2, \dots, k$, together with their associated position vectors $c_s^{(i)}$. [Of course, if $k' < k$, after sorting we leave the remaining $k - k'$ probability-position vector pairs undefined.]

(iii) Having determined all maximal probabilities $P_{m,s}^{(i)}$ for all nodes s and $1 \le i \le k$, we now determine the "top k" trees $T_1^*, T_2^*, \dots, T_k^*$ from the corresponding position vectors $c_s^{(i)}$. For each i we repeat the following process, starting at the root and proceeding until all available nodes of the tree T_{MMLA} have been exhausted.

(iiia) *Depth* $d = 0$. At the root node λ, we examine $c_\lambda^{(i)}$. If it is the all-zero vector, then T_i^* is the tree consisting of the root node only. Otherwise, we add to T_i^* the branch of m children starting at the root, and proceed to examine each of the nodes corresponding to the m children recursively.

(iiib) *Depth* $d = 1$. Reaching node $s = j$ corresponding to the jth child of the root, means that $t = c_\lambda^{(i)}(j)$ is nonzero. We examine $c_s^{(t)}$: If it is the all-zero vector, then we prune from T_i^* all the descendants of s and move to the next unexamined node; otherwise, we add to T_i^* the branch of m children starting at s, and proceed to examine each of the nodes corresponding to the m children recursively.

(iiic) *General depth* $1 \leq d \leq D-1$. Reaching a node sj at depth d from its parent node s means that we decided to visit sj because $t = c_s^{(u)}(j)$ is nonzero for the appropriate index u (corresponding to the position vector $c_s^{(u)}$ that was examined at node s). We examine $c_{sj}^{(t)}$: If it is the all-zero vector, then we prune from T_i^* all the descendants of sj and move to the next unexamined node; otherwise, we add to T_i^* the branch of m children starting at sj, and proceed to examine each of the nodes corresponding to the m children recursively.

(iiid) *Depth* $d = D$. Reaching a node s at depth D means we have reached a leaf of T_{MMLA}, so we simply add s to T_i^* and proceed to the next unexamined node.

(iv) Output the k resulting trees T_i^* and the k maximal probabilities at the root, $P_{m,\lambda}^{(i)}$, $i = 1, 2, \ldots, k$.

Stage III. Actual k-MAPT algorithm

(i) Initially, perform two preprocessing steps:

(ia) Execute the first two steps (i) and (ii) of the idealized k-MAPT algorithm on the complete m-ary tree of depth D, with all the count vectors a_s assumed to be equal to zero, $a_s = (0, 0, \ldots, 0)$ for all s. Since all nodes at the same depth are identical, this process can be carried out effectively, by performing the relevant computations only at a single node (instead of all m^d nodes) for each depth $0 \leq d \leq D$.

(ib) Build the tree T_{MMLA} and compute the count vectors a_s and the probabilities $P_{e,s} = P_e(a_s)$ at all nodes s of T_{MMLA}, as in preliminary steps (a)–(c).

(ii) Starting at the leaves and proceeding towards the root, at each node s we compute a list of k maximal probabilities $P_{m,s}^{(i)}$ and k position vectors $c_s^{(i)} = (c_s^{(i)}(0), c_s^{(i)}(1), \ldots, c_s^{(i)}(m-1))$, for $i = 1, 2, \ldots, k$, recursively as follows.

(iia) At each leaf s at depth D, with a nonzero count vector a_s, let $P_{m,s}^{(1)} = P_{e,s}$ and $c_s^{(1)} = (0, 0, \ldots, 0)$. For $i = 2, 3, \ldots, k$, we leave $P^{(i)}$ and $c_s^{(i)}$ undefined.

(iib) Similarly, at each leaf s at depth D, with an all-zero count vector a_s, let $P_{m,s}^{(1)} = P_{e,s} = 1$, $c_s^{(1)} = (0, 0, \ldots, 0)$, and for $i = 2, 3, \ldots, k$, leave $P^{(i)}$ and $c_s^{(i)}$ undefined.

(iic) At each leaf s at depth $d < D$, we let the list of the maximal probabilities $P_{m,s}^{(i)}$ and position vectors $c_s^{(i)}$ of s be those that are computed for a node at depth d in the preprocessing stage (ia).

(iid) Continue with steps (iib) and (iic) as in the idealized version of k-MAPT.

(iii) We perform the same steps as described in $(iiia)$–$(iiid)$ of the idealized version, with the following addition:

(iiie) While examining a node s at depth $0 < d < D$ in step $(iiib)$ or $(iiic)$ of the idealized algorithm, we may reach a point where the algorithm dictates that we examine its m children, when these children are *not* included in the tree T_{MMLA}. In that case, we add them to T_{MMLA}, and we define their corresponding maximal probabilities and position vectors according to the initialization described in step (iib) or (iic) above, depending on whether $d = D-1$ or $d < D-1$, respectively.

(iv) As in the idealized version, output the k resulting trees T_i^* and the k maximal probabilities at the root, $P_{m,\lambda}^{(i)}$, $i = 1, 2, \ldots, k$.

Theorem 3.1: For any $\beta \geq 1/2$, the trees $T_1^*, T_2^*, \ldots, T_k^*$ produced by the k-MAPT algorithm are indeed the k *a posteriori* most likely trees.

Moreover, the ith maximal probability at the root satisfies:

$$P_{m,\lambda}^{(i)} = P(x|T_i^*)\pi_D(T_1^*) = P(x, T_i^*), \quad i = 1, 2, \ldots, k.$$

Remarks.

1. It is not hard to show that the time-complexity of the k-MAPT algorithm is $O(nmk^mD)$. Hence, the complexity increases dramatically as we require more information about the "top" of the posterior $\pi(T|x)$ in model space. Also, note the memory required by the k-MAPT algorithm is $O(nmDk)$.

2. As mentioned earlier, in addition to identifying the k *a posteriori* most likely models, it is also possible to compute the mean marginal likelihood $P_D^*(x)$ of the observations, which makes it possible to then compute the Bayes factors and posterior odds for different models T. This is illustrated in the example presented in the next section.

3. Finally, some bibliographical comments are in order. The origin of some of the ideas behind the k-MAPT algorithm can be traced to the CTW algorithm as described in the unpublished manuscript [28]. A special case of the $k = 1$ version of the k-MAPT algorithm was introduced in [28, 30, 31]. The k-MAPT algorithm and the result of Theorem 3.1 are both new, although they were, in part, motivated by some remarks in [27].

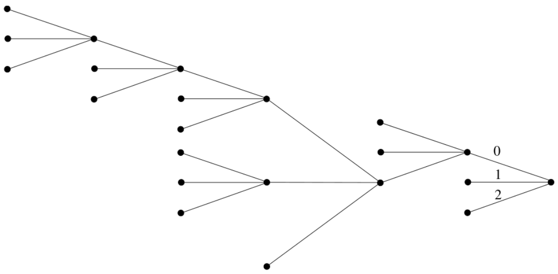
As mentioned earlier, Rissanen's early results in [19, 20] and their extensions in [26] form the real foundation of the present development. There, in addition to variable-memory Markov models, Rissanen also introduced the CONTEXT algorithm for fitting such a model to data. A different asymptotic analysis of CONTEXT was later carried out in the VLMC paper [4], and a more detailed examination of the algorithm in connection with model selection was given in [3]. In all these works, no Bayesian or other finite-sample interpretation is given for the resulting model. Finally, we mention that in the first PST paper [23], a different procedure called *Learn-PSA*, similar in spirit to CONTEXT, was described for estimating a variable-memory Markov model; see also the relevant results in [1].

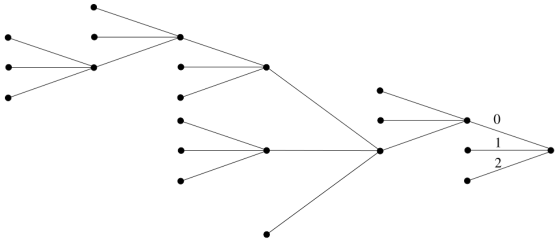
IV. A Simulation Example

We revisit the variable-memory chain $\{X_i\}$ on the alphabet $A = \{0, 1, 2\}$ discussed in the Introduction. The top $k = 3$ *a posteriori* most likely models were obtained by the k-MAPT algorithm from $n = 10000$ simulated samples. The algorithm's parameters were $m = 3$, $\beta = 3/4$, and maximum depth $D = 10$.

The most likely model was found to be the true underlying model T_1^*, shown in the Introduction; its posterior probability $\pi(T_1^*|x) \approx 0.368$ while its prior $\pi(T_1^*) \approx 5.8 \times 10^{-6}$. For the second and third most likely models T_2^*, T_3^* shown below, we have the posterior odds, $\pi(T_1^*|x)/\pi(T_2^*|x) \approx 6.2903$ and $\pi(T_1^*|x)/\pi(T_3^*|x) \approx 8.822$.

It is worth perhaps noting that the correct model is identified, based on only $n = 10^4$ samples, among well over 10^{13000} possible models of maximum depth at most 10.





References

[1] G. Bejerano and G. Yona. Variations on probabilistic suffix trees: Statistical modelling and prediction of protein families. *Bioinformatics*, 17(1):23–43, 2001.

[2] A. Berchtold and A.E. Raftery. The mixture transition distribution model for high-order Markov chains and non-Gaussian time series. *Statistical Science*, 17(3):328–356, 2002.

[3] P. Bühlmann. Model selection for variable length Markov chains and tuning the context algorithm. *Annals of the Institute of Statistical Mathematics*, 52(2):287–315, 2000.

[4] P. Bühlmann and A.J. Wyner. Variable length Markov chains. *Ann. Stat.*, 27(2):480–513, 1999.

[5] W.K. Ching, E.S. Fung, and M.K. Ng. Higher-order Markov chain models for categorical data sequences. *Naval Research Logistics (NRL)*, 51(4):557–574, 2004.

[6] P. Fiedor. Frequency effects on predictability of stock returns. In *Computational Intelligence for Financial Engineering & Economics, 2104 IEEE Conference on*, pages 247–254, 2014.

[7] A. Gabadinho and G. Ritschard. Analyzing state sequences with probabilistic suffix trees: The PST R package. *Journal of Statistical Software*, 72(3):1–39, 2016.

[8] Y. Gao, I. Kontoyiannis, and E. Bienenstock. Entropy estimation: Simulation, theory, and a case study. In *IEEE Inform. Theory Workshop*, Punta del Este, Uruguay, March 2006.

[9] Y. Gao, I. Kontoyiannis, and E. Bienenstock. From the entropy to the statistical structure of spike trains. In *IEEE Int. Symp. on Inform. Theory*, Seattle, WA, July 2006.

[10] A. Garivier and F. Leonardi. Context tree selection: A unifying view. *Stochastic Processes and their Applications*, 121(11):2488–2506, 2011.

[11] R. Gwadera, A. Gionis, and H. Mannila. Optimal segmentation using tree models. *Knowl. Inf. Syst.*, 15(3):259–283, May 2008.

[12] B. Hullár, S. Laki, and A. Gyorgy. Early identification of peer-to-peer traffic. In *Communications (ICC), 2011 IEEE International Conference on*, pages 1–6. IEEE, 2011.

[13] K. Kim, N. Kalantarova, S.S. Kozat, and A.C. Singer. Linear MMSE-optimal turbo equalization using context trees. *IEEE Transactions on Signal Processing*, 61(12):3041–3055, 2013.

[14] I. Kontoyiannis, A. Panotopoulou, and M. Skoularidou. Bayesian inference for discrete time series via tree weighting. *IEEE Inform. Theory Workshop*, Lausanne, Sept. 2012.

[15] C.J. Kusters and T. Ignatenko. DNA sequence modelling based on context trees. In *Proc. 5th Jt. WIC/IEEE Symp. Inf. Theory Signal Process. Benelux*, pages 96–103, 2015.

[16] R. Langeheine and F. Van de Pol. Fitting higher order Markov chains. *Methods of Psychological Research Online*, 5(1):32–55, 2000.

[17] M. Mächler. VLMC: Variable length Markov chains. *R package version 1.4-1*, October 2015.

[18] A.E. Raftery. A model for high-order Markov chains. *Journal of the Royal Statistical Society. Series B (Methodological)*, 47(3):528–539, 1985.

[19] J. Rissanen. A universal prior for integers and estimation by minimum description length. *Ann. Statist.*, 11(2):416–431, 1983.

[20] J. Rissanen. Complexity of strings in the class of Markov sources. *Information Theory, IEEE Transactions on*, 32(4):526–532, July 1986.

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

[22] J. Rissanen and G. Langdon. Universal modelling and coding. *IEEE Transactions on Information Theory*, 27(1):12–23, 1981.

[23] D. Ron, Y. Singer, and N. Tishby. The power of amnesia: Learning probabilistic automata with variable memory length. *Machine Learning*, 25:117–149, 1996.

[24] I. Santos, I. Minambres-Marcos, C. Laorden, P. Galán-García, A. Santamaría-Ibirika, and P.G. Bringas. Twitter content-based spam filtering. In *International Joint Conference SOCO13-CISIS13-ICEUTE13*, pages 449–458, 2014.

[25] J. Veness, K.S. Ng, M. Hutter, and D. Silver. Reinforcement learning via AIXI approximation. *arXiv preprint arXiv:1007.2049*, July 2010.

[26] M.J. Weinberger, J. Rissanen, and M. Feder. A universal finite memory source. *Information Theory, IEEE Transactions on*, 41(3):643–652, May 1995.

[27] F.M.J. Willems, A. Nowbahkt-Irani, and P.A.J. Volf. Maximum a-posteriori probability tree models. In *4th International ITG Conference on Source and Channel Coding*, Berlin, Germany, February 2002.

[28] F.M.J. Willems, Y.M. Shtarkov, and T.J. Tjalkens. Context tree weighting: Basic properties. Unpublished manuscript, summer 1993. Available online at: `www.sps.ele.tue.nl/members/F.M.J.Willems/`.

[29] F.M.J. Willems, Y.M. Shtarkov, and T.J. Tjalkens. Context tree weighting: Basic properties. *IEEE Trans. Inform. Theory*, 41(3):653–664, 1995.

[30] F.M.J. Willems and P.A.J. Volf. Context maximizing: Finding MDL decision trees. In *15th Symposium on Information Theory in the Benelux*, Louvain-la-Neuve, Belgium, May 1995.

[31] F.M.J. Willems and P.A.J. Volf. A study of the context tree maximizing method. In *16th Symposium on Information Theory in the Benelux*, Nieuwerkerk Ijsel, The Netherlands, May 1995.

[32] J. Ziv and N. Merhav. On context-tree prediction of individual sequences. *IEEE Trans. Inform. Th.*, 53(5):1860–1866, 2007.