Discrete Applied Mathematics 12 (1985) 293–302
North-Holland

UNIVERSAL RETRIEVAL TREES

R.E. KRICHEVSKY and B.Ya. RYABKO

Inst. of Math. of the Siberian Div. of the Academy of Science of the USSR, Novosibirsk, USSR 630090

Received 16 February 1983
Revised 25 March 1985

A retrieval tree identifies members of a given set. The redundancy of a tree is defined to be the difference between the maximum length of a path in the tree and the binary logarithm of the cardinality of the set. Given a family $\{A_\theta\}$ of sets, a retrieval tree is developed whose maximum redundancy over the family is minimum. The tree is used to make a biological key. The probabilistic variant of the problem is discussed.

1. Introduction

Retrieval trees are widely used to identify objects. The root of a tree is labelled with an attribute. If an object possesses that attribute, go to the left son of the root; if it does not, go to the right one. The sons are labelled, too. Moving in this way from one node to another, one finally reaches a leaf which is labelled with the name of the object.

An example of a retrieval tree is depicted in Fig. 1. This is a tree used to identify the ants of subgenera S.Formica, taken from [6]. The root is labelled with the attribute "to have a black head". The left son of the root is a leaf, labelled with the name of the species "F.uralenzis". The right son of the root is labelled with the

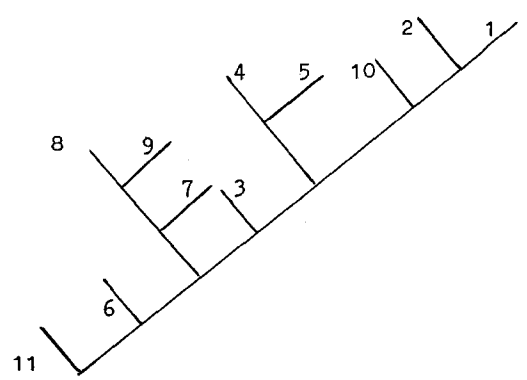

Fig. 1. Ant identifying key of subgenera S. Formica from [6].

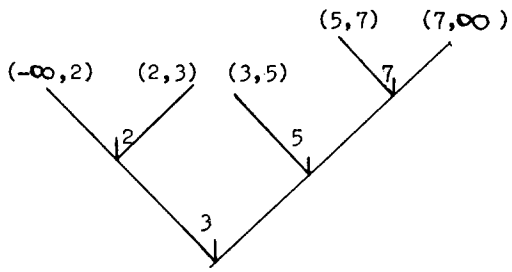

Fig. 2. A retrieval tree for the set $\{2,3,5,7\}$. Go to the left, if the integer sought for is less than the label; go to the right, if it is more. Go to the middle son, if they are equal.

attribute "to have more than three couple of hairs on the bottom of the chest", and so on. Such trees are called keys.

Similar keys are utilized in medicine, mineralogy, chemistry, computer programming, etc. [2]. Computer programming trees are used to find a number from a set. Each non-terminal node of such a tree is labelled with a number and has three sons: right, middle and left ones. If the number sought for is less than the label, go to the left son; if it equals the label, go to the middle one; otherwise go to the right son. There is a retrieval tree for the set $B=\{2,3,5,7\}$ in Fig. 2. The retrieval ends either at a circle-marked or a square-marked node depending on whether or not a number to be identified belongs to B. The intervals $(-\infty,2)$, $[2,2]$, $(2,3)$, $[3,3]$, ..., $[7,7]$, $(7,\infty)$ can be thought of as the objects to be identified.

Let L be a retrieval tree for a set A. Then for any $a\in A$ there is in L a leaf corresponding to a. Denote by $L(a)$ the length of the path from the root to that leaf. Kraft's inequality $\sum_{a\in A} d^{-L(a)}\le 1$, where d is the maximal number of sons of a node, is met [2]. We are dealing with the most often arising binary case in Sections 1–4, so all logarithms are taken there to the base 2. However, nonbinary generalization is straightforward.

The problem to find a good retrieval tree can be set in either probabilistic or combinatorial way. The probabilistic way is as follows. Let there be a probability distribution $p=\{p(a), a\in A\}$, on A. Then the average retrieval time for L is

$$\bar{c}(L,p)=\sum_{a\in A} p(a)\,L(a). \tag{1}$$

Information theory is used to develop trees with minimal average retrieval time for a given distribution p, [1,2,11,17]. The minimum of $\bar{c}(L,p)$ over the set of all trees is close to Shannon entropy $H(p)=-\sum_{a\in A} p(a)\log p(a)$, or, to be more accurate,

$$H(p)\le \min_{L} \bar{c}(L,p)\le H(p)+1.$$

The difference $\bar{r}(L,p)=\bar{c}(L,p)-H(p)$ is defined to be the redundancy of a tree L on a distribution p. Huffman's algorithm [11] produces a tree with the minimal re-

dundancy. This algorithm is rather complicated. Shannon's algorithm [17] is simpler. The redundancy of the tree developed by this algorithm is not more than 1. The pathlength $L(a)$ equals $\lceil -\log p(a) \rceil$ for that tree. Either in [14] or in [1,2,10,11,13] one can find methods to construct either biological or computer programming keys, respectively.

The combinatorial setting of the problem is as follows. There is no probability distribution now and all members of A have equal rights. The maximum retrieval time $c(L,A)$

$$c(L,A) = \max\{L(a), a \in A\} \tag{1'}$$

is to be used instead of average retrieval time. The minimum $c(L,A)$ over all vectors $(L(a))$, $a \in A$ for which Kraft's inequality is met, equals (to within an additive unit, see [2]) Hartley's entropy of A, i.e. it equals $\log |A|$. The number $r(L,A)$

$$r(L,A) = c(L,a) - \log |A| \tag{2'}$$

is the redundancy of a tree L on the set A. The problem of constructing a tree with minimum redundancy has an obvious solution: $L(a)$ equals either $\lceil \log |A| \rceil$ or $\lfloor \log |A| \rfloor$, $a \in A$.

So there are methods to construct both probabilistic and combinatorial retrieval trees with minimum redundancy. Those trees are intended for either a given probability distribution or a given set A. Very often, however, there is not such a specific distribution or not such a single set. It is necessary to develop a tree which is good enough for either a family of distributions or a family of subsets simultaneously. This is termed the problem of making a *universal tree.* We have two settings of the problem again. In the probabilistic setting there is a family $\Lambda = \{p_\theta\}$ of probability distributions on a set A. In the combinatorial setting there is a family $\Lambda = \{A_\theta\}$, $\bigcup A_\theta = A$, of subsets of A. For a retrieval tree L its redundancy on a family Λ is defined either as $\bar{R}(L,\Lambda) = \sup\{\bar{r}(L, p_\theta), p_\theta \in \Lambda\}$ (probabilistic case), or as $R(L,\Lambda) = \sup\{r(L, A_\theta), A_\theta \in \Lambda\}$ (combinatorial case). The (minimax) redundancy of a family is minimum redundancy on it of all retrieval trees. So let $R(\Lambda) = \inf R(L,\Lambda)$ be the redundancy of a family Λ of subsets and $\bar{R}(\Lambda) = \inf \bar{R}(L,\Lambda)$ be the redundancy of a family Λ of distributions. The goal is to make an optimum universal either combinatorial or probabilistic tree, i.e. either a tree L_p for which $\bar{R}(L_p,\Lambda) = \bar{R}(\Lambda)$, or a tree L_c for which $R(L_c,\Lambda) = R(\Lambda)$.

Both settings are of interest. For instance, the probability to meet a biological object is never known exactly. It depends on the year, the month, the place, etc. A family $\{p_\theta\}$ of probability distributions can be considered known for a relatively small and well explored region. So, if an identifying key is to be developed for the species of such a region, then the problem of constructing a universal tree arises in the probabilistic setting. But such a family cannot be known if an identifying key is to be developed for a big region, say, a continent or the world. The combinatorial setting of the problem fits better there. A family $\{A_\theta\}$ of subsets appears in a natural way: subsets A_θ may be the species of deserts, forests, etc. The key is to be

satisfactory for all subsets at once, that is why maximum redundancy $r(L, A_\theta)$ must be minimum.

Either probabilistic or combinatorial retrieval trees optimum to within an additive constant are developed in Sections 2 and 3. To construct a probabilistic tree, it is necessary to find the minimum of a convex function first. Combinatorial trees are constructed explicitly. An example of a retrieval biological tree is in Section 4. The results are generalized to computer programming trees in Section 5.

An instance of the problem had already been addressed in [12]. The set Λ of probability distributions consisted of all vectors with nonincreasing coordinates there.

Probabilistic universal trees are discussed by T. Fischer in [7]. But the target there is to make for a family Λ a tree L for which maximum on Λ average retrieval time $\bar{c}(L, p_\theta)$ is minimum, whereas our target is to make a tree for which maximum on Λ redundancy is minimum. If all distributions p_θ of the family Λ have nearly equal entropies $H(p_\theta)$, then ours and T. Fischer's approaches give nearly the same results. On the other hand, if those entropies differ significantly, then T. Fischer's retrieval time is significantly more than ours for distributions with small entropies. Thus, in that case our approach can be considered more suitable.

We presume that for any partition of A into two disjoint subsets B and $\bar{B}$ there is an attribute which equals 1 on B and 0 on $\bar{B}$, i.e. we are free to choose tests as we please. What should one do if it is not so, i.e. if some tests are not available? Unfortunately, the problem grows then NP-hard, as it is proved in [4] even for non-universal trees. Still, even in that more complicated, although more real situation, our result may be of interest. Having no possibility to develop exactly an optimal tree (NP-hardness!) a biologist or another expert may use our trees as a target to be approached when making keys. Our experience is that experts on ants were nearly always able to choose the tests in such a way that the key was very close to the pattern precomputed according to Sections 2 and 3.

2. Probabilistic universal trees

We reduce the problem of making a universal retrieval tree to the well-known information theory problem of finding the information rate of a channel. Computational methods to find such a rate are developed in [3].

Let $\Lambda = \{p_\theta\}$ be a family of probability distributions on a set A. For simplicity we restrict ourselves to finite families Λ only. Infinite families are discussed in [8]. Let $\Phi(\Lambda)$ be the set of all probability distributions on the family Λ. If $\varphi \in \Phi(\Lambda)$, then $\varphi(p_\theta)$ means the probability of a distribution p_θ. The mutual information $I(\varphi, \Lambda)$ and the information rate $c(\Lambda)$ are defined as follows:

$$I(\varphi, \Lambda) = \sum_{p_\theta \in \Lambda} \varphi(p_\theta) \sum_{a \in A} p(a) \log \left(p_\theta(a) \Big/ \sum_{p_\theta \in \Lambda} \varphi(p_\theta) p_\theta(a) \right),$$

$$c(\Lambda) = \sup \{ I(\varphi, \Lambda), \varphi \in \Phi(\Lambda) \}.$$

As is proved in [8], for the family Λ there is a distribution $\nu \in \Phi(\Lambda)$ such that

$$c(\Lambda) = I(\nu, \Lambda). \tag{3}$$

Theorem 1. *Let A be a finite set and let $\Lambda = \{p_\theta\}$ be a family of probability distributions on A. Then the maximum redundancy of any tree retrieval for A is no less than the information rate $c(\Lambda)$. On the other hand, there is a tree whose maximum redundancy is no more than $c(\Lambda)+1$:*

$$c(\Lambda) \le \bar{R}(\Lambda) \le c(\Lambda) + 1.$$

To make such a tree, take the distribution ν of (3) and define on A a distribution π by:

$$\pi(a) = \sum_{p_\theta \in \Lambda} \nu(p_\theta) p_\theta(a), \quad a \in A. \tag{4}$$

Develop for the distribution π the Shannon tree L, for which $L(a) = \lceil -\log \pi(a) \rceil$. For that tree the claim of the theorem is met.

The theorem belongs to the information theory. Its proof may be found in [5, 9, 15, 16].

3. Combinatorial universal trees

Let A be a set, and let $\Lambda = \{A_\theta\}$ be a family of subsets of A. Let, for $a \in A$,

$$z(a) = \{A_\theta, a \in A_\theta\}, \tag{5}$$

$$u(a) = \min\{|A_\theta|, A_\theta \in z(a)\}, \tag{6}$$

$$\mathscr{R}(\Lambda) = \sum_{a \in A} u(a)^{-1}. \tag{7}$$

Theorem 2. *If A is a set, $\Lambda = \{A_\theta\}$ is a family of it subsets, $\bigcup_{A_\theta \in \Lambda} A_\theta = A$, then the redundancy of the best for Λ tree equals* $\log \mathscr{R}(\Lambda)$ *to within an additive unit:*

$$\log \mathscr{R}(\Lambda) \le R(\Lambda) \le \log \mathscr{R}(\Lambda) + 1.$$

To make a tree for which the claim of the theorem is met, define on A a distribution γ,

$$\gamma(a) = (u(a)\mathscr{R}(\Lambda))^{-1} \tag{8}$$

and develop for γ the Shannon tree L_γ such that $L_\gamma(a) = \lceil -\log \gamma(a) \rceil$. That tree is sought for.

Proof. (i) *Lower bound.* Suppose that the lower bound is not true, i.e. there is a retrieval for A tree L such that

$$R(L, \Lambda) < \log \mathscr{R}(\Lambda).$$

From this inequality and the definitions of $R(L, \Lambda)$ and (2') we obtain

$$\max\{L(a) - \log |A_\theta|, a \in A_\theta\} < \log \mathscr{R}(\Lambda), \quad A_\theta \in \Lambda.$$

Hence, for any $a \in A$ and any $A_\theta \in \Lambda$

$$L(a) - \log |A_\theta| < \log \mathscr{R}(\Lambda).$$

This inequality and definitions (5), (6) yield

$$L(a) - \log u(a) < \log \mathscr{R}(\Lambda). \tag{9}$$

Next bound the difference $\bar{c}(L, \gamma) - H(\gamma)$ where γ is defined by (8):

$$\begin{aligned}\bar{c}(L, \gamma) - H(\gamma) &= \sum_{a \in A} L(a)\gamma(a) - H(\gamma) \\ &= \sum_{a \in A} \gamma(a)(L(a) + \log \gamma(a)) \\ &= \sum_{a \in A} \gamma(a)(L(a) - \log u(a) - \log \mathscr{R}(\Lambda) < 0.\end{aligned}$$

The first inequality follows from (1), the second one from the definition of Shannon's entropy, the third one from (8), and the last inequality from (9). Thus, we get

$$\bar{c}(L, \gamma) - H(\gamma) < 0, \tag{10}$$

a contradiction because $\bar{c}(L, \gamma) \geq H(\gamma)$, for any distribution γ.

(ii) *Upper bound.* Choose $A_\theta \in \Lambda$ and $a \in A_\theta$. We have

$$\begin{aligned}L(a) - \log |A_\theta| &\leq \log \mathscr{R}(\Lambda) + 1 + \log u(a) - \log |A_\theta| \\ &\leq \log \mathscr{R}(\Lambda) + 1.\end{aligned}$$

The first inequality follows from (7), (8), the second one from (5), (6). This chain of inequalities together with (1'), (2') proves the upper bound. □

4. Making a biological key

We use Theorem 2 to construct an identifying key for a subgenera of ants.

Biologists often conduct examination of the fauna of various regions. Usually they have to collect as many as two or even three thousands of ants and identify them with the aid of keys. It can take an expert entomologist up to 20 days to do the job. So it is worth while to try to make a more convenient key.

A key used now to identify the ants of subgenera Serviformica is shown in Fig. 1. The key is taken from the book [6]. It is used throughout the territory of the USSR. Try to improve it via Theorem 2.

The territory of the USSR falls into five vast zones: tundra, forest, partially wooden steppe, steppe and desert. Table 1 shows which species of subgenera Serviformica inhabits which zone. We want to make one key intended for use throughout

Table 1. Distribution of species of subgenera Serviformica in the zones of the USSR (+: species is present, −: species is absent).

No.	Species	Tundra	Forest	Part. wooden steppe	Steppe	Desert	Number of attributes used to identify the species		
							Key of Fig. 3	Key of Fig. 1	
1	F. fusca	−	+	+	−	−	4	7	1/7
2	F. lemani	−	+	−	−	−	3	7	1/9
3	F. picea	−	+	+	+	−	4	4	1/6
4	F. gagatoides	+	+	−	−	−	3	6	1
5	F. kozlovi	−	+	−	−	−	4	6	1/9
6	F. cinerea	−	+	+	+	−	4	2	1/6
7	F. subpilosa	−	−	−	+	+	4	4	1/3
8	F. cunicularia	−	+	+	+	+	3	5	1/3
9	F. rufibarbis	−	+	+	+	+	3	5	1/3
10	F. gagates	−	−	+	−	−	4	6	1/7
11	F. uralensis	−	+	+	+	−	3	1	1/6

the USSR, just as it is done in [6]. One can hardly expect to know something about the probability distribution of species in such immense zones with extremely changeable climate. The combinatorial approach looks here more attractive than the probabilistic one.

The set A is the set of all species here, the set A_1 is the set of all inhabitants of tundra, ..., A_5 is the set of all inhabitants of desert. First find for each species the corresponding number $u(a)$. Those numbers are displayed in Table 1. Then find $\mathscr{R}(A)$, $\gamma(a)$ and make Shannon's tree for the distribution γ. We cannot choose attributes as we please. That is why we have here obtained not an optimum, but a nearly optimum key. It is shown in Fig. 3. We do not list the characters at each node because the descriptions of tests in [6] are rather lengthy and are often accompanied with pictures.

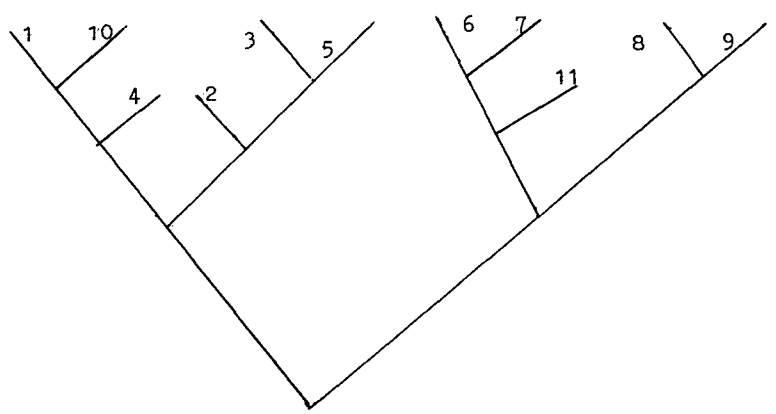

Fig. 3. Ant identifying key of subgenera S. Formica made with the aid of Theorem 2.

Table 2. Comparison of the key from [6] and that made according to Theorem 1 on real data. The number of specimens really gathered is shown. In the two bottom lines the retrieval time per specimen is shown when either the key of Fig. 3 or the key of Fig. 1 is used.

No.	Species		Steppe	P.w. steppe	Desert	Forest
1	F. fusca		60	560	–	500
2	F. lemani		–	–	–	50
3	F. picea		–	20	–	250
4	F. gagatoides			–	–	20
5	F. kozlovi		–	–	–	20
6	F. cinerea		–	2	–	15
7	F. subpilosa		–	–	450	–
8	F. cunicularia		400	160	14	–
9	F. rufibarbis		280	90	360	–
10	F. gagates		–	–	–	–
11	F. uralenis		–	–	–	–
Retrieval time per		Fig. 3	3.08	3.72	3.55	3.92
specimen with key		Fig. 1	5.16	6.31	4.45	5.85

Compare our key (Fig. 3) with that of the book [6] (Fig. 1) on real data. A Siberian entomologist J.I. Resnikova explored the ants of various zones of the USSR. The members of different species of serviformica ants she gathered are displayed in Table 2. Identifying time per specimen is shown in either the last but one or the last line of the table depending on whether our key or the key from [6] is used. It can be seen that our key is better in all zones for those data.

5. Generalization

We have, up to this moment, dealt with binary retrieval trees. The lower bounds for retrieval time were binary Shannon or Hartley entropies. But sometimes the lower bound of retrieval time even for nonbinary trees is binary entropy. It is the case of computer programming trees of Section 1. More generally, let A be a finite set, $\mathscr{L}(A)$ be a set of retrieval trees, α and β are some constants. We say that $\mathscr{L}(A)$ meets the IT-condition with constants α and β if for any probability distribution p on A there is a tree $L_p \in \mathscr{L}(A)$ such that

$$L_p(a) \leq -\log_d p(a) + \alpha, \quad a \in A$$

and for any $L' \in \mathscr{L}(A)$

$$c(L', p) \geq H(p) - \beta.$$

The constants α and β do not depend on p, but can depend on A. Logarithms are taken to a base $d \geq 2$, one and the same in those inequalities and in the sequel. For binary retrieval trees $\alpha = 1$, $\beta = 0$. Computer programming trees meet that condition as well ($\alpha = 2$, $\beta = \log\log|A| + O(1)$), see [2]. Although any node of the tree has

three sons, one of them is a leaf (Fig. 2). This is why one comparison gives not much more than one bit of information. There are other problems for which the condition holds, see [2].

Theorems 1 and 2 are generalized to the case of retrieval trees under the IT-condition. The definitions are slightly modified. Let $\bar{c}(p) = \inf\{\bar{c}(L, p),\ L \in \mathscr{L}(A)\}$, p is a probability distribution on A,

$$c(B) = \inf\{c(L, B), L \in \mathscr{L}(A)\}, \quad B \subset A,$$
$$\bar{r}_{\text{it}}(L, p) = \bar{c}(L, p) - \bar{c}(p),$$
$$r_{\text{it}}(L, B) = c(L, B) - c(B).$$

$L \in \mathscr{L}(A)$, Λ is a family of probability distribution on A or a family of subsets of A.

Let

$$\bar{R}_{\text{it}}(L, \Lambda) = \sup\{\bar{r}_{\text{it}}(L, p), p \in \Lambda\}, \qquad \bar{R}_{\text{it}}(\Lambda) = \inf\{\bar{R}_{\text{it}}(L, \Lambda), L \in \mathscr{L}(A)\},$$
$$R_{\text{it}}(L, \Lambda) = \sup\{r_{\text{it}}(L, B), B \in \Lambda\}, \qquad R_{\text{it}}(\Lambda) = \inf\{R_{\text{it}}(L, \Lambda), L \in \mathscr{L}(A)\},$$

Theorem 1′. *Let $\Lambda = \{p\}$ be a family of probability distributions on a set A, let $\mathscr{L}(A)$ be a set of retrieval trees and assume the IT-condition is met. Then there is a tree L_p for which*

$$c(\Lambda) - (\alpha + \beta) \le \bar{R}_{\text{it}}(\Lambda) \le \bar{R}_{\text{it}}(L_p, \Lambda) \le c(\Lambda) + (\alpha + \beta).$$

Theorem 2′. *Let $\Lambda = \{B\}$ be a finite family of subsets of A, $\bigcup_{B \in \Lambda} B = A$, let $\mathscr{L}(A)$ be a set of retrieval trees and assume the IT-condition is met. Then there is a tree L_c for which*

$$\log \mathscr{R}(\Lambda) - (\alpha + \beta) \le R_{\text{it}}(\Lambda) \le R_{\text{it}}(L_c, \Lambda) \le \log \mathscr{R}(\Lambda) + (\alpha + \beta).$$

Proofs. First define on a set A the probability distributions π and γ the same way as it has been done in the proofs of Theorems 1 and 2. As follows from the IT-condition, there is a tree L_p such that for any $a \in A$

$$L_p(a) \le -\log \pi(a) + \alpha$$

and there is a tree L_c such that for any $a \in A$

$$L_c(a) \le -\log \gamma(a) + \alpha$$

where α is the constant in the IT-condition. The tree L_p meets the claim of Theorem 1′, and the tree L_c meets the claim of Theorem 2′. □

Acknowledgements

The authors are very grateful to Mrs. Resnikova for placing at their disposal the data and other information concerning the ants and for her aid in key making. We

wish to thank Mrs. Korneev for her assistance in preparing the presentation and both anonymous referees whose remarks helped to improve the paper.

References

[1] A.V. Aho, J.E. Hopcroft and J.D. Ullman. The Design and Analysis of Computer Algorithms (Addison-Wesley, Reading, MA 1976).
[2] R. Ahlswede and I. Wegener, Such probleme (B.G. Teubner, Stuttgart, 1979).
[3] R.E. Blahut, Computation of channel capacity and rate distortion functions, IEEE Trans. Inform. Theory 18(4) (1972) 460–473.
[4] D. Comer and R. Sethi, The complexity of tree index construction, JACM 24(3) (1977) 428–440.
[5] L.D. Davisson and A. Leon-Garcia, A source matching approach to finding minimax codes, IEEE Trans. Inform. Theory, 26(2) (1980) 166–174.
[6] G.M. Dluskii, Ants of Genera Formica (Nauka, Moscow, 1967) (in Russian).
[7] T. Fischer, On the weighted path length of binary search trees for unknown access probabilities, Lect. Notes in Computer Sci. 74 (Springer, Berlin, 1979) 284–291.
[8] R.G. Gallager, Information Theory and Reliable Communication (Wiley, New York, 1968).
[9] R.G. Gallager, Source coding with side information and universal coding, Unpublished manuscript.
[10] E.N. Gilbert and F. Moore, Variable length binary encodings, BSTJ 38(4) (1959) 933–967.
[11] D.E. Knuth, The Art of Computer Programming, Vol. 3 (Addison-Wesley, Reading, MA, 1973).
[12] R.E. Krichevskii, B.Ya. Ryabko and A.Yu. Haritonov, Optimal key for taxons ordered in accordance with their frequencies, Discrete Appl. Math. 3 (1981) 67–72.
[13] K. Melhorn, Nearly optimal binary search trees, Acta Informatica (1975).
[14] R.W. Payne and D.A. Preece, Identification keys and diagnostics tables: a review, J. Roy. Statist. Soc. Ser. A (general), 143(3) (1980) 253–292.
[15] B.Ya. Ryabko, Encoding of a source with unknown but ordered probabilities, Problems Inform. Transmission 15(2) (1979) 71–77.
[16] B.Ya. Ryabko, Comments on "A source matching approach to finding minimax codes", IEEE Trans. Inform. Theory, 27(6) (1981) 780–781.
[17] C. Shannon, Mathematical theory of communication, BSTJ 27(3) (1949) 379–423.