

Computation of universal objects for distributions over co-trees

Henrik Densing Petersen University of Copenhagen
 Department of Mathematical Sciences
 Universitetsparken 5,
 2100 Copenhagen, Denmark
 Email: hdp@math.ku.dk

Flemming Topsøe University of Copenhagen
 Department of Mathematical Sciences
 Universitetsparken 5,
 2100 Copenhagen, Denmark
 Email: topsoe@math.ku.dk

Abstract—For an ordered set consider the model of distributions P for which an element which precedes another element is considered the more significant one in the sense that the implication $a \leq b \Rightarrow P(a) \geq P(b)$ holds. It will be shown that if the ordered set is a finite co-tree, then the universal predictor for the model or, equivalently, the corresponding universal code, can be determined exactly via an algorithm of low complexity. Natural relations to problems on the computation of capacity and on the determination of information projections are established. More surprisingly, a direct connection to a problem of isotone regression also appears possible.

Index terms – Universal code, universal predictor, isotone regression.

CONTENTS

I	The Problem	1
II	Motivation	3
III	Co-trees with full spectrum	4
IV	Preview of the algorithm	4
V	Relativization	6
VI	Ideas on the way to an algorithm	6
VII	Construction from the top	7
VIII	The central subroutine	8
IX	Co-trees with uniform branching	11
X	Conclusions and final comments	13
XI	Appendix with proofs	13
	References	16

I. THE PROBLEM

We shall study finite co-trees, i.e. finite ordered sets Λ for which every non-maximal element a has a unique immediate successor, denoted a^+ . For $a \in \Lambda$, a^\downarrow denotes the *left section* of all $b \in \Lambda$ with $b \leq a$. The elements of a co-tree may be depicted as nodes of an oriented graph. If there is only one maximal node, a , the co-tree is *suspended* with a as *top-node*. Then $a^\downarrow = \Lambda$.

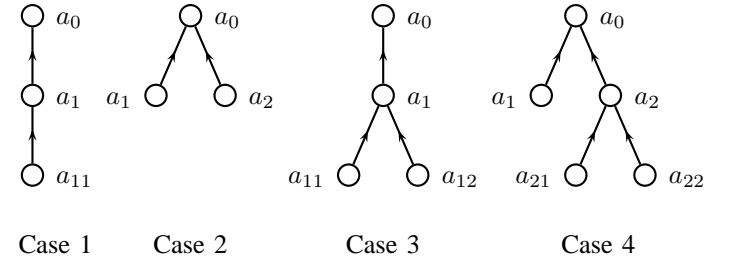


Figure 1. Some simple suspended co-trees

Simple examples are depicted in Figure 1. Case 1 is a linear order. Note that nodes are given in a *standard representation* with systematic indexing according to the *levels* of the nodes which are counted from the top with the top node a_0 in level 0. The largest level of a node is the *height* of the co-tree.

For a sequence (k_1, \dots, k_n) of natural numbers, $\Lambda[k_1, \dots, k_n]$ denotes the suspended co-tree with k_ν immediate predecessors of each node in level $\nu - 1$ for $1 \leq \nu \leq n$. These co-trees are co-trees with *uniform branching*. The sequence (k_1, \dots, k_n) is the *branching pattern*. The first three co-trees in Figure 1 are of this type, respectively, $\Lambda[1, 1]$, $\Lambda[2]$ and $\Lambda[1, 2]$.

By $M_+^1(\Lambda)$ we denote the set of all distributions (always understood to be probability distributions) over Λ . The *order model* $\mathcal{P} = \mathcal{P}(\Lambda)$, which is the object we will study, is the model of all distributions P for which $P(a) \geq P(b)$ whenever

$a \leq b$ ¹. By U_a we denote the uniform distribution over a^\perp . These distributions as well as their mixtures are all members of $\mathcal{P}(\Lambda)$. Conversely, any $P \in \mathcal{P}(\Lambda)$ can be written in a unique way as a convex mixture

$$P = \sum_{a \in \Lambda} w_a U_a. \quad (1)$$

In other words, as is easily proved²:

Proposition 1.1: The order model $\mathcal{P}(\Lambda)$ is a simplex with the distributions $(U_a)_{a \in \Lambda}$ as extremal elements.

The decomposition (1) is the *barycentric decomposition* of P and the set

$$\sigma(P) = \{a | w_a > 0\} \quad (2)$$

is the *spectrum* of P .

The set $K(\Lambda)$ of *codes over Λ* is here identified with the set of *code-length functions* $\kappa : \Lambda \rightarrow [0, \infty]$, which are required to satisfy *Kraft's equality*

$$\sum_{a \in \Lambda} e^{-\kappa(a)} = 1.$$

For $P \in M_+^1(\Lambda)$ and $\kappa \in K(\Lambda)$ we denote by $\langle \kappa, P \rangle$ the *average code-length*,

$$\langle \kappa, P \rangle = \sum_{a \in \Lambda} \kappa(a) P(a).$$

The overall goal is to choose a code so as to minimize this quantity. If P is fixed, the minimum is attained for the code *adapted to P* , given by

$$\kappa(a) = \ln \frac{1}{P(a)}; a \in \Lambda,$$

and the minimum value is the *entropy* of P ,

$$H(P) = \sum_{a \in \Lambda} P(a) \ln \frac{1}{P(a)}.$$

When κ is adapted to P , we also express this by saying that P is the distribution which *matches* κ .

The *redundancy* associated with P and κ , or, in more suggestive terms, the *redundancy of κ with P as the "true" distribution* is denoted $D(P||\kappa)$ and defined as the difference between actual and minimal possible average code-length, i.e.

$$D(P||\kappa) = \langle \kappa, P \rangle - H(P). \quad (3)$$

This quantity is nothing but the well known *Kullback-Leibler divergence* between P and the distribution Q which matches κ , in standard notation,

$$D(P||Q) = \sum_{a \in \Lambda} P(a) \ln \frac{P(a)}{Q(a)}.$$

¹This model – and not the alternative choice of all order-preserving distributions – is considered to be the natural one, a main reason being that if a precedes b ($a < b$) this is taken as a sign that a is more "significant" than b , hence, for sensible distributions, one should have $P(a) \geq P(b)$ rather than the other way round. In terms of coding (see below) our choice appears even more natural as it reflects the good sense of associating the shorter code words to the more significant events.

²please note that in order to facilitate a focus on essentials, we shall relegate all proofs of results which are not either well known or considered "easily proved" to an appendix.

We define *minimax redundancy* associated with the order model as the quantity

$$R_{\min} = \inf_{\kappa \in K(\Lambda)} R(\kappa)$$

with $R(\kappa)$ given by

$$R(\kappa) = \sup_{P \in \mathcal{P}} D(P||\kappa); \kappa \in K(\Lambda).$$

It may be seen directly, and also follows from our results, that there exists a unique code κ^* , *the universal code*, such that $R(\kappa^*) = R_{\min}$. The distribution which matches the universal code is the *universal predictor*. It is considered the most unbiased representation of the model \mathcal{P} . The two universal objects identified, are those we shall aim at characterizing by an algorithm of low complexity. In order to achieve this, we appeal to a special instance of a result from optimization theory, which is much used in information theory and there often identified by a reference to Kuhn and Tucker. We formulate the result in a way adapted to our needs:

Proposition 1.2 (Kuhn-Tucker criterion): Consider the order model $\mathcal{P} = \mathcal{P}(\Lambda)$ associated with a finite co-tree Λ . Let $P^* \in \mathcal{P}$ and let κ^* be the code adapted to P^* . Then P^* is the universal predictor (equivalently, κ^* is the universal code) if and only if, for some constant R , the following two conditions hold:

$$D(U_a||\kappa^*) = R \text{ for } a \in \sigma(P^*), \quad (4)$$

$$D(U_a||\kappa^*) \leq R \text{ for all } a \in \Lambda. \quad (5)$$

When this is so, R is the minimax redundancy: $R_{\min} = R$.

The role of the result for our approach, is further commented on in the next section.

As one consequence, one may reduce the search for the universal objects associated with a general finite co-tree to the search for suspended co-trees³.

Objects P^* and κ^* will from now on denote the universal predictor and the universal code of a co-tree Λ under discussion. The *spectrum* of Λ is the spectrum of P^* : $\sigma(\Lambda) = \sigma(P^*)$. Nodes in $\sigma(\Lambda)$ are referred to as *active nodes* of Λ . The co-tree Λ has *full spectrum* if all nodes are active.

Consider the four examples of Figure 1. By Proposition 1.2 it is easy to check that P^* and R_{\min} are as indicated in Table I. The universal code is the code adapted to P^* . Except for Case 3, where a_1 is inactive, the examples have full spectrum. Apparently, even simple co-trees can have inactive nodes. Offhand, there is little we can say:

Proposition 1.3: Every maximal node of a co-tree is active.

We leave the simple proof to the reader. and add that less trivially, cf. Proposition 5.2, also the minimal nodes of a

³Indeed, if Λ is the direct sum of (suspended) co-trees Λ_ν ; $\nu = 1, \dots, m$ with associated minimax redundancies R_ν and universal predictors P_ν^* then

$$P^* = \sum_{\nu=1}^m \frac{e^{R_\nu}}{e^{R_1} + \dots + e^{R_m}} P_\nu^*$$

is the universal predictor for Λ and $\ln \sum_{\nu} e^{R_\nu}$ the associated minimax redundancy.

	Case 1	Case 2	Case 3	Case 4
$\sigma(\Lambda)$	Λ	Λ	$\Lambda \setminus \{a_1\}$	Λ
$P^*(a_0)$	$\frac{4}{27} \cdot \frac{1}{Z}$	$\frac{1}{27} \cdot \frac{1}{Z}$	$\frac{1}{16} \cdot \frac{1}{Z}$	$\frac{27}{3125} \cdot \frac{1}{Z}$
$P^*(a_1)$	$\frac{1}{4} \cdot \frac{1}{Z}$	$2 \times \frac{1}{Z}$	$\frac{1}{16} \cdot \frac{1}{Z}$	$\frac{1}{Z}, \frac{1}{27} \cdot \frac{1}{Z}$
$P^*(a_2)$	$\frac{1}{Z}$		$2 \times \frac{1}{Z}$	$2 \times \frac{1}{Z}$
$R_{\min} = \ln Z$	$\ln \frac{151}{108}$	$\ln \frac{55}{27}$	$\ln \frac{17}{8}$	$\ln \frac{256979}{84375}$

Table 1
UNIVERSAL PREDICTORS FOR THE CO-TREES IN FIGURE 1

co-tree are active. In general, the structure of $\sigma(\Lambda)$ is quite intricate and one may view the algorithmic determination of this set for any co-tree as the main problem to be solved. Once $\sigma(\Lambda)$ is determined, κ^* (hence also P^*) and R_{\min} can easily be determined, cf. (15) and Corollary 5.1.

II. MOTIVATION

The natural interpretations related to codes as well as the significance of the problem outlined as one of *general universal prediction and coding* (general, because many other models than models related to order structure may be considered) is recognized in the information theoretical literature since long. Early works in this area include Fitingof [1] and Davisson [2]. The reader may also consult the survey article [3] by Feder and Merhav.

The original motivation behind our very special study related only to order models in co-trees consists basically of three parts. Firstly, to the best of our knowledge, this class is the most comprehensive class for which an exact determination of universal objects can be provided either directly or via a reasonable algorithm. For the subclass of order models based on linearly ordered sets, a complete result already exists. It is due to Ryabko who developed a closed formula for the universal predictor, cf. [4]. For the larger subclass of co-trees with uniform branching, an algorithm was announced in Topsøe [5] but the details were never published.

Secondly, our main results, Theorems 7.1, 8.1 and 9.1, may also be considered as useful reservoirs of examples which may serve as test cases for future research. However, we remark that it appears very difficult, even theoretically impossible, to develop exact results expressed in terms of standard functions for other desirable models than those here considered, either based on order structures other than co-trees (e.g. trees) or on other constructs (such as Bernoulli models). Thus, the idea to look into models based on sequences rather than individual observations from an order model \mathcal{P} , is bound to fail. Severe obstacles to such a program exists as will be revealed by a reference to Galois theory (details will be provided in research by Harremoës and Topsøe, in preparation).

As a final basic motivation we note, as pointed out to us by Boris Ryabko, cf. also [6], that for certain applications to biology, information about biological species is sometimes

available only in inconclusive form resulting – not in the direct determination of their relative numbers – but only in an ordering among the species, from the more frequent to the less frequent ones. Modelling as done here based on a co-tree is one possibility, though modelling based on trees rather than co-trees appear just as interesting, or perhaps even more so. However, models with trees in place of co-trees are without reach if you insist on expressing the universal objects in closed form.

When you look back and consider the methods applied, further aspects appear which contribute to motivate the present research. The reliance on Proposition 1.2 points to the interesting connection between minimax redundancy and maximal transmission rate, *capacity*, i.e. the *redundancy-capacity theorem* of Gallager and Ryabko, see [7]. In our situation, the result involves the discrete memoryless channel with Λ as input- as well as output alphabet and with the distributions $(U_a)_{a \in \Lambda}$ as the conditional output distributions, given an input letter (here a node in Λ). By the redundancy-capacity theorem, the optimal distribution on the input side is given by the barycentric coordinates of the universal predictor P^* and the optimal distribution on the output side is P^* itself. We shall not exploit this connection in the sequel. Rather, the situation is that the results which we shall develop can be used to show how to determine the optimal distributions and the capacity of the special discrete memoryless channels that can arise in the way described.

Another observation concerns a connection to the well known problem of determining *information projections*. In fact, to any co-tree we can associate a probability distribution W^* through simple calculations such that the sought P^* is the information projection of W^* on $\mathcal{P}(\Lambda)$ (for W^* take the measure obtained by normalization of W from Theorem 3.1). This fact – or the connection to the problem related to capacity pointed to above – may be exploited to calculate P^* via standard numeric algorithms. However, we stress that this will only lead to approximate determinations of P^* . As we are here concerned with precise determinations, either via a direct formula (possible only in special cases) or via an algorithm which stops with the exact result after finitely many steps, we shall not pursue this possibility. Another matter is that standard algorithms may be useful anyhow in order to guess what the spectrum is, and then exact formulas are easy to derive, cf. the discussion related to (15) in Section VI.

Finally we comment on a connection to a problem of *isotone regression* within least squares analysis. This depends on an observation of an anonymous referee who, based on the information projection formulation given above, encouraged the authors to investigate connections to an algorithm for isotone regression worked out by Pardalos and Xue, cf. [8]. Looking into this has led to a conjecture involving the function k defined by $k(a) = \ln \frac{1}{W(a)}$; $a \in \Lambda$, viz. that the associated isotone regression coincides with the relativized universal code $\tilde{\kappa}^*$ defined in Section V. In other terms, the conjecture states that $f = \tilde{\kappa}^*$ with f the unique isotone function on Λ (i.e. $a \leq b \Rightarrow f(a) \leq f(b)$) which minimizes $\sum_{a \in \Lambda} |f(a) - k(a)|^2$.

The two algorithms, ours and that of Pardalos and Xue, are different but similar in structure – e.g. both work with a notion of “blocking” – and may essentially compute the same objects. Regarding complexity, Pardalos and Xue demonstrate that their algorithm is in general very efficient whereas we limit a detailed study of such issues to the case of co-trees with uniform branching, see Section IX.

III. CO-TREES WITH FULL SPECTRUM

In this section we investigate when a given co-tree Λ has full spectrum. The criterion we shall find will be expressed in terms of the numbers

$$N(a) = |a^\downarrow|; a \in \Lambda$$

where $|\dots|$ denotes the number of elements in \dots . We also need the numbers

$$\overline{N}(a) = N(a) \ln N(a); a \in \Lambda.$$

Further, for $a \in \Lambda$, we introduce the notation a^- for the set of *immediate predecessors* of a , i.e. the set of all $b < a$ for which no node c satisfies $b < c < a$. If e.g. a is a minimal node, $a^\downarrow = \{a\}$, $N(a) = 1$, $\overline{N}(a) = 0$, and $a^- = \emptyset$.

We associate the following *weights* with the nodes of Λ :

$$W(a) = \frac{\prod_{b \in a^-} N(b)^{N(b)}}{N(a)^{N(a)}}; a \in \Lambda \quad (6)$$

and also introduce the resulting *normalizing factor*

$$Z = \sum_{a \in \Lambda} W(a). \quad (7)$$

Theorem 3.1: A co-tree Λ has full spectrum if and only if, for every pair of nodes (b, a) with $b \in a^-$, the inequality $W(b) \geq W(a)$ holds. And when this condition is satisfied, the universal predictor is given by normalization of W , i.e. $P^*(a) = W(a)/Z$ for any $a \in \Lambda$. Furthermore, $R_{\min} = \ln Z$ and the universal code is given by

$$\kappa^*(a) = R_{\min} + \left(\overline{N}(a) - \sum_{b \in a^-} \overline{N}(b) \right).$$

[*Proof in appendix*]

We remark that when the condition stated holds, the strict inequality $W(b) > W(a)$ will actually hold for all nodes with $b \in a^-$.

Note that the cases 1, 2, and 4 from Section I can be handled based on this theorem.

If we conceive the result as an algorithm to check whether the co-tree in question has full spectrum or not, we note that the algorithm is very efficient as the number of inequalities which need to be checked is at most the number of nodes in the co-tree.

Let us have a closer look at Theorem 3.1 in the case of a co-tree $\Lambda[k_1, \dots, k_n]$ with uniform branching. For such a co-tree, we denote by N_ν the common number of $N(a)$ for

nodes in level ν ($\nu = 0, \dots, n$). The N_ν 's may be calculated recursively as follows:

$$N_n = 1, \quad N_\nu = 1 + k_{\nu+1}N_{\nu+1} \text{ for } \nu = n-1, \dots, 0. \quad (8)$$

From Theorem 3.1 we derive the following corollary:

Corollary 3.1: The co-tree $\Lambda = \Lambda[k_1, \dots, k_n]$ has full spectrum if and only if, for every $\nu = 0, 1, \dots, n-2$,

$$\left(\frac{1}{N_\nu} \right)^{\frac{N_\nu}{\rho_\nu}} \left(\frac{1}{N_{\nu+2}} \right)^{1 - \frac{N_\nu}{\rho_\nu}} \leq \frac{1}{N_{\nu+1}}, \quad (9)$$

where the numbers $\rho_0, \dots, \rho_{n-1}$ are given by

$$\rho_\nu = (1 + k_{\nu+1})N_{\nu+1} = N_\nu + N_{\nu+1} - 1.$$

[*Proof in appendix*]

Specializing further we obtain the following corollary which extends Ryabko's theorem [4] in a natural way (Ryabko's theorem corresponds to the case $k_1 = \dots = k_n = 1$ which gives $N_\nu = n - \nu + 1$; $\nu = 0, 1, \dots, n$):

Corollary 3.2: Every co-tree $\Lambda = \Lambda[k_1, \dots, k_n]$ with $k_1 \geq k_2 \geq \dots \geq k_n$ has full spectrum and the universal predictor P^* is given by

$$\begin{aligned} P^*(a) &= N_\nu^{-N_\nu} (N_{\nu+1})^{k_{\nu+1}N_{\nu+1}} / Z \\ &= N_\nu^{-N_\nu} (N_{\nu+1})^{N_\nu - 1} / Z \end{aligned}$$

for all points a in level ν ($\nu = 0, 1, \dots, n$) with Z a normalization constant.

IV. PREVIEW OF THE ALGORITHM

For the preview in this section as well as for later usage we introduce some special notation and concepts for an arbitrary co-tree Λ . A set $B \subseteq \Lambda$ is *hereditary* if $b \leq a$, $a \in B$ implies $b \in B$. A *blocking set* for a node $a \in \Lambda$ is a hereditary subset B of $a^\downarrow \setminus \{a\}$ which contains every minimal node of $a^\downarrow \setminus \{a\}$. The largest such set is $a^\downarrow \setminus \{a\}$. The *exterior of B in a*, denoted $S_B(a)$, and the *ceiling of B in a*, denoted $T_B(a)$, are the sets

$$S_B(a) = a^\downarrow \setminus B, \quad (10)$$

$$T_B(a) = \text{set of maximal nodes of } B. \quad (11)$$

The nodes in $T_B(a)$ are the first nodes in B you meet on paths from a to a minimal node. The exterior $S_B(a)$ is always non-empty. The same is true for the sets B and $T_B(a)$ unless a is a minimal node of Λ , in which case only the empty set is blocking for a .

For any function ϕ on Λ , ϕ^σ denotes the *accumulated function* defined as the set-function

$$\phi^\sigma(\Delta) = \sum_{b \in \Delta} \phi(b)$$

with Δ any subset of Λ . Using this notation, we define the *bracket of a in B*, with B a blocking set for a , as the number

$$[a, B] = \frac{\overline{N}(a) - \overline{N}^\sigma(T_B(a))}{|S_B(a)|} = \frac{\overline{N}(a) - \overline{N}^\sigma(T_B(a))}{N(a) - N^\sigma(T_B(a))}. \quad (12)$$

By $[a]_{\max}$ we denote the maximal value of $[a, B]$ with B a blocking set for a . It turns out that among the blocking sets B

⁴Added in proof: The conjecture has now been settled. Details will be published elsewhere.

for a with maximal bracket, there exists a set-theoretically largest one (Proposition 7.3). This uniquely defined set is denoted $B^*(a)$ and the corresponding ceiling and exterior are denoted respectively $T^*(a)$ and $S^*(a)$.

Our algorithm is in two parts. Part I determines all the B^* 's and lists the corresponding maximal brackets. The result can be shown graphically for co-trees of moderate size, cf. Figures 2, 3 and 4. There we have also indicated in black the top-nodes as well as all nodes in ceilings (T^* 's) for the B^* 's constructed during Part I. We remark that all nodes of the co-tree will be black after Part I if and only if the co-tree has full spectrum (and if and only if all tests introduced below are positive).

The search carried out during Part I starts from the bottom with the minimal nodes. The resulting brackets are all 0 and ceilings are empty. Then move up the co-tree you are investigating. This is done incrementally so that when you work with a specific node, say a , you have already inspected every node in $a^\perp \setminus \{a\}$. The first thing to test is if $B = a^\perp \setminus \{a\}$ could be the sought blocking set $B^*(a)$. This is done by testing if the bracket $[a, B]$ dominates all maximal brackets calculated for nodes in $T_B(a)$ ($= a^-$ for this first attempt). If this is not the case, you indicate the failure on the graph by adding a “dagger” after the value $[a, B]$. And then you replace B by a somewhat smaller set and repeat the procedure until the test carried out is positive. For examples of moderate size as in the figures shown, you can guess which replacements to make. For the formal algorithm – to be developed rigorously in the sequel – this is done in a systematic way as indicated later in the flow diagram in Figure 6, page 9.

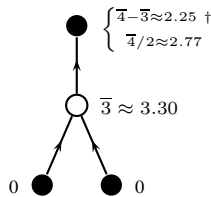


Figure 2. The algorithm for the Case-3 co-tree

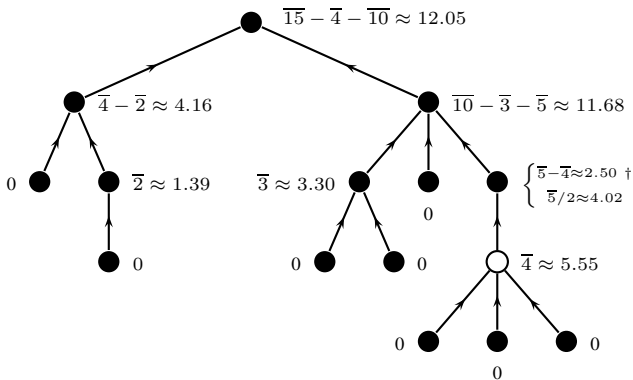


Figure 3. The algorithm for a “general” co-tree Λ .

For the co-trees of figures 2, 3 and 4 you are done after completion of Part I. In these cases the spectrum coincides with the set of black nodes. The brackets calculated for the active nodes are the universal code-lengths measured relative to the

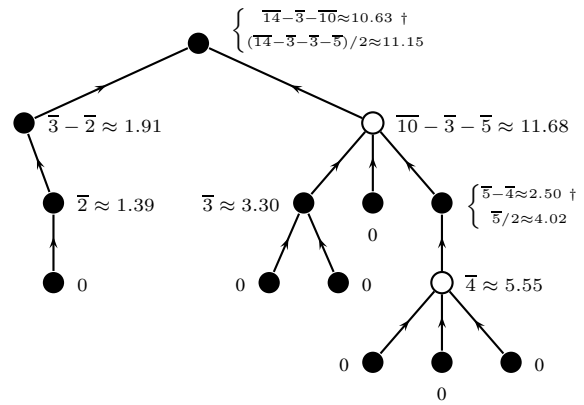


Figure 4. The algorithm for the previous co-tree with deletion of one minimal node.

shortest code-length. The calculation of minimum redundancy and the universal code is straight forward as explained in the section to follow.

To get a feel for the numerical figures, we mention that for the co-tree in Figure 3, Corollary 5.1 implies that

$$R_{\min} = \ln \left(8 + 2^{-2} + 3^{-3} + 2 \cdot 5^{-5/2} + 2^{-6} + 2^{-10} 3^3 5^{-5} + 2^{18} 3^{-15} 5^{-5} \right),$$

which is approximately 2.12 measured in natural units, corresponding to 3.06 bits. This may be compared with the 3 bits necessary to encode the 8 minimal nodes which are equally probable under the universal predictor.

The sensitivity of the spectrum due to small changes of the co-tree is illustrated by removing one minimal node from the co-tree in Figure 3. This leads to the co-tree in Figure 4 for which an extra inactive node emerges in a part of the co-tree which has not been affected by the removal. Thus one cannot decide “locally” if a node is active or not. For the co-tree in Figure 4 one finds $R_{\min} \approx 2.01$ natural units ≈ 2.90 bits – compared to the approximately 2.81 bits needed to encode the 7 minimal nodes which have equal probabilities under the universal predictor.

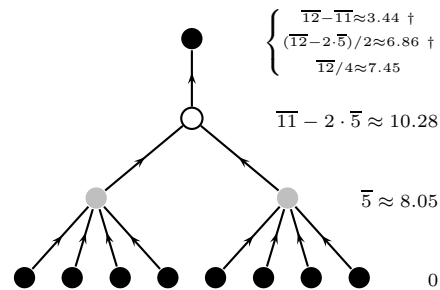


Figure 5. A co-tree with “overshadowing”.

Our last concrete example is shown in Figure 5. For this co-tree, Part I leads to the nodes in levels 0, 2 and 3 as the possible active nodes (the “black nodes”). However, the two nodes in level 2 are not active. This is seen by applying Part II of the algorithm which runs from the top and down, at each step turning a black node into an inactive node if

the corresponding exterior set, say $S^*(a)$, “overshadows” the black node in question. In the case shown, the set consisting of the 8 minimal nodes is the blocking set with maximal bracket for the top-node, hence the two nodes in level 2 are “shadowed away” and become inactive. For another example illustrating how Part I and Part II of the algorithm play together, the reader may turn to the discussion in Section IX of the co-tree $\Lambda[1, 1, 1, 4, 5, 1, 2, 3]$, cf. Table 2. The reader may also check that no black nodes are “shadowed away” during Part II of the algorithm for the co-trees in Figures 2, 3 and 4.

More details about the algorithm as well as a proof of its correctness will be given in the following sections.

V. RELATIVIZATION

With this section we embark on the technical development leading to an algorithm.

Experience tells us that for typical optimization problems of the nature we are studying, “normalization” (via a “partition function” or constant) is natural. This is for instance reflected by the appearance of Z in Table 1. A natural idea then is to facilitate the search for universal objects by a prior normalization. We find it advantageous to work with codes rather than with distributions. Then, rather than normalizing via a division, we should normalize by a suitable subtraction. This leads to objects measured relative to optimal performance and we speak about a process of *relativization*.

Relativization may be defined quite generally. However, we shall only have co-trees in mind for the present study. Therefore, let Λ denote a fixed co-tree and denote as usual by R_{\min} the minimax redundancy for the order model $\mathcal{P} = \mathcal{P}(\Lambda)$. Motivated by the considerations above, we introduce the *relativized universal code* as the function

$$\tilde{\kappa}^* = \kappa^* - R_{\min} .$$

We first characterize this function among all *monotone* functions $\phi : \Lambda \rightarrow \mathbb{R}$. Here, monotonicity means that $\phi(b) \leq \phi(a)$ whenever $b \leq a$. A node $a \in \Lambda$ is ϕ -*active* if either a is a maximal node or else $\phi(a) < \phi(a^+)$ (recall that a^+ denotes the immediate successor of a). If a is not ϕ -active, a is ϕ -*inactive*. If $\phi = \tilde{\kappa}^*$ (or if $\phi = \kappa^*$), we regain the notion of active and inactive nodes introduced in Section I.

Proposition 5.1: A real-valued function ϕ defined on Λ coincides with the relativized universal code $\tilde{\kappa}^*$ if and only if it is monotone and satisfies the two requirements:

$$\phi^\sigma(a^\downarrow) = \overline{N}(a) \text{ for every } \phi\text{-active node } a, \quad (13)$$

$$\phi^\sigma(a^\downarrow) \leq \overline{N}(a) \text{ for every node } a \in \Lambda. \quad (14)$$

[*Proof in appendix*]

Inspection of the proof shows how to obtain the universal code κ^* from the relativized universal code $\tilde{\kappa}^*$ by a simple process of de-relativization:

Corollary 5.1: The minimum redundancy can be obtained from the relativized universal code by the formula $R_{\min} = \ln \sum_{a \in \Lambda} e^{-\tilde{\kappa}^*(a)}$ and the universal code is given by $\kappa^* = \tilde{\kappa}^* + R_{\min}$.

For each node $a \in \Lambda$, the left-section a^\downarrow defines a co-tree in its own right. As another corollary to Proposition 5.1 the following result is easily proved:

Corollary 5.2: If $a \in \sigma(\Lambda)$, then the relativized universal code for the co-tree a^\downarrow is obtained by restricting the relativized universal code for Λ to a^\downarrow . In particular, $\sigma(a^\downarrow) = \sigma(\Lambda) \cap a^\downarrow$.

We stress the importance of the assumption that a be active. Without that assumption new active nodes in a^\downarrow may appear, e.g. a itself, cf. Proposition 1.3 and Case 3 from Figure 1 (and Figure 2).

For the further study, consider, for any $a \in \Lambda$, the *control* of a , denoted \bar{a} , defined as the closest active node greater than or equal to a (i.e. $\bar{a} \in \sigma(\Lambda)$, $\bar{a} \geq a$ and no $c \in \sigma(\Lambda)$ satisfies $a \leq c < \bar{a}$). By Proposition 1.3, \bar{a} is well defined for all $a \in \Lambda$. Clearly, $\bar{a} = a$ if and only if $a \in \sigma(\Lambda)$. The significance of the notion is summarized in the following simple facts:

Lemma 5.1: Let $a \in \Lambda$. Then, for any b with $a \leq b \leq \bar{a}$, $P^*(b) = P^*(\bar{a})$ and, therefore, also $\tilde{\kappa}^*(b) = \tilde{\kappa}^*(\bar{a})$ holds, whereas, if $b > \bar{a}$, $P^*(b) < P^*(a)$ and $\tilde{\kappa}^*(b) > \tilde{\kappa}^*(a)$ hold. [*Proof in appendix*]

Together with other facts, the lemma is used for the proof of the following useful result:

Proposition 5.2: Every minimal node of Λ is active. The relativized universal code is non-negative and vanishes on the minimal nodes – and nowhere else. The universal predictor assumes its maximal value on every minimal node and any other node has a strictly smaller probability. [*Proof in appendix*]

The proposition illuminates the definition of $\tilde{\kappa}^*$. Indeed, we realize that $\tilde{\kappa}^*$ measures code-length relative to the shortest codeword. This property is specific to co-trees and need not hold if relativization is considered more generally.

As a last application of the structure related to the notion of control, we establish the following result:

Proposition 5.3: For any node which is not active, the inequality of (14) is sharp, i.e. for such a node, $\tilde{\kappa}^{\sigma}(a) < \overline{N}(a)$. [*Proof in appendix*]

VI. IDEAS ON THE WAY TO AN ALGORITHM

Again, we consider the order model \mathcal{P} for a co-tree Λ . We aim at developing an efficient algorithm for the determination of the universal objects. Instead of going directly into this we shall take time in this section first to explain the ideas behind.

First, in order to motivate the introduction of blocking sets and brackets, we observe that if, somehow, the spectrum $\sigma(\Lambda)$ is known, $\tilde{\kappa}^*$ is easy to calculate. As $\tilde{\kappa}^*(a) = \tilde{\kappa}^*(\bar{a})$ holds generally, we need only worry about the values of $\tilde{\kappa}^*$ for active nodes. So, let $a \in \sigma(\Lambda)$. If a is minimal, $\tilde{\kappa}^*(a) = 0$. If a is not minimal, denote by T the set of maximal nodes in $(a^\downarrow \cap \sigma(\Lambda)) \setminus \{a\}$. By Proposition 5.2, T is a “cross-section” of a^\downarrow in the sense that every path from a to a minimal node meets T in exactly one point. We put $B = \bigcup_{t \in T} t^\downarrow$ and $S = a^\downarrow \setminus B$. Then $\tilde{\kappa}^*$ assumes the same value, $\tilde{\kappa}^*(a)$, on all nodes in S

and, considering the decomposition of a^\downarrow in the two sets S and B , we find from Proposition 5.1 that

$$\begin{aligned}\overline{N}(a) &= (\tilde{\kappa}^*)^\sigma(a^\downarrow) = |S| \cdot \tilde{\kappa}^*(a) + (\tilde{\kappa}^*)^\sigma(B) \\ &= |S| \cdot \tilde{\kappa}^*(a) + \sum_{t \in T} (\tilde{\kappa}^*)^\sigma(t^\downarrow) = |S| \cdot \tilde{\kappa}^*(a) + \sum_{t \in T} \overline{N}(t) \\ &= |S| \cdot \tilde{\kappa}^*(a) + \overline{N}^\sigma(T),\end{aligned}$$

and conclude that

$$\tilde{\kappa}^*(a) = \frac{\overline{N}(a) - \overline{N}^\sigma(T)}{|S|} = \frac{\overline{N}(a) - \overline{N}^\sigma(T)}{N(a) - N^\sigma(T)}, \quad (15)$$

recognizable as a bracket according to the definition (12). Note that the formula holds for all active nodes, including the minimal ones (for which $T = B = \emptyset$ and $S = \{a\}$).

The further development depends on certain relations between blocking sets and their associated brackets. The properties we need are derived from certain *transitivity identities*, stated in Lemma 7.1. Of special interest are blocking sets with maximal brackets. As indicated in Section IV, for each node there exists a set-theoretically largest blocking set for a with maximal bracket. This is the set $B^*(a)$ and the associated exterior and ceiling are the sets $S^*(a)$ and $T^*(a)$.

We can now define sets T_0^*, T_1^*, \dots , the *ceiling hierarchy*, by a construction “from the top”: We start with T_0^* , by definition the set of maximal nodes of Λ . Then, as T_1^* , we take the union of all sets of the form $T^*(t)$ with $t \in T_0^*$. We continue “down the co-tree”. Formally, for $i \geq 1$, we put

$$T_i^* = \bigcup_{t \in T_{i-1}^*} T^*(t). \quad (16)$$

Clearly, the sets T_i^* are eventually empty. By $\overline{\sigma}(\Lambda)$ we denote the union

$$\overline{\sigma}(\Lambda) = \bigcup_{i \geq 0} T_i^*. \quad (17)$$

For any node a , the *projection* of a on $\overline{\sigma}(\Lambda)$ is the unique node $pr(a) \in \overline{\sigma}(\Lambda)$ for which $a \in S^*(pr(a))$. We do not know if this notion coincides with the notion of control, i.e. if $pr(a) = \bar{a}$ holds generally. Anyhow, it is sufficiently close that we can argue with it in much the same way as in the beginning of this section, thereby deriving a formula for $\tilde{\kappa}^*$.

To any node $s \in \overline{\sigma}(\Lambda) \setminus T_0^*$ we associate the unique node $\mu(t) \in \overline{\sigma}(\Lambda)$ for which $s \in T^*(\mu(t))$ (it is the “mother” of s).

Using the notions just introduced, we can, in our first main result, Theorem 7.1, characterize the universal code as well as the spectrum $\sigma(\Lambda)$. We point out that perhaps $\overline{\sigma}(\Lambda) = \sigma(\Lambda)$ holds generally, but we do not know this. In spite of this unsettled issue, Theorem 7.1 is satisfactory as no essential saving in efficiency seems to result if $\overline{\sigma}(\Lambda) = \sigma(\Lambda)$ was known to hold.

The construction behind Theorem 7.1 depends on the blocking sets $B^*(a)$. During Part I of the algorithm all these blocking sets will be constructed. A naive search will require exponential time in the size of the problem. To develop an efficient algorithm, new ideas are needed. What we will do is to revert the construction and work “from the bottom” through the *minimality components* M_0, M_1, \dots, M_h . Here, h is the

height of Λ and the decomposition $\Lambda = M_0 \cup M_1 \cup \dots \cup M_h$ is obtained by successive removals of minimal nodes, i.e. M_i is the set of minimal nodes of the co-tree

$$\Lambda \setminus \bigcup_{0 \leq j < i} M_j.$$

The reason why a construction from the bottom is to prefer is that when you work from the top, and consider candidates for the B^* -, S^* - and T^* -sets without knowing these sets for nodes further down the co-tree, you risk that after some time an inconsistency occurs and this will force you to discard previous work, and to start afresh. Quite differently, when you work from the bottom, the sets concerned remain unchanged once constructed as they are not influenced by the development further up in the co-tree. It should, however, be remarked that sets already constructed may later turn out to be superfluous as sets associated with nodes higher up in the co-tree, say nodes $b > a$, may “overshadow” sets already constructed in the sense that $S^*(b) \supseteq S^*(a)$ may happen, cf. the discussion of $\Lambda[1, 2, 4]$ in Figure 5. The insight needed to see that Part I of the algorithm works as intended will be developed in Section VIII.

A basic element of the algorithm is a subroutine, referred to as the *central subroutine*. It is called several times during Part I. The flow diagram is sketched in Figure 6. As input to the subroutine one takes a node $a \in \Lambda$, and as output the subroutine provides you with $B^*(a)$, $T^*(a)$ and $[a]_{\max} = [a, B^*(a)]$. It is understood that the corresponding objects associated with nodes in $a^\downarrow \setminus \{a\}$ are already known when the subroutine for a is called. When the central subroutine has been called for all nodes in the co-tree as input, all ceilings $T^*(a)$ and all maximal brackets $[a]_{\max}$ will be known and Part I of the overall algorithm is completed. For the final part of the algorithm, Part II, we work “from the top” by appealing to Theorem 7.1. This provides you directly with the relativized universal code from which the universal code (hence also the universal predictor) may easily be constructed as explained in Section V.

VII. CONSTRUCTION FROM THE TOP

We start by developing some properties of blocking sets and brackets.

Proposition 7.1: Let B be a blocking set for a . Then the bracket $[a, B]$ vanishes if a is a minimal node and is positive otherwise. [*Proof in appendix*]

We shall show that the universal code can be constructed based on the brackets alone. Proposition 5.1 is an important step in this direction but there are obstacles to overcome in connection with the necessary checking of inequalities related both to (14) and to the requirement of monotonicity. It turns out that these problems can be overcome, based on certain identities which allows one to compare brackets among each other. Two simple constructions are involved, *filling* and *restriction*. Specifically, if B is a subset of Λ , and b any node, the *filling of B at b* is the set $B \vee b = B \cup b^\downarrow$ and the *restriction of B to b^\downarrow* is the set $B \wedge b = B \cap b^\downarrow$. Typically, these constructions are used if B is a blocking set for a , $b < a$

and $b \notin B$. Then $B \vee b$ is a new blocking set for a and $B \wedge b$ is a blocking set for b .

Lemma 7.1 (transitivity identities, basic case): Let $a > b$, let B be a blocking set for a and assume that $b \notin B$. Put $B^+ = B \vee b$ and $B^- = B \wedge b$. Then the following identities hold:

$$|S_{B^+}(a)| \left([b, B^-] - [a, B^+] \right) = |S_B(a)| \left([b, B^-] - [a, B] \right), \quad (18)$$

$$|S_B(a)| \left([a, B] - [a, B^+] \right) = |S_{B^-}(b)| \left([b, B^-] - [a, B^+] \right), \quad (19)$$

$$|S_{B^-}(b)| \left([b, B^-] - [a, B] \right) = |S_{B^+}(a)| \left([a, B] - [a, B^+] \right). \quad (20)$$

[Proof in appendix]

In order to ease the notation, we agree that if a set of the form $B \wedge b$ is blocking for b , we may say that B is blocking for b and write $S_B(b)$ in place of $S_{B \wedge b}(b)$ and $[b, B]$ in place of $[b, B \wedge b]$. In the formulation of Lemma 7.1 we may thus write $S_B(b)$ rather than $S_{B^-}(b)$ and $[b, B]$ rather than $[b, B^-]$.

As all terms of the form $|S(\cdot)|$ are positive, it is clear that we can use (18)-(20) for comparisons of brackets. We shall soon see instances of this. For now we note that the lemma implies that the numbers $[a, B]$, $[a, B^+]$ and $[b, B]$ are either identical or else $[a, B]$ lies strictly between $[a, B^+]$ and $[b, B]$, i.e. either $[a, B^+] < [a, B] < [b, B]$ or $[b, B] < [a, B] < [a, B^+]$ holds.

The transitive nature of the lemma is best revealed by generalizing the result⁵.

With reference to notions from Section IV, a blocking set for $a \in \Lambda$ with maximal bracket is *set-theoretically maximal (minimal)* if it is not a proper subset (superset) of some other blocking set for a with maximal bracket.

Proposition 7.2: Let B^* be a blocking set for $a \in \Lambda$ with maximal bracket.

(i) (monotonicity): The inequality $[a]_{\max} \geq [b]_{\max}$ holds for every $b \in T_{B^*}(a)$. If B^* is set-theoretically minimal, the sharp inequality $[a]_{\max} > [b]_{\max}$ holds;

(ii) (boundedness): The inequality $[a]_{\max} \leq [b, B^*]$ holds for every $b \in S_{B^*}(a) \setminus \{a\}$, and the inequality is sharp if B^* is set-theoretically maximal. [Proof in appendix]

Exploiting these results we obtain a useful uniqueness property:

Proposition 7.3: (uniqueness) For every node a , there exist two uniquely defined blocking sets for a with maximal bracket,

⁵what we have in mind is the following result, which can be proved by induction: Let $k \geq 2$ and consider nodes a_1, \dots, a_k with $a_1 > \dots > a_k$. Assume that B is a blocking set for a_1 and that $a_k \notin B$. Put $B_i = B \vee a_i$ for $i = 2, \dots, k$ and $B_{k+1} = B$. Then

$$\begin{aligned} & \sum_{i=2}^k |S_{B_i}(a_1)| \left([a_{i-1}, B_i] - [a_i, B_{i+1}] \right) \\ &= |S_B(a_1)| \left([a_1, B] - [a_k, B] \right). \end{aligned}$$

$B^*(a)$ and $B_*(a)$, characterized as, respectively the set-theoretically largest such set and the set-theoretically smallest such set. In particular, for every blocking set B for a with maximal bracket, the inclusions $B_*(a) \subseteq B \subseteq B^*(a)$ hold. [Proof in appendix]

We do not know if $\sigma(\Lambda) = \bar{\sigma}(\Lambda)$. This will be the case if, for $a \in \Lambda$, there is a unique blocking set for a with maximal bracket, i.e. if $B_*(a) = B^*(a)$ holds generally.

For the constructions to follow, we have chosen to focus on the largest sets, the $B^*(a)$'s. We denote by $T^*(a)$ the ceiling in a associated with $B^*(a)$ and by $S^*(a)$ the exterior in a associated with $B^*(a)$. These are the sets we shall use for the construction of $\tilde{\kappa}^*$.

Consider the ceiling hierarchy $(T_i^*)_{i \geq 0}$ introduced in Section VI. Here and below, the largest index with $T_i^* \neq \emptyset$ is denoted δ . Clearly, $\delta \leq h$, the height h of Λ , but often it is smaller, e.g. for Case 3 of Figure 1, $\delta = 1$ and $h = 2$. In the extreme case when every maximal node is also a minimal node, T_0^* is the only non-empty set in the hierarchy and $\delta = 0$.

Based on the ceiling hierarchy we define a decomposition $(S_i^*)_{0 \leq i \leq \delta}$ of Λ as follows:

$$S_i^* = \bigcup_{a \in T_i^*} S^*(a) = \{a \in \Lambda \mid pr(a) \in T_i^*\}. \quad (21)$$

With reference to the ceiling hierarchy and associated notions, we can now state the main result of this section:

Theorem 7.1: The relativized universal code is given by

$$\tilde{\kappa}^*(a) = [pr(a)]_{\max} \text{ for all } a \in \Lambda, \quad (22)$$

and the spectrum of Λ is the following subset of $\bar{\sigma}(\Lambda)$:

$$\sigma(\Lambda) = T_0^* \cup \{t \in \bar{\sigma}(\Lambda) \setminus T_0^* \mid [\mu(t)]_{\max} > [t]_{\max}\}. \quad (23)$$

[Proof in appendix]

VIII. THE CENTRAL SUBROUTINE

We continue the study of universal objects associated with the model $\mathcal{P}(\Lambda)$ over a co-tree Λ .

The construction in Theorem 7.1 builds on the sets $B^*(a)$. As noted in Section VI, the theorem cannot be used directly to obtain an algorithm of low complexity. Instead, we speed up the construction by working ‘‘from the bottom’’ based on the decomposition $\Lambda = M_0 \cup M_1 \cup \dots \cup M_h$ in minimality components.

We shall determine the B^* -sets for all nodes. For nodes in M_0 this is trivial, and we start by considering nodes in M_1 , continue with nodes in M_2 , and so on until we get at the nodes in M_h . We will assume that the decomposition in minimality components is given off-hand and not be concerned with the time it takes to determine this decomposition. Anyhow, this can be achieved by an efficient algorithm based on a systematic indexing of the nodes as in the examples shown in Figure 1.

The two propositions to follow are important technical tools needed to develop an efficient algorithm.

Proposition 8.1: (Γ -structure) Let $a \in \Lambda$. Then, for every $b \in S^*(a)$, the inclusion $S^*(b) \subseteq S^*(a)$ or, equivalently, $B^*(a) \wedge b \subseteq B^*(b)$ holds. [Proof in appendix]

The name attached to the result lies in the shape of the letter “T” and will appear natural when we specialize to co-trees with uniform branching in the next section.

The stronger result actually proved in the appendix supports the view that “normally” $B_*(a) = B^*(a)$.

For our second auxiliary result, let us agree to say that a blocking set B for a node a has the *monotonicity property* if $[a, B] \geq [t]_{\max}$ for every $t \in T_B(a)$.

Proposition 8.2: (Characterization): For any $a \in \Lambda$, $B^*(a)$ can be characterized as the largest blocking set for a with the monotonicity property. [Proof in appendix]

When applying this result we have a construction “from the bottom” in mind. Then the characterization makes good sense since, when searching for the set $B^*(a)$, all sets $B^*(b)$ with $b \in a^\downarrow \setminus \{a\}$ will be known and thus the monotonicity property can be checked for any candidate set B we may suggest for $B^*(a)$.

We emphasize that when estimating the complexity of the algorithms under development, we will neglect any contribution from efforts to make basic information about co-trees studied accessible to us in a convenient form. We shall thus talk about *essential complexity* of the algorithms. The basic information we will need can be listed as follows:

- the decomposition in minimality components, $\Lambda = M_0 \cup \dots \cup M_h$,
- the map $a \rightsquigarrow a^-$ which makes the immediate predecessors of any node accessible to us,
- the map $a \rightsquigarrow a^\downarrow$ which gives access to the left sections,
- the map $a \rightsquigarrow N(a)$ and, finally,
- the map $a \rightsquigarrow \overline{N}(a)$.

Of course, there is some redundancy in this list (especially, \overline{N} is given in terms of N). However, the list is chosen for convenience in view of the algorithm to follow. It is clear that if we identify a co-tree using the standard representation, the basic information can be provided by efficient algorithms operating on the underlying set of finite sequences.

The algorithm we shall now describe is based on Theorem 7.1 which shows that if we know, for every node a , the ceiling $T^*(a)$ as well as the maximal bracket $[a]_{\max}$, then it is easy to determine the relativized universal code, and hence the universal code and the universal predictor. The algorithm calls several times the *central subroutine*, see Figure 6, which, for a given input a , calculates the key objects associated with a , taken to be the sets $B^*(a)$ and $T^*(a)$ and the number $[a]_{\max}$. Note that we find it convenient to work with both $B^*(a)$ and $T^*(a)$, though the one may of course be determined from the other.

For the minimal nodes $a \in M_0$, we already know what the key objects are and there is no reason to call any subroutine for these nodes. To determine the key objects associated with any node, we first call the central subroutine for nodes in the minimality component M_1 , then for nodes in M_2 and so on until we get to the nodes in M_h (with h the height of Λ).

Let us have a closer look at the central subroutine. Consider a particular input $a \in \Lambda \setminus M_0$. When the subroutine is called it is assumed that key objects about preceding nodes have already been determined. This will be the case by the

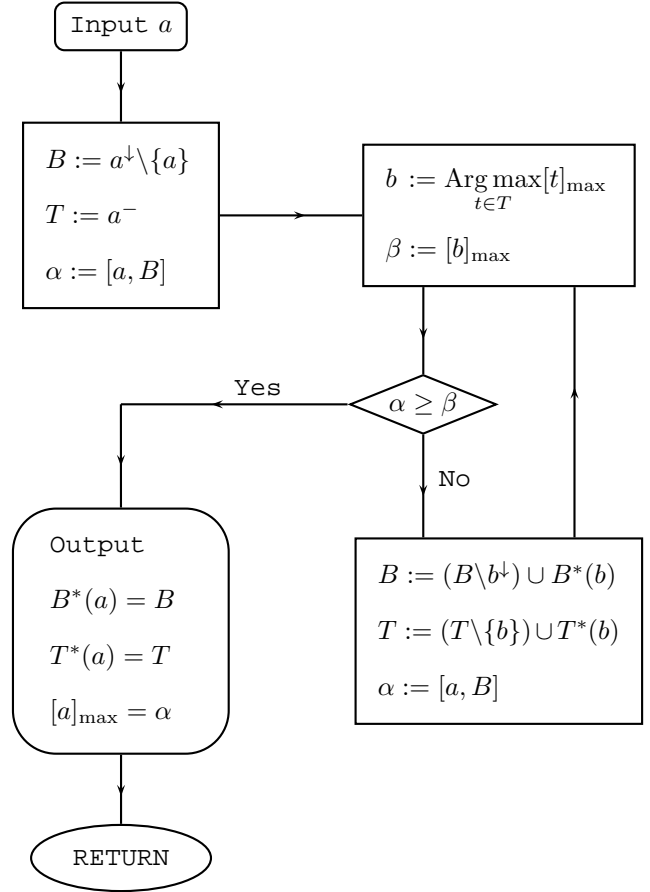


Figure 6. Flow diagram for the central subroutine

procedure chosen as nodes in $(M_0), M_1, \dots, M_h$ are called in succession.

We use B , T and α as place-holders for the sought key objects associated with a . The largest blocking set for a altogether is $a^\downarrow \setminus \{a\}$. This is the first set we will test and our initial assignment box puts $B := a^\downarrow \setminus \{a\}$. We also right away assign the appropriate set to T and the appropriate value to α .

After the introductory assignments, we arrive at the central box, the (b, β) -box. It is important that when we come to this box, which may occur many times during the execution of the subroutine, B , T , and α are known to have certain properties: B must be a blocking set for a , $T = T_B(a)$ and $\alpha = [a, B]$ must hold, and then we stress that $B^*(a) \subseteq B$ must be known to hold. In order to carry out the calculations in the (b, β) -box, it is understood that there is a natural way to list the nodes in T , say as t_1, \dots, t_k (the standard representation of Λ may be used for this purpose). For the calculation, we go through all brackets $[t]_{\max}$ with $t \in T$, note the largest value and then consider the first node among t_1, \dots, t_k for which the corresponding bracket attains this value. By definition, this is the Arg max -node. As place-holders for this node and for the corresponding maximal bracket we use b , respectively β and

thus carry out the assignments

$$b := \text{Arg max}_{t \in T} [t]_{\max}; \beta := [b]_{\max}.$$

Concerning the calculation of brackets in the central box and elsewhere in the subroutine, this is based on basic information about the co-tree (N 's and \overline{N} 's) and on output (T^* 's) from previous calls of the subroutine according to the formula

$$[t]_{\max} = \frac{\overline{N}(t) - \sum_{s \in T^*(t)} \overline{N}(s)}{N(t) - \sum_{s \in T^*(t)} N(s)}. \quad (24)$$

After the central box comes the test-box “ $\alpha \geq \beta$?”. We realize that what is tested is really if B has the monotonicity property. If it does, $B = B^*(a)$ by Proposition 8.2 and we go to the output box and then return to the algorithm.

Assume now that the test is negative, i.e. $[a, B] < [b]_{\max}$. It is a key point of the algorithm that then $b \in S^*(a)$ must hold. Assume the contrary. Then, as $B^*(a) \subseteq B$, $b \in T^*(a)$ and by monotonicity we then have $[a]_{\max} \geq [b]_{\max}$. Consider any $t \in T$ and note that

$$[t]_{\max} \leq [b]_{\max} \leq [a]_{\max}.$$

By boundedness, we must conclude from this that $t \in B^*(a)$ since, if $t \in S^*(a)$, $[a]_{\max} < [t, B^*(a)] \leq [t]_{\max}$ would hold, contradicting the inequalities above. Thus $T \subseteq B^*(a)$. Since T is the ceiling of B in a and since $B^*(a) \subseteq B$ we conclude that in fact $B = B^*(a)$ must hold. Then B does after all have the monotonicity property of Proposition 8.2. This contradicts the result of the test. All in all we conclude that indeed $b \in S^*(a)$.

Knowing this, we can apply the gamma structure, Proposition 8.1, and find that $B^*(a)$ is a subset of the set $(B \setminus \{b\}) \cup B^*(b)$. This set is a blocking set for a as b cannot be a minimal node (then $\beta = 0$ would hold and the test would have been positive). We take this set as our new set to be tested and make the proper assignments of B , T and α in the next box of the flow diagram. These possible key objects are then fed into the (b, β) -box and we continue until, eventually, the test for the monotonicity property is positive.

Remarks. Naturally, if the test is negative and there are several nodes in T with $[b, B^*(b)]$ maximal, we may economize and restrict the candidate set further. In more detail, put $m = \max_{t \in T} [t]_{\max}$ and assume that there are several nodes in T , say b_1, \dots, b_k with maximal bracket m . Then we may as our new assigned key objects take

$$B := \left(B \setminus \bigcup_{\nu=1}^k b_\nu^\downarrow \right) \cup \bigcup_{\nu=1}^k B^*(b_\nu), \quad (25)$$

$$T := \left(T \setminus \bigcup_{\nu=1}^k \{b_\nu\} \right) \cup \bigcup_{\nu=1}^k T^*(b_\nu), \quad (26)$$

$$\alpha := [a, B]. \quad (27)$$

It follows from our analysis above that the new set B still contains $B^*(a)$. Further, the α 's increase through the subroutine. One way to see this when multiple reductions are performed as in (25)-(27) is to make the reductions step

by step. First, put $B_0 = B$ (the old set B) and then define successive reductions by putting

$$B_\nu = (B_{\nu-1} \setminus b_\nu^\downarrow) \cup B^*(b_\nu)$$

for $\nu = 1, \dots, k$. Then the set B_k is equal to the set defined in (25). This relies on successive applications of (19) and on monotonicity. Details are left to the reader. This remark will be important for the special co-trees to be discussed in the next section.

Other modifications may speed up the execution of the subroutine, e.g. one may note that nodes in M_1 can also, just as minimal nodes, be dealt with outside the subroutine and that some of the information about calculated brackets $[t]_{\max}$ at one stage may be reused for the next stage. We shall not be concerned here with such fine-tunings for general co-trees.

The full algorithm for the calculation of $\tilde{\kappa}^*$, and hence the sought universal objects, consists of the following steps:

- initialization providing basic information about the co-tree,
- trivial assignment of key objects to nodes in M_0 ,
- call of the central subroutine for all nodes in M_1 ,
- ...
- call of the central subroutine for all nodes in M_h ,
- top-down construction of the ceiling hierarchy and simultaneous listing of the values of $\tilde{\kappa}^*$, cf. Theorem 7.1.

The steps until the final step constitutes Part I of the algorithm. The final step is Part II.

By the foregoing discussion, it is clear that the algorithm does indeed calculate the desired objects. It is also pretty clear that this is achieved in polynomial time in the size of the problem. Let us discuss this in more detail but only aim at a rough estimate of the efficiency of the algorithm. Firstly, as remarked before, we shall neglect the time consumed during initialization. Also, we shall not be concerned with the memory requirements of the algorithm or with the extra cost incurred by administrative operations involved in the memory management. Further, we shall not discriminate between various basic operations such as additions, subtractions, multiplications, divisions and comparisons of numbers as well as 0, 1-tests (based on known entities). The *essential complexity* of the algorithm, denoted $C(\Lambda)$, is then taken to be the number of basic operations needed from start to end of the algorithm with the reservations as indicated above.

We shall estimate $C(\Lambda)$ in terms of the number n of nodes in Λ . Clearly, $C(\Lambda) \leq n \cdot \max_{a \in \Lambda} C(a)$ where $C(a)$ denotes the essential complexity of the central subroutine when it is called with the node a as input.

For a fixed, we can estimate $C(a)$. Regarding the initial assignments, only the calculation of $[a, B]$ needs to be taken into account. As $[a, B] = \overline{N}(a) - \sum_{t \in a^-} \overline{N}(t)$, at most $|a^-|$ basic operations are needed, hence at most n such operations.

For the cycle “ (b, β) -box to test-box to new assignments”, this will be visited at most $|a^\downarrow|$ many times, hence at most n times. And for one run through the cycle we need at most $|T| \leq n$ basic operations for the determination of (b, β) (as the numbers $[t]_{\max}$ with $t \in T$ are already known). We permit

ourselves to ignore the minimal requirement needed to carry out the $\alpha \geq \beta$ test. But we have to consider the requirement related to the new assignments of B, T and α . Regarding B , we need to know, for each node, whether the node is in the set or not. This can be decided by checking membership for each of the three sets B, b^\downarrow and $B^*(b)$. As the sets b^\downarrow and $B^*(b)$ are known, we only need to test membership for B , and this requires at most n tests. Similarly for T . And regarding α , we realize from (15) that at most $2 \cdot |T| \leq 2n$ basic operations are needed. The new assignments thus require at most $4n$ basic operations.

The rough estimates above show that $C(a) \leq n + n(n + 4n) \leq 6n^2$.

We have now completed all elements in the proof of our second main theorem:

Theorem 8.1: The algorithm described above calculates the ceiling hierarchy and thereby the universal objects associated with a co-tree Λ in polynomial time. The essential complexity as defined above is at most $6 \cdot n^3$ where n is the number of nodes in Λ .

Remark. By studying “worst possible scenarios” regarding the possibilities for the geometric locations of the ceilings calculated by the central subroutine it should be possible to bring down the estimate $6n^3$ quite significantly. We shall look into this in Section IX, but only for co-trees with uniform branching.

IX. CO-TREES WITH UNIFORM BRANCHING

Consider a co-tree Λ of height n with uniform branching. Let (k_1, \dots, k_n) be the branching pattern. Denote by Λ_ν the set of all nodes in level ν . Put $K_\nu = |\Lambda_\nu|$ and, for a node $a \in \Lambda_\nu$, put $N_\nu = N(a)$ and $\bar{N}_\nu = \bar{N}(a)$. Clearly, $K_\nu = k_1 \cdots k_\nu$, thus, recursively,

$$K_0 = 1, \quad K_\nu = k_\nu K_{\nu-1} \text{ for } \nu = 1, \dots, n. \quad (28)$$

Regarding the convenient calculation of the N_ν 's, see (8).

For the determination of $\tilde{\kappa}^*$, we shall specialize the algorithm of the previous section to the present situation of a co-tree with uniform branching. For reasons of symmetry – see also the discussion related to (25)-(27) – we need only work with certain special blocking sets. By $[\nu, \mu]$ we denote the bracket $[a, B]$ for a node $a \in \Lambda_\nu$ with the blocking set $B = a^\downarrow \cap \bigcup_{i \geq \mu} \Lambda_i$ for which then $T_B(a) = a^\downarrow \cap \Lambda_\mu$. These brackets are well-defined for points (ν, μ) with $0 \leq \nu \leq n-1$ and $\nu+1 \leq \mu \leq n$. We extend the definition by adding the point $(n, n+1)$. This point represents a minimal node and the empty blocking set. Therefore, we put $[n, n+1] = 0$. For all other brackets we find that

$$[\nu, \mu] = \frac{\bar{N}_\nu - k_{\nu+1} \cdots k_\mu \bar{N}_\mu}{N_\nu - k_{\nu+1} \cdots k_\mu N_\mu}. \quad (29)$$

$$= \frac{K_\nu \bar{N}_\nu - K_\mu \bar{N}_\mu}{K_\nu N_\nu - K_\mu N_\mu}. \quad (30)$$

The *bracket diagram* consists of all these brackets. A numerical example is shown in Table 2.

Given ν , define $[\nu]_{\max}$ and τ_ν by

$$[\nu]_{\max} = \max_{\mu > \nu} [\nu, \mu], \quad (31)$$

$$\tau_\nu = \text{Arg max}_{\mu > \nu} [\nu, \mu]. \quad (32)$$

Then, for a node $a \in \Lambda_\nu$, $T^*(a) = a^\downarrow \cap \Lambda_{\tau_\mu}$ ⁶. The numbers $[\nu]_{\max}$ are the *maximal brackets* and the τ_ν 's are the *ceiling numbers*.

									0.00
8	12.62	12.69	12.76	12.84	9.55	5.76	6.59	5.55	8
7	18.51	18.77	19.04	19.32	13.19	5.97	8.68		7
6	25.53	26.24	27.00	27.83	16.94	3.25			6
5	81.21	91.91	106.17	126.14	85.39				5
4	77.03	100.59	147.73	289.12					4
3	6.33	6.33	6.33	3					
2	6.33	6.33	2						
1	6.34	1							
μ/ν									0

Table 2. Bracket diagram for $\Lambda[1, 1, 1, 4, 5, 1, 2, 3]$

The ceiling numbers can be determined directly from the bracket diagram. For instance, for $\Lambda[1, 1, 1, 4, 5, 1, 2, 3]$, we find from the column in Table 2 with $\nu = 2$ that $\tau_2 = 4$ and that $[2]_{\max} \approx 147.73$. Then, by Theorem 7.1, the nodes in levels 0, 5, 7 and 8 are the active nodes. Further, the values of $\tilde{\kappa}^*$ for nodes in levels 0,1,2,3 and 4 is 81.21 and the values of $\tilde{\kappa}^*$ for nodes in levels 5,6,7 and 8 are, respectively 5.97, 5.97, 5.55 and 0.

Using the strategy as exemplified above for the calculation of $\tilde{\kappa}^*$, the full bracket diagram must be calculated and this amounts to about $n^2/2$ basic computations. This can be improved considerably by appeal to the algorithm developed in Section VIII. For $\Lambda[1, 1, 1, 4, 5, 1, 2, 3]$ one may for instance reduce the number of calculations of brackets from 36 (corresponding to Table 2) to 13 (will follow from results below). The basic facts we need are Propositions 8.1 and 8.2. The algorithm dictates that the bracket diagram is calculated for descending values of ν and ascending values of μ . To initialize, one sets $[n]_{\max} = 0$ and $\tau_n = n + 1$. Then one calculates in succession $[n-1]_{\max}$ and τ_{n-1} , then $[n-2]_{\max}$ and τ_{n-2} and so on until $[0]_{\max}$ and τ_0 are calculated. On the way, the only tests that are performed are of the type “ $[\nu, \mu] \geq [\mu, \tau_\mu]$?” and, in fact, not all these tests have to be performed as the result is bound to be negative (and hence $\tau_\nu > \mu$) in case, for a value of ξ with $\nu < \xi < \mu$, one has already found that $\tau_\xi > \mu$. This follows by Proposition 8.1.

In order to study more closely which tests can be neglected and which not, we introduce the abstract notion of a Γ -*diagram*. These diagrams are first discussed in their own right. After having developed a main property, Lemma 9.1 below, we return to the actual problem concerning co-trees.

Given are natural numbers t_0, \dots, t_n with $n \geq 1$ such that:

$$t_n = n + 1, \quad (33)$$

$$\nu + 1 \leq t_\nu \leq n \text{ for all } 0 \leq \nu \leq n - 1, \quad (34)$$

$$\text{if } \nu \leq \mu < t_\nu, \text{ then } t_\mu \leq t_\nu. \quad (35)$$

⁶to be sure, the *Argmax* in (32) has to be understood as the first index for which the maximum is reached, since we have not been able to exclude the possibility that the maximum is reached for several values of μ .

Then the Γ -diagram $G = G(t_0, \dots, t_n)$ consists of all points (ν, μ) with $0 \leq \nu \leq n$ for which $\nu + 1 \leq \mu \leq t_\nu$. More precisely, G is a Γ_n -diagram and n is the *height* of G . As a singular case we allow that $n = 0$. There is only one Γ_0 -diagram, the *trivial diagram* consisting only of the point $(0, 1)$.

By (35), if you consider the column from $(\nu, \nu + 1)$ to (ν, t_ν) and place a horizontal bar on top of and to the right of (ν, t_ν) then you meet no points in G until you reach the diagonal element $(t_\nu, t_\nu + 1)$. Having the shape of the letter “ Γ ” in mind, this property accounts for the terminology “ Γ -diagram”. For a possibly more illuminating way of expressing the key property, see below.

A site $(\nu_0, \mu_0) \in G = G(t_0, \dots, t_n)$ is a *test site* for G , if $\nu < n$ and $G(s_0, \dots, s_n)$ is also a Γ_n -diagram where all the s_i are equal to t_i except s_ν which is set to μ . For example, all sites (ν, τ_ν) and $(\nu, \nu + 1)$ with $\nu < n$ are test sites. For the Γ_{10} -diagram displayed in Figure 8, we have 17 test sites, corresponding to the marked positions. For a general Γ -diagram G , we denote by $\langle G \rangle$ the number of test sites.

Two operations on Γ -diagrams are considered: The *restriction* of $G(t_0, \dots, t_n)$ to $\{\nu, \dots, n\}$ is the $\Gamma_{n-\nu}$ -diagram $G(t_\nu - \nu, \dots, t_n - \nu)$ and the *direct sum* of the two Γ -diagrams $G(t_0, \dots, t_n)$ and $G(s_0, \dots, s_m)$ is the Γ_{n+m} -diagram $G(t_0, \dots, t_{n-1}, s_0 + n, \dots, s_m + n)$. Figure 10, page 15 provides an example of a direct sum.

For a Γ_n -structure $G = G(t_0, \dots, t_n)$ we define the *spectral levels* $\sigma_0, \dots, \sigma_\gamma$ by $\sigma_0 = 0, \sigma_i = t_{\sigma_{i-1}}$ for all values of $i \geq 1$ until you reach the index γ with $\sigma_\gamma = n$. We call $\gamma = \gamma(G)$ the *spectral index* of G . The spectral index of the trivial Γ -structure is 0, all other Γ -structures have positive spectral indices.

Note that the spectral levels $\sigma_0, \dots, \sigma_\gamma$ can be constructed geometrically as indicated in Figure 9 by “letting the sun shine from the left” and noting the column numbers of the sunlit columns. The spectral index $\gamma(G)$ is the number of sunlit columns minus 1. Using the “sunshine terminology” we can also express the essential Γ -structure, formally given by the requirement (35), by saying that when the sun illuminates part of a column, it illuminates the entire column. And this property must also hold for restrictions of the Γ -diagram.

The combinatorial result we need is the following:

Lemma 9.1: For any Γ_n -structure G , $\langle G \rangle = 2n - \gamma(G)$, in particular, $\langle G \rangle \leq 2n$. [Proof in appendix]

After this excursion into combinatorics we return to the study of a given co-tree $\Lambda = \Lambda[k_1, \dots, k_n]$ with ceiling numbers τ_0, \dots, τ_n . The Γ -diagram associated with Λ is the diagram $G = G(\tau_0, \dots, \tau_n)$. That this is indeed a Γ -diagram follows from Proposition 8.1.⁷

The algorithm we shall discuss consists of three parts:

- initialization,
- construction of the Γ -diagram,
- determination of the spectral levels, final output.

⁷In passing, we conjecture that every Γ -diagram can arise in this way. To illustrate the conjecture, observe that there are 5 Γ_3 -diagrams and these may be realized as Γ -diagrams associated with the co-trees with branching patterns, respectively $(1, 1, 1)$, $(1, 1, 2)$, $(1, 2, 3)$, $(2, 1, 2)$ and $(1, 2, 4)$ (regarding the last pattern, see also Figure 5).

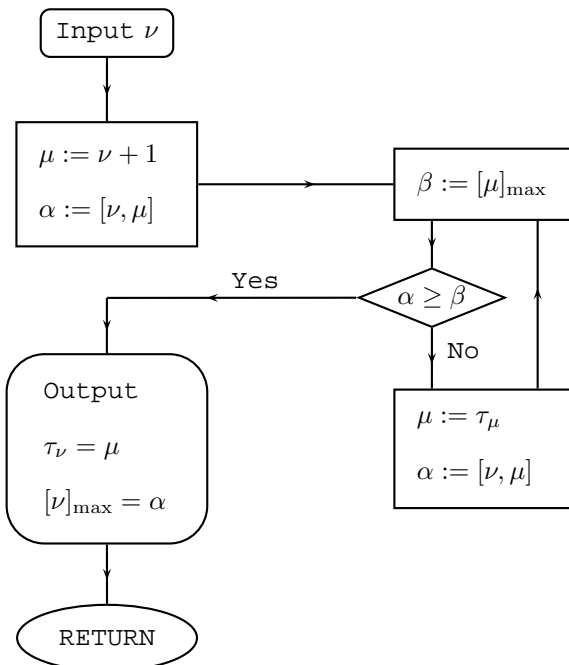


Figure 7. The central subroutine for co-trees with uniform branching

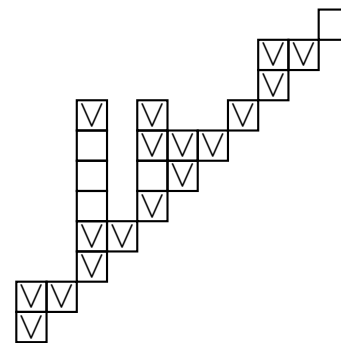


Figure 8. A Γ_{10} -diagram with test sites

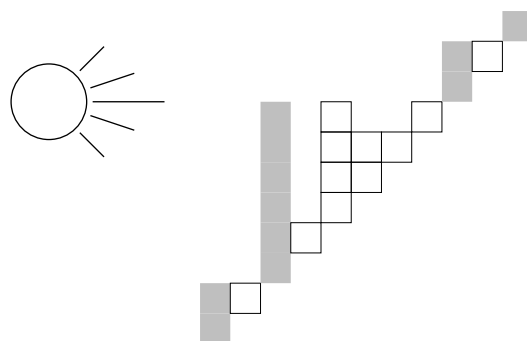


Figure 9. Sunlit columns for the diagram in Figure 8

The initialization consists of the calculation of the numbers N_ν , K_ν , $K_\nu N_\nu$ and $K_\nu \overline{N}_\nu$ for $\nu = 0, \dots, n$. For this, the formulas (8), (28) and (30) are used. In total, $4n$ basic operations are needed for the calculations. You may also consider as part of the initialization the assignment of start values $\tau_n = n + 1$ and $[n]_{\max} = 0$ for the next step in the algorithm.

The key part of the algorithm is the calculation of the Γ -diagram, i.e. the numbers τ_ν , as well as the calculation of the associated maximal brackets, the $[\nu]_{\max}$'s. This is achieved by successive calls of the *central subroutine*. Though basically the same as for general co-trees, there are essential simplifications as also indicated earlier. This is partly achieved by symmetry considerations, partly by our focus only on those sites in the Γ -diagram where we really have to make a test. The flow diagram is sketched in Figure 7. The subroutine is called for all ν , starting with the highest value, $n - 1$, and ending with the value $\nu = 0$. When all these calls have been made you realize that you only have to calculate a bracket (following (30)) and to perform a test corresponding to test sites of the Γ -diagram. Therefore, referring to Lemma 9.1, no more than $4n$ basic operations are involved in the calls of the subroutine (for this, a test “ $\alpha \geq \beta$?” as well as a calculation $\alpha := [\nu, \mu]$ is counted as a basic operation).

The final part of the algorithm is a “top-down” determination of the output of the algorithm, understood to be the spectral levels $\sigma_0 = 0, \sigma_1 = \tau_{\sigma_0}, \dots, \sigma_\gamma = n$ and the associated maximal brackets. We suggest that these data are listed in the form $(\sigma_0, [\sigma_0]_{\max}) = (0, [0]_{\max})$, $(\sigma_1, [\sigma_1]_{\max}), \dots, (\sigma_\gamma, [\sigma_\gamma]_{\max}) = (n, 0)$. Each such pair is considered to involve only one basic operation, thus adding at most n such operations. If you also want to calculate R_{\min} as part of the final output, another n basic operations are needed.

Considering the above discussion, we have proved our last main result which may be summarized as follows:

Theorem 9.1: Consider a co-tree $\Lambda = \Lambda[k_1, \dots, k_n]$. Apply the modification of the algorithm from Section VIII as described above. Then the number of tests performed during execution of the algorithm is at most $2n$ and the essential complexity of the entire algorithm, understood as the number of basic operations needed to carry out initialization, determination of the Γ -diagram and the listing of all pairs of spectral levels and associated maximal brackets is at most $9n$. [*Proof in appendix*]

X. CONCLUSIONS AND FINAL COMMENTS

The paper offers a reasonably complete study of algorithms for the precise determination of universal objects associated with the model of all distributions over a co-tree for which the implication $a < b \Rightarrow P(a) \geq P(b)$ holds. A relatively simple situation corresponds to the case when the universal predictor satisfies the implication above with strict inequality. The key result here is Theorem 3.1 with Corollary 3.2 as a natural extension of Ryabko's result from 1979.

However, the main results concern general co-trees. It appears convenient to introduce a notion of *relativization* applied

to codes. This concept is believed to be of interest also outside the scope of the present paper.

Theorem 7.1 provides basic insight into the structure of the universal objects, but quite some extra work is involved before a reasonable algorithm, presented in Theorem 8.1 is in house. Only a crude estimate of the complexity of that algorithm is discussed. For the special case of co-trees with uniform branching this is much refined. The main result is Theorem 9.1.

The key to the results are purely combinatorial facts, isolated in the transitivity identities in Lemma 7.1 for the general algorithm and supplied with a count of test sites in Lemma 9.1 for the special case of co-trees with uniform branching.⁸

It is a curious feature of the technical analysis that the logarithmic function only appears rather sporadically. Accordingly, other functions may be considered. Without going into details, this may result in computations of universal objects tied to other notions of entropy and divergence than the standard notions of pure Shannon theory.

Finally we note that the interesting connection to a problem of isotone regression which was pointed out by a referee, cf. the end of Section II, deserves a closer investigation.

XI. APPENDIX WITH PROOFS

Proof of Theorem 3.1: Assume that Λ has full spectrum. A straightforward analysis shows that the Kuhn-Tucker conditions of Proposition 1.2 can only be fulfilled with P^* given via W as described. As $P^* \in \mathcal{P}(\Lambda)$, the stated inequalities must hold. That, conversely, P^* is as stated when the inequalities hold amounts to simple checking based on Proposition 1.2. The formula $R_{\min} = \ln Z$ follows e.g. by noting that $D(U_a \| P^*) = \ln Z$ when a is a minimal node. ■

Proof of Corollary 3.2: Once we have proved that (9) holds when $k_1 \geq \dots \geq k_n$, the formula for P^* follows from Theorem 3.1 and (6). Note that the left hand side of (9), call it G , is a geometric mean and that the corresponding arithmetic mean is

$$A = \frac{1}{\rho_\nu} + \frac{\rho_\nu - N_\nu}{\rho_\nu N_{\nu+2}} = \frac{1 + k_{\nu+2}}{N_{\nu+1}(1 + k_{\nu+1})} \leq \frac{1}{N_{\nu+1}},$$

hence also $G \leq \frac{1}{N_{\nu+1}}$, i.e. (9) does indeed hold. ■

Proof of Proposition 5.1: We start with some preliminary observations related to any function ϕ on Λ . Put $R = \ln \sum_{a \in \Lambda} e^{-\phi(a)}$ and define κ as the code obtained from ϕ by “de-relativization”, i.e. $\kappa = \phi + R$. Then $\kappa \in \mathcal{K}(\Lambda)$. Let P denote the matching distribution. We claim that, for every node a , the equivalences

$$D(U_a \| P) = R \Leftrightarrow \phi^\sigma(a^\downarrow) = \overline{N}(a), \quad (36)$$

$$D(U_a \| P) \leq R \Leftrightarrow \phi^\sigma(a^\downarrow) \leq \overline{N}(a) \quad (37)$$

⁸There may well be simpler, more direct proofs of Lemma 9.1 based on links to other combinatorial structures. As an indication of this we note that the number of Γ_n -structures is the *Catalan number* $\frac{1}{n+1} \binom{2n}{n}$, which appears in many other contexts of combinatorial analysis. The formula for the number of Γ_n -structures may be proved by noting that these numbers satisfy the recursion relation $\alpha_n = \sum_{\nu=1}^n \alpha_{\nu-1} \alpha_{n-\nu}$.

hold. These equivalences are proved in a similar manner and we only give the details regarding (37). Add $\ln N(a)$ to the inequality on the left hand side and appeal to the identity (3), and you realize that this inequality is equivalent to the inequality $\kappa^\sigma(a^\downarrow) \leq \bar{N}(a) + N(a)R$, hence also, as claimed, to the inequality $\phi^\sigma(a^\downarrow) \leq \bar{N}(a)$.

Now assume that the conditions stated in the lemma hold for a function ϕ on Λ . By monotonicity of ϕ , $P \in \mathcal{P}$. Then, by the assumptions (13) and (14), we see from (36) and (37) that the conditions of the Kuhn-Tucker criterion, Proposition 1.2, are fulfilled. Therefore P is the universal predictor and hence $\phi = \tilde{\kappa}^*$.

Necessity of the conditions of the lemma follow in a similar way from necessity of the Kuhn-Tucker conditions. ■

Proof of Lemma 5.1: We may assume that $a \in \Lambda \setminus \sigma(\Lambda)$. Let $P^* = \sum_{c \in \Lambda} w_c U_c$ be the barycentric decomposition of P^* . Then, for every c with $a \leq c < \bar{a}$, $w_c = 0$, hence $P^*(a) = P^*(\bar{a})$. If $b > \bar{a}$, $P^*(b) < P^*(\bar{a}) = P^*(a)$ as $w_{\bar{a}} > 0$. The stated properties follow. ■

Proof of Proposition 5.2: Let a be a minimal node and put $b = \bar{a}$. Then $\tilde{\kappa}^*(a) = \tilde{\kappa}^*(b)$. By monotonicity of $\tilde{\kappa}^*$, $\tilde{\kappa}^*(b)$ is bounded below by the average $\frac{1}{N(b)}(\tilde{\kappa}^*)^\sigma(b^\downarrow)$ thus, by (13), $\tilde{\kappa}^*(b) \geq \ln N(b)$. Now,

$$0 = \bar{N}(a) \geq (\tilde{\kappa}^*)^\sigma(a^\downarrow) = \tilde{\kappa}^*(a) = \tilde{\kappa}^*(b) \geq \ln N(b)$$

and $N(b) = 1$, hence $b = a$ follows. We conclude that a is active. Thus minimal nodes are indeed active. We leave the proof of the remaining parts of the proposition to the reader, referring to the fact just established and to Proposition 5.1. ■

Proof of Proposition 5.3: If $\bar{a} \notin \sigma(\Lambda)$ then $\bar{a} > a$ and $\tilde{\kappa}^{*\sigma}(a) = \tilde{\kappa}^{*\sigma}(\bar{a}) = \bar{N}(\bar{a}) > \bar{N}(a)$ follows. ■

Proof of Proposition 7.1: If a is minimal, $B = \emptyset$ and the definition gives $[a, \emptyset] = 0$. If a is not minimal, put $M = \sum_{t \in T_B(a)} N(t)$ and note that $N(a) > M \geq 1$, hence the positivity of $[a, B]$ follows from the manipulations

$$\begin{aligned} & |S_B(a)|[a, B] \\ &= N(a) \ln N(a) - \sum_{t \in T_B(a)} N(t) \ln \frac{N(t)}{M} - M \ln M \\ &\geq N(a) \ln N(a) - M \ln M > 0. \end{aligned}$$

Proof of Lemma 7.1: In view of the equality

$$|S_B(a)| = |S_{B^+}(a)| + |S_{B^-}(b)|,$$

each of the three identities can be derived from any of the other two. It therefore suffices to verify (18). For this, we exploit the equality above and the fact that $T_{B^+}(a)$ is the disjoint union of $\{b\}$ and the proper set-difference $T_B(a) \setminus T_{B^-}(b)$.

Appealing also to the definition of brackets, we find that

$$\begin{aligned} & |S_B(a)| \left([b, B^-] - [a, B] \right) \\ &= |S_B(a)| [b, B^-] - \bar{N}(a) + \bar{N}^\sigma(T_B(a)) \\ &= \left(|S_{B^+}(a)| + |S_{B^-}(b)| \right) [b, B^-] \\ &\quad - \bar{N}(a) + \bar{N}^\sigma(T_{B^+}(a)) - \bar{N}(b) + \bar{N}^\sigma(T_{B^-}(b)) \\ &= |S_{B^+}(a)| \left([b, B^-] - [a, B^+] \right), \end{aligned}$$

thus (18) holds. ■

Proof of Proposition 7.2: (i): If b is minimal, a cannot be minimal and the result follows from Proposition 7.1. Assume then that b is not minimal and denote by B any blocking set for b with maximal bracket. The set $B_0 = (B^* \setminus \{b\}) \cup B$ is a proper subset of B^* which is blocking for a . Then $[a]_{\max} \geq [a, B_0]$ with sharp inequality if B^* is set-theoretically minimal. By (19) applied to the set B_0 it then follows that $[a, B^*] \geq [b, B]$, i.e. $[a]_{\max} \geq [b]_{\max}$, with sharp inequality if B^* is set-theoretically minimal.

(ii): This follows by applying (20) with $B = B^*$. ■

Proof of Proposition 7.3: Let B^* be a set-theoretically maximal blocking set for a with maximal bracket. Let \tilde{B}^* be any blocking set for a with maximal bracket. We shall prove that $\tilde{B}^* \subseteq B^*$. Assume the contrary. Then there exists $b \in \tilde{B}^* \setminus B^*$, hence there also exists $b' \in \tilde{T}^* \setminus B^*$ where \tilde{T}^* denotes the ceiling of \tilde{B}^* in a . By monotonicity, $[a, \tilde{B}^*] \geq [b', B^*]$. And, by boundedness, $[b', B^*] > [a, B^*]$. The two inequalities show that $[a, \tilde{B}^*] > [a, B^*]$ which is a contradiction as $[a, B^*] = [a, \tilde{B}^*] = [a]_{\max}$. We conclude, as desired, that $\tilde{B}^* \subseteq B^*$. The reverse inclusion is proved in a similar way when also \tilde{B}^* is set-theoretically maximal. As any two set-theoretically maximal blocking sets for a with maximal brackets are equal, the largest such set, denoted $B^*(a)$, is well defined.

The facts needed to establish the results pertaining to minimal blocking sets are proved in a similar way. Details are left to the reader. ■

Proof of Theorem 7.1: Denote by ϕ the function on Λ defined by $\phi(a) = [pr(a)]_{\max}$.

We shall verify the conditions of Proposition 5.1.

First, to prove monotonicity of ϕ , consider any path from a maximal node to a minimal node. Let $t_0 > t_1 > \dots > t_k$ be the nodes in $\bar{\sigma}(\Lambda)$ on the path (thus t_k is a minimal node of Λ). Then, by monotonicity, cf. Proposition 7.2, $\phi(t_0) \geq \phi(t_1) \geq \dots \geq \phi(t_k)$ and, by the definition of ϕ , $\phi(a) = \phi(t_i)$ for nodes on the path with $t_{i-1} < a \leq t_i$ (here, $0 \leq i < k$). This proves monotonicity along any path connecting a maximal node with a minimal node. Clearly then, ϕ is monotone on all of Λ . The argument above also shows that all nodes b with $t_i > b > t_{i+1}$ are inactive, thus

$$\sigma(\Lambda) \subseteq \bar{\sigma}(\Lambda). \quad (38)$$

Next, we consider a node $a \in \bar{\sigma}(\Lambda)$, say $a \in T_i^*$, and show that (13) holds. Put $U_j = T_j^* \cap a^\downarrow$ and $V_j = S_j^* \cap a^\downarrow$ for $j \geq i$. Let k be the largest integer such that $U_j \neq \emptyset$. Then (13) for

the node a follows from the string of equalities:

$$\begin{aligned}\phi^\sigma(a^\downarrow) &= \sum_{j=i}^k \sum_{b \in V_j} \phi(b) = \sum_{j=i}^k \sum_{t \in U_j} |S^*(t)| \phi(t) \\ &= \sum_{j=i}^k \sum_{t \in U_j} (\overline{N}(t) - \overline{N}^\sigma(T^*(t))) \\ &= \sum_{j=i}^k (\overline{N}^\sigma(U_j) - \overline{N}^\sigma(U_{j+1})) = \overline{N}(a).\end{aligned}$$

By (38), the validity of (13) for all $a \in \sigma(\Lambda)$ follows.

Finally, consider a node $b \in \Lambda \setminus \overline{\sigma}(\Lambda)$. To finish the proof, we need only establish the inequality (14) for b . In fact, we shall show that the sharp inequality $\phi^\sigma(b^\downarrow) < \overline{N}(b)$ holds. To this end, put $a = pr(b)$ and $B = B^*(a) \cap b^\downarrow$ and use results already established and the boundedness property of Proposition 7.2, to find that

$$\begin{aligned}\phi^\sigma(b^\downarrow) &= |S_B(b)| \phi(b) + \sum_{t \in T_B(b)} \phi^\sigma(t^\downarrow) \\ &= |S_B(b)| [a]_{\max} + \sum_{t \in T_B(b)} \overline{N}(t) \\ &< |S_B(b)| [b, B] + \overline{N}^\sigma(T_B(b)) = \overline{N}(b).\end{aligned}$$

We have now seen that $\phi = \tilde{\kappa}^*$, hence (22) holds. As the spectrum consists of the points of increase of $\tilde{\kappa}^*$, (23) follows. ■

Proof of Proposition 8.1: We shall actually prove a formally stronger result, viz. that, for $b \in S^*(a) \setminus \{a\}$,

$$B^*(a) \wedge b \subseteq B_*(b). \quad (39)$$

Assume, for the purpose of an indirect proof, that this is not the case. Then, for some $b \in S^*(a) \setminus \{a\}$, there exists $t \in (T^*(a) \wedge b) \setminus B_*(b)$. We find that

$$\begin{aligned}[a]_{\max} &\geq [t]_{\max} \geq [t, B_*(b)] \geq [b, B_*(b)] \\ &= [b]_{\max} \geq [b, B^*(a)] > [a]_{\max}.\end{aligned}$$

Indeed, the first inequality follows by monotonicity as $t \in T^*(a)$, the second follows as $B_*(b)$ is blocking for t , the third follows by boundedness as $t \in b^\downarrow \setminus B_*(b)$, the equality is trivial, the next inequality follows as $B^*(a)$ is blocking for b and the last one follows by boundedness as $b \in S^*(a) \setminus \{a\}$. From the resulting contradiction we conclude that (39) holds. ■

Proof of Proposition 8.2: Let B be the largest blocking set for a with the monotonicity property. As $B^*(a)$ is a blocking set for a and has the monotonicity property, $B^*(a) \subseteq B$. To prove the reverse inclusion, assume, for the purpose of an indirect proof, that this is not the case. Then there exists $t \in T_B(a) \cap S^*(a)$ and we find that

$$[a, B] \geq [t]_{\max} \geq [t, B^*(a)] > [a]_{\max} \geq [a, B].$$

This is a contradiction and we conclude that $B \subseteq B^*(a)$, hence $B = B^*(a)$ as claimed. ■

Proof of Lemma 9.1: The proof is by induction on the spectral index. To start the induction we have to prove the

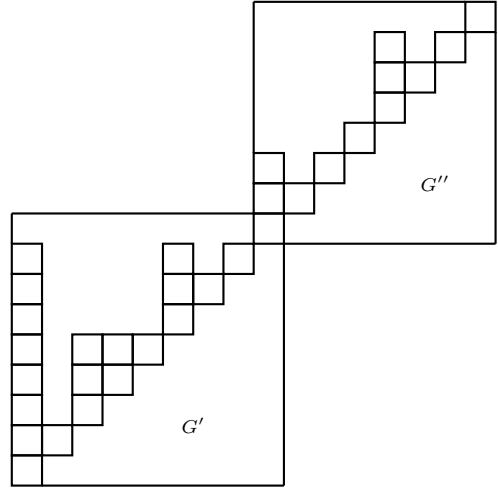


Figure 10. Illustration of last part of the proof of Lemma 9.1

implication $G \in \Gamma_n$, $\gamma(G) = 1 \Rightarrow \langle G \rangle = 2n - 1$. This is proved by induction on n . The induction start is easy. Then assume that the implication holds for indices smaller than n . Let $G \in \Gamma_n$ satisfy $\gamma(G) = 1$, i.e. $(0, n) \in G$. Consider

$$G^* = G \setminus \{(n, n+1)\} \setminus \{(\nu, n) \mid 0 \leq \nu \leq n-2\}.$$

Then $G^* \in \Gamma_{n-1}$ and $\gamma(G^*) = 1$. Thus $\langle G^* \rangle = 2n - 3$ by the induction hypothesis.

In order to compute $\langle G \rangle$ and $\langle G^* \rangle$, first remark that for a point (ν, μ) with $\mu \leq n - 2$, the equivalence $(\nu, \mu) \in G \Leftrightarrow (\nu, \mu) \in G^*$ holds and the point is a test site for G if and only if it is a test site for G^* . It remains to consider points (ν, μ) with $\mu = n$ or $\mu = n - 1$. Let $\{\nu < n - 1 \mid t_\nu = n\} = \{r_1, \dots, r_k\}$ with $r_1 < \dots < r_k$. Then $k \geq 1$ and $r_1 = 0$. Likewise, let $\{\nu < n - 2 \mid t_\nu = n - 1\} = \{s_1, \dots, s_l\}$ with $s_1 < \dots < s_l$. Here, $l = 0$ may happen corresponding to the case with no sites of the form requested. Note that by the Γ -structure of G , $s_1 > r_k$. All $k + 1$ sites $(\nu, \mu) \in G$ with $\mu = n$ are test sites for G , whereas G^* only has one site with $\mu = n$ and this site $((n - 1, n))$ is not a test site. Among the $k + l + 1$ sites (ν, μ) with $\mu = n - 1$ in G as well as in G^* , there are $1 + l + 1$ test sites in G (the sites $(r_k, n - 1), (s_1, n - 1), \dots, (s_l, n - 1)$ and $(n - 2, n - 1)$), whereas all these sites are test sites for G^* . It follows that there are $((k + 1) + (l + 2)) - (0 + (k + l + 1)) = 2$ more test sites in G than in G^* , hence $\langle G \rangle = 2n - 1$ as desired.

We now go back to the main induction proof and assume that the claimed result holds for all Γ -structures with a spectral index less than some fixed number $\gamma \geq 2$. Consider a Γ -diagram $G = G(t_0, \dots, t_n)$ with $\gamma(G) = \gamma$. Note that G is the direct sum of $G' = G(t_0, \dots, t_{t_0}, t_0 + 1)$ and the restriction G'' of G to $\{t_0, \dots, n\}$ as indicated in Figure 10. As $\gamma(G') = 1$, $\langle G' \rangle = 2t_0 - 1$ by the first part of the proof and by the induction hypothesis, $\langle G'' \rangle = 2(n - t_0) - (\gamma - 1)$. Clearly, $\langle G \rangle = \langle G' \rangle + \langle G'' \rangle$. Therefore, we find that $\langle G \rangle = 2n - \gamma(G)$. This is the desired result and the induction is complete. ■

ACKNOWLEDGMENTS

Boris Ryabko introduced the second named author to the general problems involved and several useful discussions are

acknowledged. The complex of problems have also been discussed with Peter Harremoës and a number of related results, but for other models, have been developed, though not yet published. The recommendations of the referees resulted in substantial changes of the presentation intended to improve readability and in the observation of an interesting connection, presently not fully understood, to the problem of isotone regression (cf. end of Section II). Finally, we thank Jan Caesar for technical assistance.

REFERENCES

- [1] B. Fitingof, "Coding in the case of unknown and changing message statistics," *Probl. Inform. Transmission*, vol. 2, no. 2, pp. 3–11, 1966, in Russian.
- [2] L. D. Davisson, "Universal noiseless coding," *IEEE Trans. Inform. Theory*, vol. 19, pp. 783–795, 1973.
- [3] N. Merhav and M. Feder, "Universal prediction," *IEEE Trans. Inform. Theory*, vol. 44, no. 6, pp. 2124–2147, Oct. 1998.
- [4] B. Y. Ryabko, "Encoding of a source with unknown but ordered probabilities," *Problems of Information Transmission*, vol. 15, pp. 71–77, 1979, russian original in *Probl. Peredachi Inf.*, 1978.
- [5] F. Topsøe, "Exact Prediction and Universal Coding for Trees," in *Proceedings ISIT 2001*. Washington: IEEE, June 2001.
- [6] B. Ryabko and F. Topsøe, "Universal coding for sources with partially ordered probabilities," in *Proceedings IEEE International Symposium on Information Theory*. Washington: IEEE, June 2001.
- [7] B. Y. Ryabko, "Comments on "a source matching approach to finding minimax codes";" *IEEE Trans. Inform. Theory*, vol. 27, pp. 780–781, 1981, including also the ensuing Editor's Note.
- [8] P. Pardalos and G. Xue, "Algorithms for a Class of Isotonic Regression Problems," *Algorithmica*, vol. 23, pp. 211–222, 1999.