Maximum a posteriori probability tree models

Frans M.J. Willems, Ali Nowbakht, and Paul A.J. Volf

Abstract— **The context-tree weighting method (Willems, Shtarkov, and Tjalkens [1995]) can be used to compress sequences generated by tree sources. Its redundancy behavior is optimal in the sense that Rissanen's lower bound [1984] is achieved. Here we study some questions related to the context-tree weighting method. First we stress again that the a priori distribution over all tree models that is mainly considered in the basic CTW paper is not the only one that can be used in context-tree weighting. E.g. uniform weighting over all tree models is also realizable. (a) After observing a source sequence, one could be interested in the a posteriori probability of a specified model. We derive a very simple procedure to determine such a probability. (b) In a two-pass source coding procedure after observing the source sequence, the "best" model is determined and described to the decoder. After that the source sequence is encoded given this best model and also transmitted to the decoder. We consider a procedure that finds the model minimizing the total description length for a priori distributions over the tree models other than the standard one. The best model that is found is the maximum a posteriori probability (MAP) model. (c) A two-pass method decoder can have a bounded complexity. E.g. assume that it can not process tree models with more than C leaves. We describe a method that determines the best model having a number of leaves not exceeding threshold C. (d) Finally we develop a procedure to determine the N models with the largest a posteriori probabilities.**

I. Introduction: Context-tree weighting

A. Arithmetic Coding

Denote the binary sequence $x_1x_2\cdots x_T$ by x_1^T. Given a coding distribution $P_c(x_1^T)$ over all binary sequences of length T, the Elias algorithm (see e.g. Jelinek [1]) generates codewords (satisfying the prefix condition) with lengths

$$L(x_1^T) < \log_2 \frac{1}{P_c(x_1^T)} + 2. \tag{1}$$

Implementations of this method are called arithmetic coding methods (e.g. Rissanen [2], Pasco [3]). The codeword length that we obtain in this way are at most two binary digits longer than the length that we desire (i.e. the ideal codeword length $-\log_2 P_c(x_1^T)$). We say that the *coding redundancy* is smaller than 2. Therefore universal source coding is mainly concerned with finding good coding distributions.

F. Willems and A. Nowbakht are with the Electrical Engineering Department, Eindhoven University of Technology, Eindhoven, The Netherlands.

P. Volf is with Philips Semiconductors, Nijmegen, The Netherlands.

The research of A. Nowbakht is supported by Technologiestichting STW under project EEL4643.

B. Krichevski-Trofimov estimator

The actual probability $\Pr\{X_1^t = x_1^t\}$ of a source sequence x_1^t for $t = 1, T$ is denoted as $P_a(x_1^t)$. For an independent identically distributed (i.i.d.) binary source with an unknown parameter $\theta = P_a(1)$ we should use

$$P_e(a,b) = \frac{(a-\frac{1}{2})(a-\frac{3}{2})\cdots\frac{1}{2}(b-\frac{1}{2})(b-\frac{3}{2})\cdots\frac{1}{2}}{(a+b)(a+b-1)\cdots 1} \tag{2}$$

as coding probability for a sequence containing a zeroes and b ones. This assignment is called the Krichevsky-Trofimov [4] estimate. Consider a sequence x_1^T with a zeroes and b ones, then from (1) we may conclude that

$$L(x_1^T) < \log_2 \frac{1}{P_e(a,b)} + 2. \tag{3}$$

Define the individual redundancy for sequence x_1^T as

$$\rho(x_1^T) \triangleq L(x_1^T) - \log_2 \frac{1}{P_a(x_1^T)}, \tag{4}$$

then this redundancy for a sequence x_1^T with a zeroes and b ones satisfies

$$\begin{aligned} \rho(x_1^T) &< \log_2 \frac{1}{P_e(a,b)} + 2 - \log_2 \frac{1}{(1-\theta)^a\theta^b} \\ &= \log_2 \frac{(1-\theta)^a\theta^b}{P_e(a,b)} + 2 \\ &\leq \frac{1}{2}\log_2(a+b) + 3, \end{aligned} \tag{5}$$

where we used lemma 1 of Willems, Shtarkov and Tjalkens [5] to upper bound the $\log_2 P_a/P_e$-term. This term, called the *parameter redundancy*, is never larger than $\frac{1}{2}\log_2(a+b)+1$. Hence the individual redundancy is not larger than $\frac{1}{2}\log_2 T + 3$ for all x_1^T and all $\theta \in [0,1]$. Therefore this estimator is asymptotically optimal (see Rissanen [6]).

C. Tree Sources

Consider figure 1. For a *tree source* the probability $P_a(X_t = 1|\cdots, x_{t-2}, x_{t-1})$ is determined by starting in the root λ of the tree and moving along the path $x_{t-1}, x_{t-2}, \cdots$ until a leaf of the tree is reached. In this leaf s we find the desired probability (parameter) θ_s. The suffix set, or tree, $\mathcal{S}$ is called the *model* of the source.

For the source in figure 1 the (conditional) probability of the source generating the sequence 01101 given the past symbols $\cdots 010$ is:

$$\begin{aligned} P_a(01101|\cdots 010) &= (1-\theta_{10})\theta_{00}\theta_1(1-\theta_1)\theta_{10} \\ &= 0.00945. \end{aligned}$$

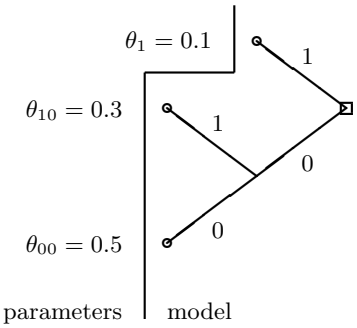

Fig. 1. Model (suffix set) and parameters.

D. Unknown Parameters, Known Model

The source model (tree) $\mathcal{S}$ partitions the source sequence in i.i.d. sub-sequences, one for each leaf $s \in \mathcal{S}$. If the parameters of the source are unknown we can use the Krichevski-Trofimov estimator for each of these sub-sequences. E.g. for $\mathcal{S} = \{00, 10, 1\}$ we get:

$$P_e(x_1^T|\mathcal{S}) = P_e(a_{00}, b_{00}) \cdot P_e(a_{10}, b_{10}) \cdot P_e(a_1, b_1). \tag{6}$$

These estimated probabilities should be used as coding probabilities. In general we obtain for the estimated probabilities for tree-model $\mathcal{S}$:

$$P_e(x_1^T|\mathcal{S}) = \prod_{s\in\mathcal{S}} P_e(a_s, b_s), \tag{7}$$

where a_s is the number of zeroes in the subsequence of x_1^T corresponding to leaf s, and b_s the number of ones in this subsequence.

This results in a (parameter plus coding) redundancy

$$\begin{aligned}\rho(x_1^T) &< \log_2 \frac{1}{P_e(x_1^T|\mathcal{S})} + 2 - \log_2 \frac{1}{P_a(x_1^T)} \\ &\leq \left(\frac{|\mathcal{S}|}{2} \log_2 \frac{T}{|\mathcal{S}|} + |\mathcal{S}|\right) + 2, \end{aligned} \tag{8}$$

for $T \geq |\mathcal{S}|$. Moreover $\rho(x_1^T) \leq T + 2$ for $T < |\mathcal{S}|$.

E. Weighting

Consider two sources. For the first source we should use coding distribution $P_c^1(x_1^T)$ to obtain good performance, whatever we mean by that. For the second source we should use distribution $P_c^2(x_1^T)$. If we need a single code that is good for both sources then

$$P_w(x_1^T) = \frac{P_c^1(x_1^T) + P_c^2(x_1^T)}{2} \tag{9}$$

would be a good coding distribution. It leads to codeword length

$$\begin{aligned} L(x_1^T) &< \log_2 \frac{2}{P_c^1(x_1^T) + P_c^2(x_1^T)} + 2 \\ &\leq \log_2 \frac{1}{P_c^i(x_1^T)} + 3, \text{ for } i = 1, 2. \end{aligned} \tag{10}$$

We loose at most one binary digit with this weighting technique!

F. Unknown Model

Suppose that the actual source model is unknown, but that its depth is not larger than D. Note that to each context s, there corresponds a substring of $x_1 \cdots x_T$ of symbols that are produced by the source following this context s. Let a_s be the number of zeroes in this subsequence and b_s the number of ones. The structure containing a node for all contexts s having depth not larger than D is called the context tree $\mathcal{T}_\mathcal{D}$. A good estimator for the subsequence corresponding to a context s at depth D is $P_w^s = P_e(a_s, b_s)$. Now let the depth of s be less than D. For the subsequence corresponding to s we have two alternatives now. We can use the estimator for the entire subsequence corresponding to s which results in $P_e(a_s, b_s)$ or we can split up this subsequence in two sub-sub-sequences and use the product $P_w^{0s} P_w^{1s}$ of the "weighted" probabilities P_w^{0s} and P_w^{1s} as estimator. If we weight these two alternatives we obtain

$$P_w^s = \begin{cases} \frac{1}{2} P_e(a_s, b_s) + \frac{1}{2} P_w^{0s} P_w^{1s} & \text{if depth}(s) < D, \\ P_e(a_s, b_s) & \text{otherwise.} \end{cases} \tag{11}$$

The weighted probability P_w^λ in the root can now be used as coding probability for the entire sequence x_1^T.

G. Performance

The individual redundancy $\rho(x_1^T)$ with respect to the actual source for sequence x_1^T can be upper bounded by

$$\begin{aligned} \rho(x_1^T) &= L(x_1^T) - \log_2 \frac{1}{P_a(x_1^T)} \\ &< 2|\mathcal{S}| - 1 + \frac{|\mathcal{S}|}{2} \log_2 \frac{T}{|\mathcal{S}|} + |\mathcal{S}| + 2, \end{aligned} \tag{12}$$

for $T \geq |\mathcal{S}|$. Moreover $\rho(x_1^T) \leq 2|\mathcal{S}| - 1 + T + 2$ for $T < |\mathcal{S}|$.

The three terms in this bound are the cost of specifying the model i.e. $2|\mathcal{S}| - 1$, the cost of specifying the parameters which is $\frac{|\mathcal{S}|}{2} \log_2 \frac{T}{|\mathcal{S}|} + |\mathcal{S}|$ and the loss of 2 binary digits due to arithmetic coding.

Observe that an identical bound holds for the redundancy with respect to any other tree source with depth $\leq D$.

II. Uniform weighting over all models

The coding distribution P_w^λ defined recursively by (11) yields model cost not more $2|\mathcal{S}|-1$, i.e. linear in $|\mathcal{S}|$ (If we assume that $\mathcal{S}$ has leaves at depth D this upper bound can be improved.) This is achieved by giving equal weight to $P_e(a_s, b_s)$ and $P_w^{0s} P_w^{1s}$ in the expression for P_w^s in each (internal) node s in

the context tree. It is very well possible however to assume that these weights are not equal, and even to suppose that they are different for different nodes s. In this section (as in [5]) we will assume that the weigthing in a node s depends on the depth $l(s)$ of this node in the context tree. Hence for $s \in \mathcal{T}_{\mathcal{D}}$

$$P_w^s \triangleq \alpha_{l(s)} P_e(a_s, b_s) + (1 - \alpha_{l(s)}) P_w^{0s} P_w^{1s}, \quad (13)$$

with $\alpha_D = 1$. Now note that each model can be regarded as the empty (memoryless) model $\{\lambda\}$ to which a number of nodes may have been added. The cost of the empty model is $-\log_2 \alpha_0$, we can also say that the model cost of the first parameter is $-\log_2 \alpha_0$ bits. Our objective is now that, if we add a new node (parameter) to a model, the model cost does not increase, no matter at what level d we add this node. In other words

$$\frac{1 - \alpha_d}{\alpha_d} \cdot \alpha_{d+1}^2 = 1, \quad (14)$$

for $0 \leq d \leq D - 1$, or consequently

$$\left(\frac{1}{\alpha_d}\right) = \left(\frac{1}{\alpha_{d+1}}\right)^2 + 1. \quad (15)$$

We obtain that all models that fit into $\mathcal{T}_{\mathcal{D}}$ have *equal a priori probabilities*, for $(\alpha_{D-1})^{-1} = 2, (\alpha_{D-2})^{-1} = 5, (\alpha_{D-3})^{-1} = 26, (\alpha_{D-4})^{-1} = 677$, etc. For $D = 4$ this yields a cost of $-\log_2 \alpha_0 = \log_2 677 = 9.403$ binary digits for all 677 models in $\mathcal{T}_4$ and 150.448 binary digits for $D = 8$, etc. Note that the number of models in $\mathcal{T}_{\mathcal{D}}$ grows very fast with D. Incrementing the depth D of the context tree by one results roughly in squaring the number of models that fit into $\mathcal{T}_{\mathcal{D}}$. The context-tree weighting method is working on all these models simultaneously in a very efficient way!

III. A posteriori probabilities

A. Standard weighting

Consider a sub-(tree-)model $\mathcal{S}_s$ (a "complete" set of strings all having a common suffix s, but with no string being the suffix of any other string in the set) rooted in node s of $\mathcal{T}_{\mathcal{D}}$ and fitting in the context tree $\mathcal{T}_{\mathcal{D}}$. Then we define the "conditional" probability of the sub-tree $\mathcal{S}_s$ given x_1^T as

$$Q_w^s(\mathcal{S}_s) \triangleq \frac{2^{-\Gamma_D(\mathcal{S}_s)} \prod_{s \in \mathcal{S}_s} P_e(a_s, b_s)}{P_w^s}, \quad (16)$$

where a_s and b_s are the number of zeroes respectively ones that follow context s in sequence x_1^T given a certain past. Moreover the cost of sub-model $\mathcal{S}_s$ is defined as

$$\Gamma_D(\mathcal{S}_s) \triangleq 2|\mathcal{S}_s| - 1 - |\{s \in \mathcal{S}_s, \text{depth}(s) = D\}|, \quad (17)$$

where it is assumed that sub-model $\mathcal{S}_s$ fits into $\mathcal{T}_{\mathcal{D}}$.

Now if $|\mathcal{S}_s| > 1$, note that the node s can not be at level D then, we can split up the sub-model $\mathcal{S}_s$ into a sub-model $\mathcal{S}_{0s}$ and a sub-model $\mathcal{S}_{1s}$ and we obtain for the conditional probability

$$\begin{aligned} Q_w^s(\mathcal{S}_s) &= \frac{2^{-\Gamma_D(\mathcal{S}_{0s})} \prod_{s \in \mathcal{S}_{0s}} P_e(a_s, b_s)}{P_w^{0s}} \\ &\quad \cdot \frac{2^{-\Gamma_D(\mathcal{S}_{1s})} \prod_{s \in \mathcal{S}_{1s}} P_e(a_s, b_s)}{P_w^{1s}} \\ &\quad \cdot \frac{P_w^{0s} P_w^{1s}}{P_e(a_s, b_s) + P_w^{0s} P_w^{1s}} \\ &= Q_w^{0s}(\mathcal{S}_{0s}) Q_w^{1s}(\mathcal{S}_{1s}) \frac{1}{\beta_s + 1}. \end{aligned} \quad (18)$$

where

$$\beta_s \triangleq \frac{P_e(a_s, b_s)}{P_w^{0s} P_w^{1s}}, \quad (19)$$

for nodes $s \in \mathcal{T}_{\mathcal{D}}$ with depth $< D$.

When the sub-model $\mathcal{S}_s$ contains only a root s, not at depth D, then

$$\begin{aligned} Q_w^s(\mathcal{S}_s) &= \frac{P_e(a_s, b_s)}{P_e(a_s, b_s) + P_w^{0s} P_w^{1s}} \\ &= \frac{\beta_s}{\beta_s + 1}. \end{aligned} \quad (20)$$

Observe that if the sub-model $\mathcal{S}_s$ consists only of a single node s at level D then

$$Q_w^s(\mathcal{S}_s) = 1. \quad (21)$$

Summarizing we can write

$$Q_w^s(\mathcal{S}_s) = \begin{cases} Q_w^{0s}(\mathcal{S}_{0s}) Q_w^{1s}(\mathcal{S}_{1s}) \frac{1}{\beta_s + 1} & \text{if } |\mathcal{S}_s| > 1, \\ \frac{\beta_s}{\beta_s + 1} \text{ if depth}(s) < D & \text{for } |\mathcal{S}_s| = 1, \\ 1 \text{ if depth}(s) = D & \text{for } |\mathcal{S}_s| = 1. \end{cases} \quad (22)$$

This expression suggests that we can write for the a posteriori probability of a given model $\mathcal{S}$ after having observed the source sequence x_1^T

$$P_w(\mathcal{S}|x_1^T) = \frac{2^{-\Gamma_D(\mathcal{S})} \prod_{s \in \mathcal{S}} P_e(a_s, b_s)}{P_w^{\lambda}} = Q_w^{\lambda}(\mathcal{S}). \quad (23)$$

To compute the a posteriori probability corresponding to a model $\mathcal{S}$ we just have to form a product which consists of a factor $1/(\beta_{s'} + 1)$ for each internal node s' of the model $\mathcal{S}$ and a factor $\beta_{s"}/(\beta_{s"} + 1)$ for each leaf $s"$ of the model $\mathcal{S}$ not at level D.

B. Arbitrary weighting

In the previous subsection we have assumed that $(1/2, 1/2)$-weighting was applied in the CTW-procedure, i.e.

$$\alpha_s = 1/2 \text{ for all internal nodes } s \in \mathcal{T}_{\mathcal{D}}. \quad (24)$$

Suppose here that we have an *arbitrary weighting* i.e. for all $s \in \mathcal{T}_{\mathcal{D}}$

$$P_w^s \triangleq \alpha_s P_e(a_s, b_s) + (1 - \alpha_s) P_w^{0s} P_w^{1s}, \quad (25)$$

where $\alpha(s) = 1$ for s at depth D. Now we define for a sub-model rooted in node $s \in \mathcal{T}_\mathcal{D}$ the following conditional probability

$$Q_w^s(\mathcal{S}_s) \triangleq \frac{P^*(\mathcal{S}_s) \prod_{s \in \mathcal{S}_s} P_e(a_s, b_s)}{P_w^s}, \tag{26}$$

where the a priori probability $P^*(\mathcal{S}_s)$ of a sub-model $\mathcal{S}_s$ is defined as

$$P^*(\mathcal{S}_s) \triangleq \prod_{s' \in \mathcal{S}_s} (1 - \alpha_{s'}) \prod_{s" \in \mathcal{S}_s} \alpha_{s"}, \tag{27}$$

where s' runs over the internal nodes and s" over the leaves of $\mathcal{S}_s$. Then we can write

$$Q_w^s(\mathcal{S}_s) = \begin{cases} Q_w^{0s}(\mathcal{S}_{0s}) Q_w^{1s}(\mathcal{S}_{1s}) \frac{1}{\frac{\alpha_s}{1-\alpha_s}\beta_s + 1} & |\mathcal{S}_s| > 1, \\ \frac{\frac{\alpha_s}{1-\alpha_s}\beta_s}{\frac{\alpha_s}{1-\alpha_s}\beta_s + 1} \text{ if depth}(s) < D & |\mathcal{S}_s| = 1, \\ 1 \text{ if depth}(s) = D & |\mathcal{S}_s| = 1. \end{cases} \tag{28}$$

Again we obtain for the a posteriori probability of model $\mathcal{S}$

$$P_w(\mathcal{S}|x_1^T) = Q_w^\lambda(\mathcal{S}). \tag{29}$$

Note that again from inspection of the β's in some of the nodes of the context tree we can form the a posteriori probability of any tree model that fits in $\mathcal{T}_\mathcal{D}$.

The idea of using β's was proposed by Willems and Tjalkens [7] to simplify the implementation of the CTW-method.

IV. Two-pass methods

A. Introduction

The CTW-method is a *one-pass algorithm*. The source sequence x_1^T is processed in a sequential way, i.e. the first source symbol x_1 is observed, code is produced, the second symbol x_2 is observed, code is produced, etcetera. In a *two-pass system* the entire source sequence x_1^T is observed first. Only after that a codeword is constructed. Consider the following *two-pass method*:

1. After observing x_1^T determine the "best model" $\mathcal{S}$ matching to x_1^T.
2. Encode this model $\mathcal{S}$.
3. Encode the sequence x_1^T given this model $\mathcal{S}$.

To specify such an algorithm we have to specify the parts out of which it consists. Some questions that arise now are:

- What is the best model $\mathcal{S}$? How can it be determined efficiently?
- How do we encode the best model $\mathcal{S}$ and x_1^T given $\mathcal{S}$?

V. Maximum a posteriori model selection

A. Standard weighting

The context-tree maximizing method (see e.g. Volf and Willems [8]) chooses for the sequence x_1^T, the model $\mathcal{S}$ that maximizes

$$2^{-\Gamma_D(\mathcal{S})} \prod_{s \in \mathcal{S}} P_e(a_s, b_s). \tag{30}$$

This method can be used to minimize the total codeword length if we use a two-pass universal source code. The best model is determined recursively, using a context tree, by setting

$$P_m^s = P_e(a_s, b_s) \tag{31}$$

if s is at level D in the context tree $\mathcal{T}_\mathcal{D}$, and

$$P_m^s = \max[\frac{1}{2} P_e(a_s, b_s), \frac{1}{2} P_m^{0s} P_m^{1s}] \tag{32}$$

if s has depth $< D$. It is assumed that the entire sequence x_1^T was processed in the context tree. We finally will find the best model $\mathcal{S}$ by tracking the maximization procedure, starting in the *root* λ of the context tree. If in a node s in the context tree $P_e(a_s, b_s) \geq P_m^{0s} P_m^{1s}$ then s is a leaf of the best tree $\mathcal{S}$ and we do not have to investigate the sub-tree rooted in s any further. Otherwise s is an internal node of the best model and we have to check the nodes $0s$ and $1s$.

B. Arbitrary weighting

Suppose that we want to maximize

$$P^*(\mathcal{S}) \prod_{s \in \mathcal{S}} P_e(a_s, b_s), \tag{33}$$

for a priori probability distribution $P^*(\cdot)$ over the models as defined in (27) resulting from the assignments $(\alpha_s, 1 - \alpha_s)$ for all internal nodes s of $\mathcal{T}_\mathcal{D}$. Then, as before, we have to set

$$P_m^s = P_e(a_s, b_s) \tag{34}$$

if s is at level D in the context tree $\mathcal{T}_\mathcal{D}$, but now

$$P_m(s) = \max[\alpha_s P_e(a_s, b_s), (1 - \alpha_s) P_m^{0s} P_m^{1s}] \tag{35}$$

if s has depth $< D$.

Tracking the described procedure leads to the maximum a posteriori (MAP) model. If in a node s in the context tree $\alpha_s P_e(a_s, b_s) \geq (1 - \alpha_s) P_m^{0s} P_m^{1s}$ then s is a leaf of the best tree $\mathcal{S}$ and we do not have to investigate the sub-tree rooted in s any further, etcetera.

If we assume an assignment $(\alpha_s, 1 - \alpha_s)$ for all s in $\mathcal{T}_\mathcal{D}$ that results in uniform a priori probabilities over all models then the proposed method leads to the maximum likelihood (ML) model after having observed the source sequence x_1^T.

VI. Encoding the best model and the sequence given this model

A. Encoding the model $\mathcal{S}$, standard weighting

Suppose that the best model $\mathcal{S} = \{00, 10, 1\}$. A recursive coding method for model $\mathcal{S}$ yields:

$$\begin{aligned} \text{code}(\lambda) &= 1, \text{code}(0), \text{code}(1) \\ &= 1, 1, \text{code}(00), \text{code}(10), 0 \\ &= 1, 1, 0, 0, 0. \end{aligned}$$

The rule here is that the code of a sub-tree rooted in s is 0 if this sub-tree only contains one node, and 1 followed by the codes of the sub-trees rooted in $0s$ and $1s$ otherwise. In general we need $2|\mathcal{S}| - 1$ binary digits to specify a model $\mathcal{S}$, hence the model redundancy is equal to $2|\mathcal{S}|-1$ bits. Note that nodes at level D need no code.

B. Encoding the model $\mathcal{S}$, arbitrary weighting

Arithmetic coding can be used for encoding the best model if the a priori probability distribution $P^*(\cdot)$ over the models is as defined in (27) (resulting from the assignment $(\alpha_s, 1 - \alpha_s)$ for all nodes s in $\mathcal{T}_\mathcal{D}$).

Then we obtain for the codeword-length

$$L(\mathcal{S}) < \log_2 \frac{1}{P^*(\mathcal{S})} + 2. \tag{36}$$

In practice this can be accomplished by first encoding the model $\mathcal{S}$ into a binary sequence as was described in the previous subsection and then encoding this binary sequence with an arithmetic coder subdividing the intervals according to the appropriate α_s.

C. Encoding sequence x_1^T given model $\mathcal{S}$

As we have seen before in subsection I-D we can use arithmetic coding to encode x_1^T given the model $\mathcal{S}$. Good coding probabilities for the sub-sequences in the leaves are obtained from the Krichevsky-Trofimov [4] estimator $P_e(\cdot)$. This results in redundancy

$$\rho(x_1^T) < \left(\frac{|\mathcal{S}|}{2} \log_2 \frac{T}{|\mathcal{S}|} + |\mathcal{S}| \right) + 2, \tag{37}$$

for $T \geq |\mathcal{S}|$.

VII. Bounded-complexity maximizing

The maximizing algorithms described in section V can be modified such that they produce a model $\mathcal{S}$ with C leaves (parameters) or less. This is done to limit the complexity of the decoder. To determine such a bounded-complexity model we walk through the context tree in a depth-first search. In every node s we determine a list which contains for all $c = 1, C$ the maximized probability achievable with the best sub-tree rooted in that node having c leaves. Entry c in this list also contains how the c leaves should be distributed over the sub-trees rooted in the children $0s$ and $1s$ of node s to obtain this maximized probability. In each node s such a list can be computed by comparing the estimated probability in that node $P_e(a_s, b_s)$ times α_s to all the products $P_w^{0s} P_w^{1s}$ of the maximized probabilities in the lists from its two children $0s$ and $1s$ times $1 - \alpha_s$ and taking the best C alternatives. Finally one finds a list in the root λ of the context tree with for every number of leaves up to C, the corresponding maximized probability.

To determine the list in the root one needs at most $D+1$ open (active) lists. Once one knows the appropriate total number of leaves $c \leq C$ in the root, one knows which distribution of the number of leaves over each child of the root resulted in this "optimal" solution. In this way the problem is reduced to the same problem for two trees of depth $D - 1$. If one applies this technique recursively, we will find the best constrained model. For obvious reasons we have called this method the yoyo-method (see e.g. Volf and Willems [8]).

A slight improvement over the strategy that we have just described is obtained if we realize that a list in a node at level d need not contain more than $C - d$ items. For the practical case in which C is considerably larger than d this is not a significant improvement however. Note that this improvement implies that empty nodes (nodes that contain an empty sequence) are counted.

VIII. Finding the N best models

Next suppose that we are interested in retrieving the N best models after a sequence x_1^T is processed. How should we determine these models in an efficient way?

Assume that for both children $0s$ and $1s$ of a node s there is a list containing the N best sub-models rooted at $0s$ and $1s$ respectively. Then we form a list in node s by combining all sub-models from the list in $0s$ with those of the list in $1s$ and comparing these combinations with the memoryless sub-model of node s and putting the N best alternatives in the list for node s. In this way we can work towards the root λ of the context tree. There we then have a list containing the N best tree models.

A question still remains how the n-th best model (for $n = 1, N$) is stored in a certain node s. Instead of storing the entire model $\mathcal{S}_s$ (using e.g. a code as was described in subsection VI-A) we can store (n_0, n_1) which means that this model is a combination of the n_0-th best sub-model in node $0s$ and the n_1-th best model in node $1s$. Note that the sub-model $\mathcal{S}_s$ can also contain only one node.

Again to determine the list in the root λ of the context tree one needs at most $D + 1$ open lists.

Once the N best tree models are known in the root, one knows for each of the N models which combination of the best sub-models of the children of the root resulted in this "optimal" solution. Therefore as in the previous section the problem is reduced to the same problem for two trees of depth $D-1$. Applying this technique recursively, we will find the best N models. Again this is a yoyo approach.

IX. Conclusion

In this paper we have investigated some questions related to the context-tree weighting method [5]. Most of these questions were about maximizing and a posteriori probabilities of tree models. It would be interesting to extend the results that we have obtained here to more general finite context sources than tree sources. Weighting algorithms for general finite context sources were already described in Willems, Shtarkov, and Tjalkens [9].

References

[1] F. Jelinek, *Probabilistic Information Theory,* New York: McGraw-Hill, 1968, pp. 476-489.

[2] J. Rissanen, "Generalized Kraft Inequality and Arithmetic Coding," *IBM J. Res. Devel.*, vol. 20, p. 198, 1976.

[3] R. Pasco, *Source Coding Algorithms for Fast Data Compression,* Ph.D. thesis, Stanford University, 1976.

[4] R.E. Krichevsky and V.K. Trofimov, "The Performance of Universal Encoding," *IEEE Trans. Inform. Theory,* vol. IT-27, pp. 199-207, March 1981.

[5] F.M.J. Willems, Y.M. Shtarkov and Tj.J. Tjalkens, "The Context-Tree Weighting Method: Basic Properties," *IEEE Trans. Inform. Theory,* vol. IT-41, pp. 653-664, May 1995.

[6] J. Rissanen, "Universal Coding, Information, Prediction, and Estimation," *IEEE Inform. Theory,* vol. IT-30, pp. 629-636, July 1984.

[7] F.M.J. Willems and Tj.J. Tjalkens, "Reducing complexity of the context-tree weighting method," *Proc. IEEE International Symposium on Information Theory,* Cambridge, Mass., August 16-21, 1998, p. 347.

[8] P.A.J. Volf and F.M.J. Willems, "A Study of the Context Tree Maximizing Method," *Proceedings 16th Symposium of Information Theory in the Benelux,* Niewerkerk a/d Ijssel, May 18 & 19, 1995, pp. 3 - 9.

[9] F.M.J. Willems, Y.M. Shtarkov and Tj.J. Tjalkens, "Context Weighting for General Finite Context Sources," *IEEE Trans. on Inform. Theory,* vol. IT-42, pp. 1514 - 1520, September 1996.