MULTIPLE CHOICE TRIES AND DISTRIBUTED HASH TABLES

Luc Devroye

School of Computer Science
McGill University
3450 University Street
Montreal H3A 2K6
Canada
luc@cs.mcgill.ca

Gábor Lugosi

ICREA and Department of Economics
Universidad Pompeu Fabra
25-27 Ramon Trias Fargas
Barcelona
Spain
lugosi@upf.es

Gahyun Park and Wojciech Szpankowski

Department of Computer Sciences
Purdue University
250 N. University Street
West Lafayette, Indiana, 47907-2066
USA
{gpark,spa}@cs.purdue.edu

April 10, 2006

Corresponding authors' address: Luc Devroye, School of Computer Science, McGill University, 3480 University Street, Montreal, Canada H3A 2K6.

The first author's research was sponsored by NSERC Grant A3456 and FQRNT Grant 90-ER-0291. The second author acknowledges support by the Spanish Ministry of Science and Technology and FEDER, grant BMF2003-03324 and by the PASCAL Network of Excellence under EC grant no. 506778. The last two authors were supported by NSF Grants CCR-0208709, CCF-0513636, and DMS-0503742, AFOSR Grant FA8655-04-1-3074, and NIH Grant R01 GM068959-01.

Abstract. In this paper we consider tries built from n strings such that each string can be chosen from a pool of k strings, each of them generated by a discrete i.i.d. source. Three cases are considered: $k = 2$, k is large but fixed, and $k \sim c \log n$. The goal in each case is to obtain tries as balanced as possible. Various parameters such as height and fill-up level are analyzed. It is shown that for two-choice tries a 50% reduction in height is achieved when compared to ordinary tries. In a greedy on-line construction when the string that minimizes the depth of insertion for every pair is inserted, the height is only reduced by 25%. In order to further reduce the height by another 25%, we design a more refined on-line algorithm. The total computation time of the algorithm is $O(n \log n)$. Furthermore, when we choose the best among $k \geq 2$ strings, then for large but fixed k the height is asymptotically equal to the typical depth in a trie. Finally, we show that further improvement can be achieved if the number of choices for each string is proportional to $\log n$. In this case highly balanced trees can be constructed by a simple greedy algorithm for which the difference between the height and the fill-up level is bounded by a constant with high probability. This, in turn, has implications for distributed hash tables, leading to a randomized ID management algorithm in peer-to-peer networks such that, with high probability, the ratio between the maximum and the minimum load of a processor is $O(1)$.

Keywords and phrases. Random tries, random data structures, probabilistic analysis of algorithms, algorithms on sequences, distributed hash tables.

CR Categories: 3.74, 5.25, 5.5.

1991 Mathematics Subject Classifications: 60D05, 68U05.

1. Introduction

A trie is a digital tree built over n strings (see Knuth (1997), Mahmoud (1992) and Szpankowski (2001) for an in-depth discussion of digital trees.) A string is stored in an external node of a trie and the path length to such a node is the shortest prefix of the string that is not a prefix of any other strings.

Tries are popular and efficient data structures that were initially developed and analyzed by Fredkin (1960) and Knuth (1973) as an efficient method for searching and sorting digital data. Recent years have seen a resurgence of interest in tries that find applications in dynamic hashing, conflict resolution algorithms, leader election algorithms, IP address lookup, Lempel-Ziv compression schemes, and others. Distributed hash tables arose recently in peer-to-peer networks in which keys are partitions across a set of processors. Tries occur naturally in the area of ID management in distributed hashing, though they were never explicitly named. One of the major problems in peer-to-peer networks is load balancing. We address this problem by redesigning old-fashioned tries into highly balanced trees that in turn produce an $O(1)$ balance (i.e., the ratio between the maximum and the minimum load) in the partition of processors in such networks. We accomplish this by adopting the "power-of-two" technique that already found many successful applications in hashing.

We consider random tries over $\mathcal{N}$, the set of positive integers, where each datum consists of an infinite string of i.i.d. symbols drawn from a fixed distribution on $\mathcal{N}$. The probability of the i-th symbol is denoted by p_i. The tries considered here are constructed from n independent strings $X_1, \ldots, X_n$. Each string determines a unique path from the root down in an infinite ∞-ary tree: the symbols have the indices of the child nodes at different levels, that is, the path for X_i starts at the root, takes the X_{i1}-st child, then the X_{i2}-st child of that node, and so forth. Let N_u be the number of strings traversing node u in this infinite tree. A string is associated with the node u on its path that is nearest to the root and has $N_u = 1$. The standard random trie for n strings consists of these n marked nodes, one per string, and their paths to the root. The marked nodes are thus the leaves of the tree.

The properties of the standard random trie are well-known (see Szpankowski, 2001): for example, if D_n is the depth of a random leaf (i.e., its path distance to the root), then

$$\frac{D_n}{\log n} \to \frac{1}{H} \quad \text{in probability}$$

as $n \to \infty$, where

$$H = \sum_i p_i \log\left(\frac{1}{p_i}\right)$$

is the entropy of the distribution. This results remains true even if $H = \infty$. The mean and variance of D_n were first obtained by Jacquet and Régnier (1986), Pittel (1985) and Szpankowski (1988). Jacquet and Régnier (1986), Pittel (1986) and Jacquet and Szpankowski (1991) proved that D_n properly normalized converges to the normal distribution.

If H_n denotes the height of the trie, i.e., the maximal distance between root and leaves, then

$$\frac{H_n}{\log n} \to \frac{2}{Q} \quad \text{in probability}$$

as $n \to \infty$, where

$$Q = \log\left(\frac{1}{\sum_i p_i^2}\right)$$

(Pittel, 1985). From Jensen's inequality and $(\max_i p_i)^2 \le \sum_i p_i^2 \le \max_i p_i$, we have

$$\frac{1}{H} \le \frac{1}{Q} \le \frac{1}{\log\left(\frac{1}{\max_{i\ge 1} p_i)}\right)} \le \frac{2}{Q},$$

so that the height is always at least twice as big as the typical depth of a node.

In some applications, it is important to reduce the height as much as possible. Attempts in this direction include PATRICIA trees (Morrison, 1968) and digital search trees (Coffman and Eve, 1970, Konheim and Newman, 1973). In PATRICIA trees, all internal nodes with one child are eliminated. In digital search trees, each internal node is associated with a string, namely the first string that visits that node (the order of $X_1, \ldots, X_n$ thus matters). In both cases, we have

$$\frac{H_n}{\log n} \to \frac{1}{\log\left(\frac{1}{\max_{i\ge 1} p_i)}\right)} \quad \text{in probability}$$

(Pittel, 1985). In other words, the height of the latter variants of tries improves over that of the random trie, but by not more than 50%. Also, both PATRICIA trees and digital search trees introduce slight inconveniences: "in order traversal" of digital search trees does not visit the nodes in sorted order, and internal edges of PATRICIA trees have cumbersome labels.

In the so called b-tries, one allows to store in an external node up to b strings (i.e., there are at most b strings sharing the same prefix). For such a b-trie,

$$\frac{H_n}{\log n} \to \frac{b+1}{\log\left(\frac{1}{\sum_i p_i^{b+1}}\right)} \text{ in probability}$$

(Pittel (1985), see also Szpankowski, 2001).

The height is not the only balance parameter. In a trie with finite fanout β (i.e., X_i takes values on $\{1, \ldots, b\}$), the fill-up level F_n, the distance from the root to the last level that has a full set of β^{F_n} nodes, is also important. Pittel (1986) found the typical value of the F_n in a trie built over n strings generated by mixing sources. For memoryless sources,

$$\frac{F_n}{\log n} \to \frac{1}{\log(1/p_{\min})} = \frac{1}{h_{-\infty}} \text{ in probability}$$

where $p_{\min} = \min_i\{p_i\}$ is the smallest probability of generating a symbol and $h_{-\infty} = \log(1/p_{\min})$ is the Rényi entropy of infinite order (Szpankowski (2001)). This was further extended by Pittel (1986), Devroye (1992), and Knessl and Szpankowski (2005) who proved that the fill-up level F_n is concentrated on two points k_n and $k_n + 1$, where for asymmetric sources k_n is an integer

$$\frac{1}{\log(1/p_{\min})}\left(\log n - \log\log\log n\right) + O(1)$$

while for symmetric sources (i.e., sources with $p_1 = p_2 = 1/2$) k_n is

$$\log_2 n - \log_2\log_2 n + O(1).$$

Observe that in the symmetric case we have $\log\log n$ instead of $\log\log\log n$.

To understand why balanced tries are relevant to load balancing in peer-to-peer networks, consider the following scenario discussed in Malkhi et al. (2002) and Adler et al. (2003). In peer-to-peer networks, each of n processor is given a key which is mapped into the interval $[0, 1]$. Thus, processors can be considered as infinite binary strings. These strings are organized as in a binary trie. When the keys are

uniform on $[0,1]$, then the bits are i.i.d. with $p_1 = p_2 = 1/2$, as in standard binary trie. The trie is used to locate peers with close keys, and the table of keys is also called a distributed hash table.

There are a number of performance measures for distributed hash tables. Among them the search time (the number of queries required to locate a requested item) and load balancing (how load is balanced between the processors) are the most important. Since every processor is assigned to a subinterval in $[0,1]$ (namely the one controlled or "owned" by its key, which could, but does not have to be at its center), load balancing can be measured by the ratio B_n of the largest to the smallest assigned subinterval. The goal is to design hash tables with bounded balance ratio.

We address load balancing and search time issues in the context of the associated tries. In order to construct a well balanced trie needed for an efficient distributed hash table, we design a new trie in which every processor has $O(\log n)$ keys to be tried before inserting into the trie the key that has the smallest depth of insertion. We will argue that in a problem of ID management in peer-to-peer networks the relevant quantity is the ratio

$$B_n = 2^{H_n - F_n},$$

where H_n and F_n are the height and the fill-up level in the associated trie, respectively. In view of this a well balanced network requires to design a well-balanced trie with $H_n - F_n = O(1)$. As the first step we propose the so called two-choice trie in which we deal with pairs of strings. A simple argument shows that if, in an on-line fashion, for every pair of strings one inserts in the trie the one that has smaller depth of insertion, then for symmetric sources

$$\frac{H_n}{\log n} \to \frac{3}{2Q} \text{ in probability,}$$

resulting in a 25% reduction in height compared to standard tries. However, even with such a reduction in height, one has $H_n - F_n = \Theta(\log n)$ in probability.

To reduce the height further, we design a refined version of the power-of-two tries in which one selects n strings resulting in a height that is close to the smallest possible. Call this height H_n^*. We design an on-line algorithm with total computational time $O(n \log n)$ such that

$$\frac{H_n^*}{\log n} \to \frac{1}{Q} \text{ in probability.}$$

Interestingly, one can further reduce the height by considering tries with k choices for large k. We prove that if k is sufficiently large (but fixed) then the ratio $H_n^*/\log n$ approaches $1/H$ with arbitrary precision, a bound that cannot be improved.

Finally, we consider tries with k choices for symmetric sources when k is allowed to grow with n. In particular, we show that if the number of choices per datum is proportional to $\log n$, then with high probability a nearly perfectly balanced trie exists with $H_n - F_n \leq 2$. Furthermore, we show that by a natural greedy on-line algorithm one also achieves nearly perfect balancing with $H_n - F_n \leq 7$ with probability $1 - o(1)$. This has applications for load balancing in peer-to-peer networks. In particular, the result implies that if in a peer-to-peer network in which ID's of the n hosts are organized on a circle and upon arrival, each host is allowed to try $c \log n$ randomly chosen ID's and choose the one that maximizes its distance from its neighbors then the maximal load balance ratio remains bounded with high probability.

2. Two-choice tries.

In this section we consider the situation when each datum has two independent strings X_i and Y_i drawn from our string distribution, and that we are free for pick one of the two for inclusion in the trie. Define $Z_i(0) = X_i, Z_i(1) = Y_i$, let $\{i_1, \ldots, i_n\} \subseteq \{0,1\}^n$, and let $H_n(i_1, \ldots, i_n)$ denote the height of the trie for $Z_1(i_1), \ldots, Z_n(i_n)$. Thus, with n data pairs, we have 2^n possible random tries. Let $H_n^* = \min_{i_1,\ldots,i_n} H_n(i_1, \ldots, i_n)$ be the minimal height over all these 2^n tries.

The paradigm of two choices, applied here for tries, was successfully applied in hashing, see Azar, Broder, Karlin and Upfal (1994, 1999), Czumaj and Stemann (1997) and Pagh and Rodler (2001).

We show the following:

THEOREM 1. *Assume that the vector of p_i's is nontrivial ($\max_i p_i < 1$). Then $H_n^*/\log n \to 1/Q$ in probability. In particular, for fixed $t \in \mathbf{R}$,*

$$\mathbf{P}\left\{H_n^* \geq \frac{\log n + t}{Q}\right\} \leq 8e^{-t}.$$

Also, for all $\epsilon > 0$,

$$\lim_{n\to\infty} \mathbf{P}\left\{H_n^* \leq \frac{(1-\epsilon)\log n}{Q}\right\} = 0.$$

This theorem shows that the asymptotic improvement over standard random tries is 50%. Furthermore, the height is less than or equal to the height of the corresponding PATRICIA and digital search trees.

In Section 2.2, we deal with an efficient algorithm for constructing a good trie. The upper bound in Theorem 1 will be shown to hold for H_n, the height of the trie obtained by a simple algorithm. Hence, the algorithm is optimal to within a term that is $o(\log n)$ in probability. We stress that the results in the entire section are for tries with an unlimited number of possible children. We start by constructing a two-choice trie that achieves the upper bound of Theorem 1. This is followed by a description of an $O(n \log n)$ on-line algorithm that realizes this trie. Finally, we prove the lower bound.

2.1 The upper bound.

In the infinite trie formed by all $2n$ strings, we consider all subtrees T_j, $j \geq 1$ rooted at nodes at distance d from the root. We sometimes write $T_j(d)$ to make the dependence upon d explicit. More often, we just use T_j. We say that a string visits T_j if the root of T_j is on the path of the string. Prune this forest of trees by keeping only those that contain at least two leaves. Define

$$\lambda = \sum_i p_i^2.$$

The following lemma is immediate from the definition of the trie.

LEMMA 1. *A bad datum is one in which both of its strings fall in the same T_j. The probability that there exists a bad datum anywhere is not more than*

$$n\lambda^d.$$

We also need a lemma for pairs of data.

LEMMA 2. *A colliding pair of data is such that for some $j \neq k$, each datum in the pair delivers one string to T_j and one string to T_k. The probability that there is a colliding pair of data anywhere is not more than*

$$2n^2\lambda^{2d}.$$

PROOF. Fix a pair of data. The probability that the first strings in each datum fall in the same T_j is λ^d. The probability that each datum puts its first string in T_j and second in T_k is thus not more than λ^{2d}. Summing over all pairs and combinations of collisions, we obtain the upper bound

$$\binom{n}{2} \times 4 \times \lambda^{2d}. \ \square$$

Next, we consider a multigraph $G(d)$ (or just G), whose vertices represent the $T_j(d)$'s. We connect T_j with T_ℓ if a datum deposits one string in each of these trees. With T_j we keep a list of indices of data for which at least one of the two strings visits T_j.

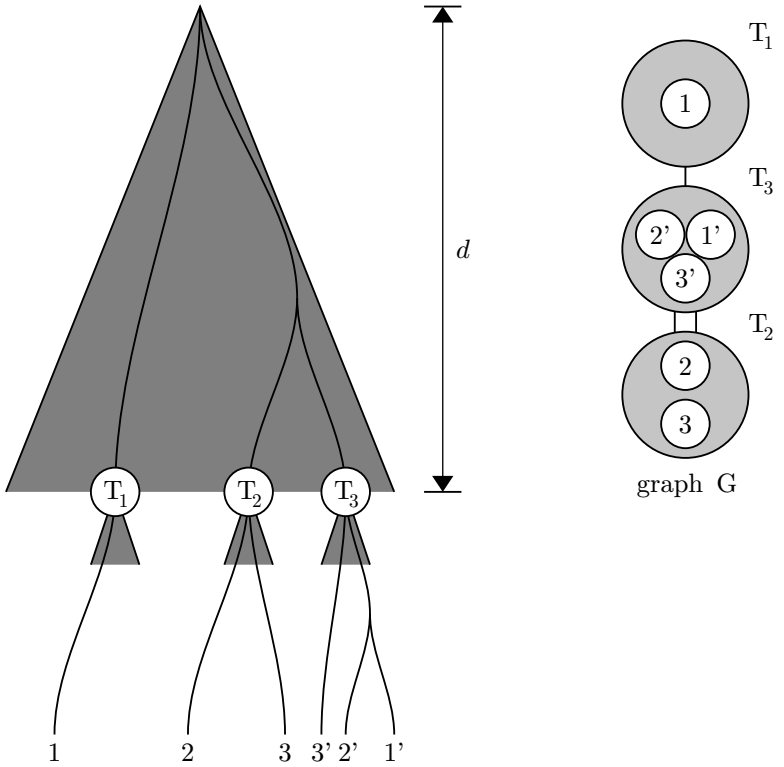

Figure 1. The multigraph G and an infinite trie for $n = 3$ pairs of strings, denoted by $(1, 1')$, $(2, 2')$ and $(3, 3')$. Note that $(2, 2')$ and $(3, 3')$ is a colliding pair.

LEMMA 3. *The probability that G has a cycle of length ≥ 3 is not more than*

$$\frac{(4n)^3\lambda^{3d}}{1 - 4n\lambda^d}.$$

PROOF. The probability of a cycle of length ℓ can be bounded by the number of possible data assignments times the probability that the ℓ pairs of data are in the given lists. This is not more than

$$2^\ell\,(2n)^\ell\,\lambda^{d\ell}.$$

Thus, the probability of a cycle of length ≥ 3 does not exceed

$$\sum_{\ell=3}^{\infty}(4n)^\ell\lambda^{d\ell} = \frac{(4n)^3\lambda^{3d}}{1-4n\lambda^d}. \quad \square$$

So, finally, assuming that there is no bad datum, no colliding data and no cycle (so that G is a forest with no multiedges), we can assign strings as follows. For each tree in turn pick any node as the root. Then choose any one of the strings in the root node's list. For all other strings in the root's list, choose the companion string of the same datum (found by following edges away from the root). This either terminates, or has an impact on one or more child trees. But for the child tree of the root, we have fixed one string (as we did for the root), and thus choose again companion strings for that child list, and so forth. This process is continued until one string of each datum is chosen for the trie. In this manner, the height H_n of the trie for the data selected by this procedure is not more than d. Therefore, we have shown that

$$\begin{aligned}\mathbf{P}\{H_n > d\} &\leq \mathbf{P}\{\text{there exists a bad datum}\} + \mathbf{P}\{\text{there exists a colliding pair}\} + \mathbf{P}\{\text{there exists a cycle}\}\\ &\leq n\lambda^d + 2n^2\lambda^{2d} + \frac{(4n)^3\lambda^{3d}}{1-4n\lambda^d}.\end{aligned}$$

If we set $A = n\lambda^d$, then

$$\mathbf{P}\{H_n > d\} \leq \min\left(A + 2A^2 + \frac{64A^3}{1-4A}, 1\right) \leq 4A1_{A\leq 1/8} + 1_{A>1/8} \leq 4A1_{A\leq 1/8} + 8A1_{A>1/8} \leq 8A.$$

We summarize:

THEOREM 2. *There exists a way of assigning the strings such that the trie satisfies, for all n and d,*

$$\mathbf{P}\{H_n > d\} \leq 8n\lambda^d.$$

In particular,

$$\mathbf{P}\left\{H_n > \frac{\log n}{Q} + t\right\} \leq 8e^{-t}$$

for all n, t and vectors $(p_1, p_2, \ldots)$.

The proof above relates to our algorithm. It is noteworthy that the algorithm has a bit of slack, because a choice of strings in the n data is only possible if and only if one of the connected components in G has more than one cycle. With one cycle, it is still possible to pick the strings. However, it is not worth the trouble to design a more complicated algorithm for a small gain in height. That limitation of the gain in height is explained by the lower bound shown below.

2.2 Algorithmic considerations.

Both the off-line and on-line constructions of the two-choice trie achieving the upper bound in Theorem 1 can be carried out by maintaining a data structure that guarantees that the height H_n is at all times the smallest integer d such that $G(d)$ is acyclic, where multiple edges between nodes are counted as cycles. The n data pairs are stored in an array, and the infinite trie for $2n$ strings, truncated at height H_n, is stored as well. Each leaf in this trie t_n (truncated at H_n) represents in fact a subtree T_i, which in turn contains the nodes of $G = G(d)$ with $d = H_n$. To find things easily, each node of G has a pointer to a linked list of data strings. And vice versa, each string in the array of n pairs of strings has a pointer to the subtree T_i in the trie to which it belongs.

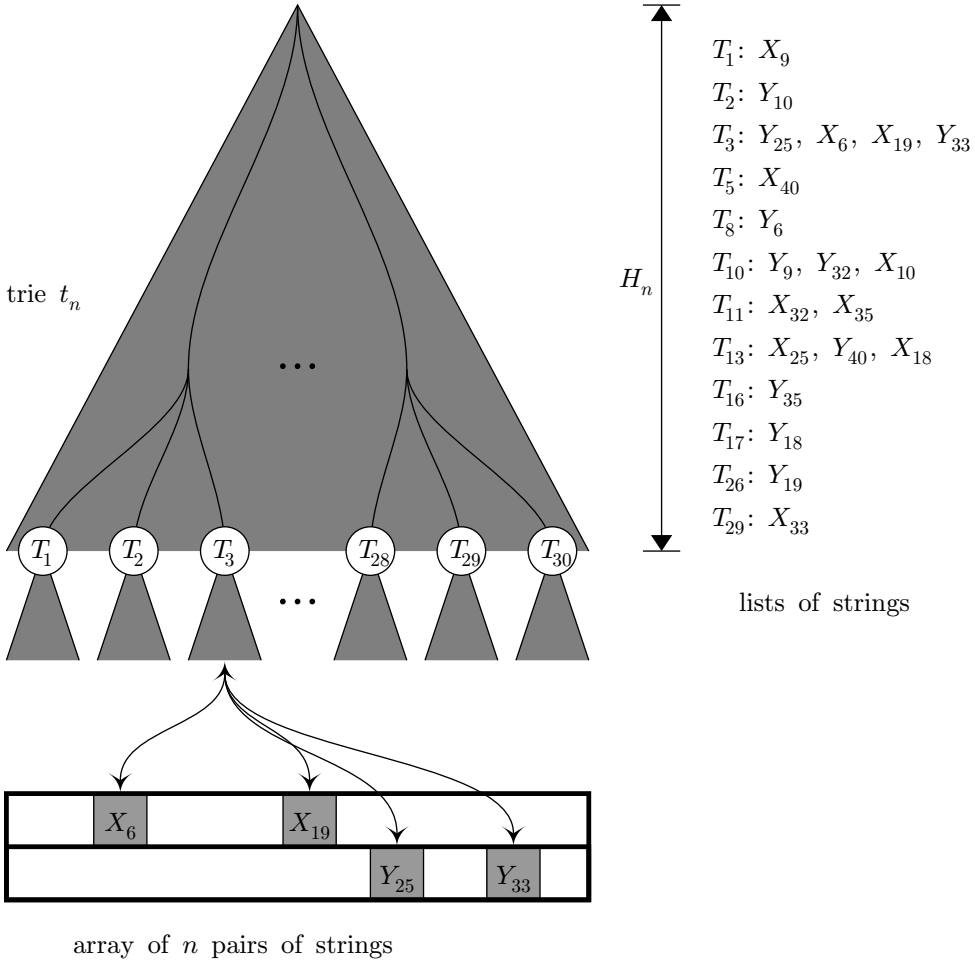

Figure 2. The basic data structure needed to efficiently construct the two-choice trie in an on-line manner.

Nodes of G are organized in turn in a parent pointer data structure commonly used for managing forests (see, e.g., Tarjan, 1983), and one linked list per tree in the forest (G). The only operations needed on this structure are findroot (self-explanatory) and join (join two components). Findroot is implemented by following links to the root. We ensure that the height of the parentpointer trees is always bounded by

$\log_2 n$, where n is the number of nodes in the tree. A join proceeds by making the root of the smallest of the two subtrees the child of the root of the other tree. The two linked lists of the nodes in the trees are joined as well. This takes constant time. By picking the smaller tree, we see that the height of the parent pointer tree is never more than $\log_2 n$.

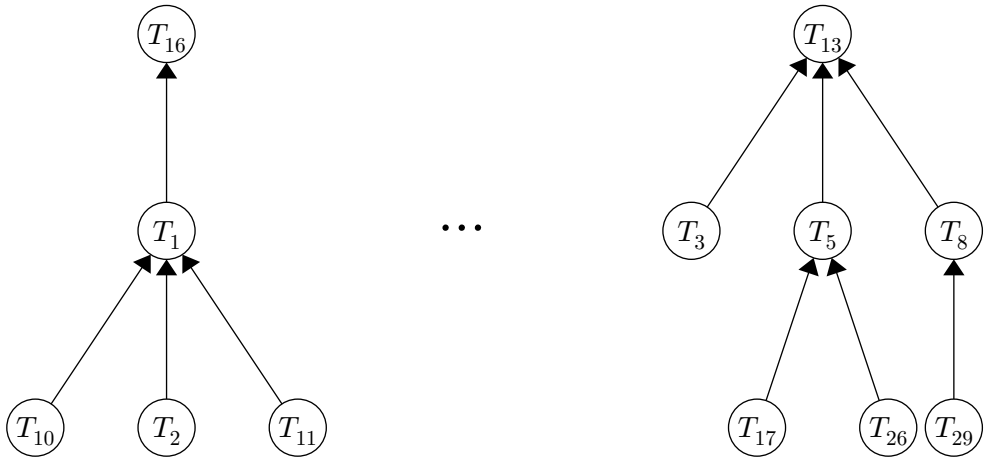

Figure 3. The forest G is maintained by organizing each tree component in a parentpointer tree. The components in the figure might correspond, for example, to the list of strings given in the previous figure.

Assume that we have maintained this structure with n data pairs and that the height of t_n is $h = H_n$. Then, inserting data pair $n+1$, say (X, Y), into the structure proceeds as follows: for X and Y in turn, determine the nodes of G in which they live, by following paths down from the root in t_n. Let these nodes of G be T_j and T_k. Run findroot on both nodes, to determine if they live in the same component. If they do not, then join the components of T_j and T_k, add X to the linked list of T_j, and add Y to the linked list of T_k. The work done thus far is $O(h + \log n)$.

If T_j and T_k are in the same component, then adding an edge between them would create a cycle in G. Thus, we destroy t_n and create t'_n of height $h+1$ from scratch in time $O(n)$ (see below how). An attempt is made to insert (X, Y) in t'_n. We repeat these attempts, always increasing h by one, until we are successful. The time spent here is $O(n\Delta h)$, where Δh is the number of attempts.

In a global manner, starting from an empty tree, we see that to create this structure of size n takes time bounded by $O(nH_n + n \log n)$. By Theorem 2, $\mathrm{E}\{H_n\} = O(\log n)$. Therefore, the expected time is $O(n \log n)$, which cannot be improved upon.

The space used up is $O(nH_n)$. It is known that for standard tries the expected number of internal nodes is $O(n/H)$, where H is the entropy (Régnier and Jacquet, 1989). While it is still true that the expected number of internal nodes in the final trie is $O(n)$ (because it is smaller than that for the trie constructed using all $2n$ data strings), the intermediate structure needed during the construction is possibly of expected size of the order of $n \log n$.

Two details remain to be decided. First, we have to choose one element in each data pair. This can be done quite simply by considering the roots for all components in turn. From the root tree in a component, say T_r, pick any of its member strings, and assign it. Then traverse the component of T_r by depth first search, where the edges are the edges of G (an edge of G is easily determined from the list of

strings in T_r, as each string points back to the tree T_i to which it belongs). At each new node visited, if possible, pick the first string whose sibling has not been assigned yet. This process cannot get stuck as there is no cycle in G, and it takes time $O(n)$. In Figures 2 and 3, the component whose root tree is T_{13} is traversed in this order: $T_{13}, T_3, T_8, T_{26}, T_{29}, T_5$ and T_{17}. The string assignments for the six string pairs in that 7-node component are, in order of assignment, $X_{25}, X_6, Y_6, Y_{19}, X_{33}, X_{40}$ and Y_{18}. After the assignment of all strings, it is a trivial matter to construct the final trie in time $O(nH_n)$.

The second detail concerns the extension of G and the necessary data structures when the height h is increased by one. Here we first update the trie by splitting all the trees T_j appropriately. Create the connected components by depth first search following the edges of G. This takes time $O(n)$. Set up the parent pointer data structure for each component of G by picking a root arbitrarily and making all other nodes children of the root.

The discussion above ensures that we can construct the two-choice trie incrementally in $O(n \log n)$ expected time. Also, in a dynamic setting, if the data structure defined above is maintained, then an insertion can be performed in $O(\log n)$ expected amortized time, under the assumptions of the theorems in this paper.

2.3 The lower bound of Theorem 1.

The notation is as above. We denote the data by (X_i, Y_i), $1 \le i \le n$, where all $2n$ components are i.i.d. strings of independent symbols. We call i a data index, and will use symbols like u, v and z as data indices below.

For fixed r, consider a collection of $3r+3$ data indices, u, u', v, v', w, w', and $z(1)$ through $z(3r-3)$, where the indices are each restricted to have their first components in disjoint blocks of size $n/(3r+3)$ each: thus, $u \in \{1, \ldots, n/(3r+3)\}$, and so forth. We assume without loss of generality that $n/(3r+3)$ is integer-valued. The collection of all such $3r+3$-tuples is denoted by S and its members are denoted by s. Clearly,

$$|S| = \left(\frac{n}{3r+3}\right)^{3r+3}.$$

In what follows, we consider the graph G for a fixed height d, which we will chose a bit later on. That d is fine-tuned to make sure that certain bi-cycled components called r-worms are likely to occur.

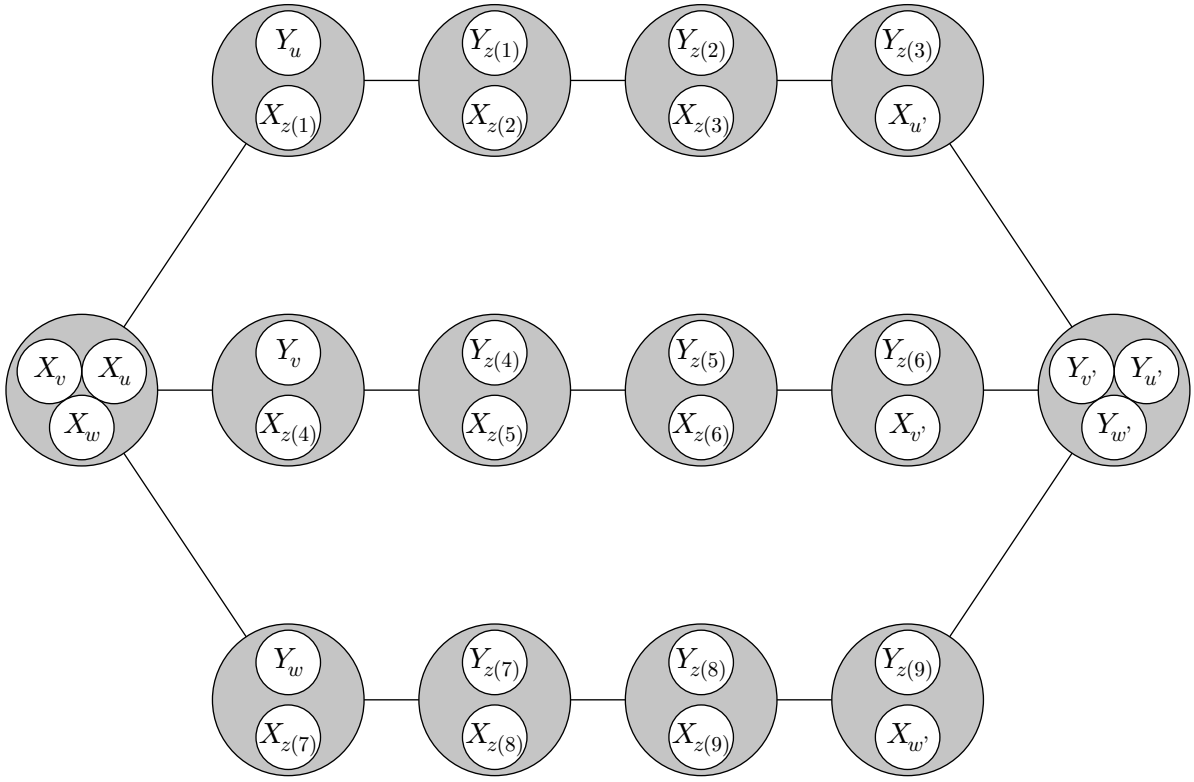

Figure 4. The multigraph G for the r-worm when $r = 4$. The leftmost occurrence of a datum such as u is for the first string of that datum (X_u), while the rightmost occurrence is for the second string (Y_u).

We say that $s \in S$ defines a r-worm if the following are in the same list:

(i) X_u, X_v, X_w.

(ii) $Y_{u'}$, $Y_{v'}$, $Y_{w'}$.

(iii) Y_u and $X_{z(1)}$; $Y_{z(1)}$ and $X_{z(2)}$; $Y_{z(2)}$ and $X_{z(3)}$; …; $Y_{z(r-1)}$ and $X_{u'}$. This defines a chain of $r-1$ lists of size two.

(iv) Y_v and $X_{z(r)}$; $Y_{z(r)}$ and $X_{z(r+1)}$; $Y_{z(r+1)}$ and $X_{z(r+2)}$; …; $Y_{z(2r-2)}$ and $X_{v'}$. This defines a chain of $r-1$ lists of size two.

(v) Y_w and $X_{z(2r-1)}$; $Y_{z(2r-1)}$ and $X_{z(2r)}$; $Y_{z(2r)}$ and $X_{z(2r+1)}$; …; $Y_{z(3r-3)}$ and $X_{w'}$. This defines a chain of $r-1$ lists of size two.

We have two lists of three, who by brotherhood are connected to each other by three chains of lists of two. Observe that the r-worm refers to the subgraph of G induced by the $3r+3$ data indices involved. If an r-worm exists, and we are forced to pick one string from each of the $3r+3$ data items, then at least one of the lists in it must accept two strings, no matter how we assign the strings for the indices represented in S (by the pigeon-hole principle, as the number of lists in the r-worm is $3r+2$).

Let N be the (random) number of r-worms. We thus have

$$\mathrm{P}\{H_n^* \le d\} \le \mathrm{P}\{N = 0\}.$$

We bound the right-hand side from above by the second moment method. Clearly,

$$\mathrm{E}\{N\} = |S|\lambda_2^{3r}\lambda_3^2,$$

where

$$\lambda_\ell \stackrel{\text{def}}{=} \left(\sum_i p_i^\ell\right)^d.$$

We have the following inequalities, which are all consequences of Jensen's inequality:

$$\left(\sum_i p_i^q\right)^{\frac{1}{q}} \downarrow$$

as $q \uparrow \infty$, and

$$e^{-H} \le \left(\sum_i p_i^q\right)^{\frac{1}{q-1}} \uparrow$$

as $1 \le q \uparrow \infty$. The last expression is the so-called Rényi entropy of order q. Using $\lambda_3^2 \ge \lambda_2^4$, we have

$$\mathrm{E}\{N\} \ge \frac{(n\lambda_2)^{3r+4}}{n(3r+3)^{3r+3}}.$$

It is convenient to use Z_s for the indicator function that $s \in S$ defines an r-worm. We write $s \sim s'$ when the data indices of s and s' overlap and $s \ne s'$. Let $\delta(s,s')$ be the number of positions in which s and s' agree (i.e., $3r+3$ minus the Hamming distance): $0 \le \delta(s,s') \le 3r+3$. Using $\mathrm{V}\{\cdot\}$ to denote the variance, we have

$$\mathrm{V}\{N\} \le \sum_s \mathrm{E}\{Z_s\} + \sum_{s\sim s'} \mathrm{E}\{Z_s Z_{s'}\} = \mathrm{E}\{N\} + \sum_{s\sim s'} \mathrm{E}\{Z_s Z_{s'}\}$$

For $0 \le \ell \le 3r+3$, we have

$$\left|\{(s,s') : \delta(s,s') = \ell\}\right| = \left(\frac{n}{3r+3}\right)^{3r+3} \times \left(\frac{n}{3r+3} - 1\right)^{3r+3-\ell} \times \binom{3r+3}{\ell} \le \frac{n^{6r+6-\ell}}{\ell!}.$$

For $s \sim s'$, set $\ell = \delta(s, s')$. In order to calculate the value of $\mathrm{E}\{Z_s Z_{s'}\}$ for two given strings $s, s' \in S$, we must take into account the positions of those indices that agree in s and s'. (There are ℓ of them.) We say that s and s' define a joint r-worm if both s and s' define r-worms (i.e., $Z_s Z_{s'} = 1$). Given s, s', consider the multigraph G for the joint r-worm (similar to Figure 2 but with s' added). The size of a node in this multigraph is defined as the number of strings corresponding to the node when s and s' define a joint r-worm. For $i = 2, 3, 4, 5$, let n_i be the number of nodes in the graph of size i. (Observe that the sizes of the nodes vary from 2 to a maximum of 5.) Then clearly

$$\mathrm{E}\{Z_s Z_{s'}\} = \lambda_2^{n_2} \lambda_3^{n_3} \lambda_4^{n_4} \lambda_5^{n_5} \le \lambda_2^{n_2} \lambda_3^{\frac{3n_3+4n_4+5n_5}{3}}.$$

We will show below in Lemma 5 that we always have $3n_3 + 4n_4 + 5n_5 \ge 12$ when $\ell \le r$. For $\ell \le r$,

$$\mathrm{E}\{Z_s Z_{s'}\} \le \lambda_2^{n_2} \lambda_3^{\frac{3n_3+4n_4+5n_5-12}{3}} \lambda_3^4 \le \lambda_2^{\frac{2n_2+3n_3+4n_4+5n_5-12}{2}} \lambda_3^4 \le \lambda_2^{6r-\ell} \lambda_3^4,$$

where we used $\lambda_3^{1/3} \le \lambda_2^{1/2}$ and the fact that $2n_2 + 3n_3 + 4n_4 + 5n_5 - 12 = 12r - 2\ell$ (see Lemma 4 below). For $\ell > r$, we simply have

$$\mathrm{E}\{Z_s Z_{s'}\} \le \lambda_2^{n_2} \lambda_3^{\frac{3n_3+4n_4+5n_5}{3}} \le \lambda_2^{\frac{2n_2+3n_3+4n_4+5n_5}{2}} = \lambda_2^{6r+6-\ell}.$$

By the second moment method Chung and Erdős (1952); see also Janson et al (2000)),

$$\begin{aligned}
\mathrm{P}\{N = 0\} &\le \frac{\mathrm{V}\{N\}}{\mathrm{V}\{N\} + (\mathrm{E}\{N\})^2} \le \frac{\mathrm{V}\{N\}}{(\mathrm{E}\{N\})^2} \\
&\le \frac{1}{\mathrm{E}\{N\}} + \frac{\sum_{s \sim s'} \mathrm{E}\{Z_s Z_{s'}\}}{(\mathrm{E}\{N\})^2} \\
&\le \frac{1}{\mathrm{E}\{N\}} + \sum_{\ell=1}^{3r+3} \sum_{s \sim s' : \delta(s,s') = \ell} \max_{s \sim s' : \delta(s,s') = \ell} \frac{\mathrm{E}\{Z_s Z_{s'}\}}{(\mathrm{E}\{N\})^2} \\
&\le \frac{1}{\mathrm{E}\{N\}} + \sum_{\ell=1}^{r} n^{6r+6-\ell} \frac{\lambda_2^{6r-\ell} \lambda_3^4}{\ell! (\mathrm{E}\{N\})^2} + \sum_{\ell > r} \frac{(n\lambda_2)^{6r+6-\ell}}{\ell! (\mathrm{E}\{N\})^2} \\
&\stackrel{\mathrm{def}}{=} I + II + III.
\end{aligned}$$

For fixed $\epsilon > 0$, we set $d = \lfloor (1-\epsilon) \log n / \log(1/\lambda) \rfloor$ and $\alpha = n\lambda_2$. Note that $\alpha = n\lambda^d \ge n^\epsilon \to \infty$. In particular, $\alpha \ge 1$ for all $n \ge 1$. Take r so large that jointly $\epsilon(3r+4) > 1$ (so that $\mathrm{E}\{N\} \to \infty$ and thus $I = o(1)$) and $\epsilon(r+3) > 2$ (needed below). Then

$$II \le \sum_{\ell=1}^{r} \frac{(3r+3)^{6r+6}}{\ell! \, (n\lambda_2)^\ell} \le (3r+3)^{6r+6} \left(e^{\frac{1}{n\lambda_2}} - 1 \right).$$

Finally, using $\mathrm{E}\{N\} \ge \frac{(n\lambda_2)^{3r+4}}{n(3r+3)^{3r+3}}$,

$$\begin{aligned}
III &\le \sum_{\ell=r+1}^{3r+3} \frac{(3r+3)^{6r+6}}{\ell! \lambda_2^2 \, (n\lambda_2)^\ell} \\
&\le \sum_{\ell=r+1}^{3r+3} \frac{(3r+3)^{6r+6} n^2}{\ell! \, (n\lambda_2)^{2+\ell}} \le \frac{(3r+3)^{6r+6} e n^2}{(r+1)! \, (n\lambda_2)^{r+3}}.
\end{aligned}$$

If C_r denotes a constant depending upon r only, we see that

$$\mathrm{P}\{N=0\} \le C_r\left(n\alpha^{-3r-4} + e^{1/\alpha} - 1 + n^2\alpha^{-r-3}\right) \le C_r\left(n^{-\epsilon(3r+4)} + e^{n^{-\epsilon}} - 1 + n^{2-(r+3)\epsilon}\right) = o(1).$$

That concludes the proof of the lower bound.

It remains to show the two structural properties of the joint r-worms defined by s and s'.

LEMMA 4.

$$2n_2 + 3n_3 + 4n_4 + 5n_5 = 12r + 12 - 2\delta(s, s').$$

PROOF. The proof goes by induction. At the outset, we start with two disjoint r-worms so that $n_2 = 6r$ and $n_3 = 4$, and join one by one the $\delta(s, s')$ entries of the r-worm defined by s' that coincide with those of s. One can verify that each such step makes $2n_2 + 3n_3 + 4n_4 + 5n_5$ decrease by two. (For example, if the u entry is processed first, then clusters of sizes 3, 3, 2 and 2 become clusters of sizes 5 and 3 in the join.) Therefore, $2n_2 + 3n_3 + 4n_4 + 5n_5 = 12r + 12 - 2\delta(s, s')$. □

LEMMA 5. *For $\ell = \delta(s, s') \le r$, we have $3n_3 + 4n_4 + 5n_5 \ge 12$.*

PROOF. The proof is by contradiction. if $3n_3 + 4n_4 + 5n_5 < 12$, then $n_3 + n_4 + n_5 < 4$, so at least one border pair u, v, w, u', v', w' must be identical. Let m_3 be the number of clusters of size 3 that involve non-border pairs. If $m_3 = 0$, then that means that each row of $r - 1$ non-border indices (the z's) is either entirely identical or entirely different in s and s'. But if one such row is identical, then $\ell \ge (r-1) + 2 = r + 1$. It is impossible to have all three rows non-identical, because then m_3 would not be zero. The case $m_3 \ge 2$ can be eliminated, as we would have $n_3 + n_4 + n_5 \ge 2 + m + 3 \ge 4$. So assume $m_3 = 1$. But then the clusters at the borders must both be hit at least once. For $m_3 = 1$, this implies that one row must necessarily be identical in s and s', and one other row has its border pair in common in s and s'. So, once again, $\ell \ge r + 1$. □

2.4 A simple greedy heuristic

Instead of using the elaborate data structure described in Section 2.2 to construct the trie, we could greedily pick one string according to a simple rule: choose the string which, at the time of its insertion would yield the leaf nearest to the root. Once a selection is made, it is impossible to undo it at a later time. This greedy heuristic yields a height that is guaranteed to be 25% better than that of the ordinary trie (see Theorem 3 below), but it cannot achieve the 50% improvement of the main method described in this paper (see Theorem 4 below). In this section, H_n refers to the height of the trie obtained by this greedy heuristic.

THEOREM 3. *Assume the probabilistic model of Theorem 1. For all integer $d > 0$,*

$$\mathrm{P}\{H_n \ge d\} \le 4n^3 e^{-2dQ} + 2n^2 e^{-3dQ/2}.$$

Thus, for any $t > 0$,

$$\mathrm{P}\left\{H_n \ge \frac{3\log n + t}{2Q}\right\} \le 4e^{-t} + 2n^{-1/4}e^{-3t/4}.$$

PROOF. We consider once again the multigraph $G = G(d)$ of Section 2.1, whose vertices represent the $T_j(d)$'s. We connect T_j with T_ℓ if a datum deposits one string in each of these trees. Remove from this graph all T_j's that have just one datum, and their incident edges. Call the remaining graph G'. We claim that if $H_n > d$, then G' has at least one edge. Indeed, assume the contrary. Then consider (X_i, Y_i) at time of processing by the algorithm. Let X_i share a prefix of length at least d with some strings X_j or Y_j for $j < i$ (collected in a set A_i), Let B_i similarly be all strings X_j or Y_j for $j < i$ that share a prefix of length d with Y_i. By assumption, either A_i or B_i is empty. So, the greedy algorithm can pick X_i or Y_i so that no other previous string shares a prefix of length d or more. Let C_i denote the event that X_i and Y_i have a common prefix of length d. By negative association of the components of a multinomial random variable,

$$\begin{aligned}
\mathrm{P}\{H_n \geq d\} &\leq \sum_{i=1}^{n} \mathrm{P}\{\min(|A_i|, |B_i|) \geq 1\} \\
&\leq \sum_{i=1}^{n} \left(\mathrm{P}^2\{|A_i| \geq 1\} + \mathrm{P}\{C_i, |A_i| \geq 1\} \right) \\
&\leq \sum_{i=1}^{n} (2i-2)^2 \left(\sum_{\ell} p_\ell^2 \right)^{2d} + \sum_{i=1}^{n} (2i-2) \left(\sum_{\ell} p_\ell^3 \right)^{d} \\
&\leq 4n^3 e^{-2dQ} + 2n^2 e^{-3dQ/2}. \quad \square
\end{aligned}$$

The bound of Theorem 3 is tight in the finite equiprobable case $p_1 = \cdots = p_\beta = 1/\beta$, and thus establishes the suboptimality of the greedy heuristic in that important special case. By continuity, the suboptimality carries over to an open neighborhood of the equiprobable probability distribution (with respect to the total variation metric).

THEOREM 4. *Assume the probabilistic model of Theorem 1, and that there exists an integer β such that $p_1 = \cdots = p_\beta = 1/\beta$. Then, for all $\epsilon > 0$,*

$$\lim_{n \to \infty} \mathrm{P}\left\{ H_n \leq \frac{(3-\epsilon)\log n}{2Q} \right\} = 0.$$

PROOF. Since all strings of length m have the same probability $1/\beta^m$, it is clear that if we randomize the selection in case of a tie, then the events $[Z_i = X_i]$ (in the notation of the proof of Theorem 3) are independent Bernoulli (1/2) random variables. Also, $1_{Z_i = X_i}$ is independent of X_i. Let S be the set of indices for which $Z_i = X_i$. For fixed t, we call a triple of indices $(\ell > i > j)$ good if $i, j \in S$, and if $L(X_\ell, X_i) \geq t$ and $L(Y_\ell, X_j) \geq t$, where $L(.,.)$ is the lenghth of the common prefix of the two argument strings. If a good triple exists, then $H_n \geq t$. The expected number of good triples (N) is

$$\binom{n}{3} \frac{1}{4} \left(\sum_i p_i^2 \right)^{2t} \sim \frac{n^3 e^{-2tQ}}{24}.$$

This tends to infinity if we take $t \sim 3(1-\epsilon)\log n/(2Q)$ for $\epsilon > 0$. It is a routine matter to apply the second moment method to establish that in that case, $\mathrm{P}\{N > 0\} \to 1$. We sketch the proof. It suffices to establish that $\mathrm{V}\{N\} = o((\mathrm{E}\{N\})^2)$. Simple case by case studies reveal that

$$\mathrm{V}\{N\} = O\left(n^5 \left(\lambda_2^{2t} \lambda_3^t + \lambda_3^{2t} \right) \right) + O\left(n^4 \left(\lambda_5^t + \lambda_2^t \lambda_3^t + \lambda_3^{2t} \right) \right)$$

where $\lambda_\ell \stackrel{\text{def}}{=} \sum_i p_i^\ell$. Using the fact that in the equiprobable case, $\lambda_\ell = \lambda_2^{\ell-1}$, we have $\mathrm{V}\{N\} = O\left(n^5\lambda_2^{4t} + n^4\lambda_2^{3t}\right)$, and this is easily seen to be $(\mathrm{E}\{N\})^2 \times \phi_n$, where $\phi_n = O(1/n + 1/(n^2\lambda_2^t)) = o(1/\sqrt{n})$. □

REMARK 1. If we perform the greedy heuristic with k choices, then an easy extension of the proof of Theorem 4 shows that for fixed $t > 0$,

$$\mathrm{P}\left\{H_n \geq \frac{(k+1)\log n + t}{kQ}\right\} \leq k^k e^{-t} + o(1).$$

3. Multiple choice tries.

3.1 The entropy lower bound.

If we have k choices per datum, then the height can be further reduced. However, in any case, we cannot go beyond the entropy bound, as we will prove in this section. Recall that for an ordinary trie, $D_n = o(\log n)$ in probability when $H = \infty$. We will not deal with those cases here. Let $k \geq 2$ be a fixed integer. Consider n data, each composed of k independent strings of i.i.d. symbols drawn from any distribution on $\mathcal{N}$. Let $H_n^*(k)$ denote the minimal height of any trie of n strings that takes one string of each datum. We have the following lower bound:

THEOREM 5. *If the vector of p_i's is nontrivial ($\max_i p_i < 1$) and $H < \infty$, then for all $k \geq 2$ and for all $\epsilon > 0$,*

$$\lim_{n\to\infty} \mathrm{P}\left\{H_n^*(k) \leq \frac{(1-\epsilon)\log n}{H}\right\} = 0.$$

PROOF. We fix $d = \lfloor (1-\epsilon)\log n/H \rfloor$, and consider the partition of space defined by all different prefixes of length d. There is an obvious analogy with cells or buckets. Partition the space of all strings into buckets, where each bucket corresponds uniquely to a string of length d. We say that a string falls in a given bucket if its length d prefix coincides with that for the bucket. Consider the kn strings, k per datum, and let M_n denote the number of occupied buckets. It is clear that

$$[H_n^*(k) \leq d] \subseteq [M_n \geq n].$$

Thus, it suffices to show that

$$\lim_{n\to\infty} \mathrm{P}\{M_n \geq n\} = 0.$$

From an extension of Talagrand's inequality (see, e.g., Boucheron, Lugosi and Massart (2000, 2003) or Devroye (2002)),

$$\mathrm{P}\{M_n \geq \mathrm{E}\{M_n\} + t\} \leq \exp\left(-\frac{t^2}{2\mathrm{E}\{M_n\} + 2t/3}\right) \ , \ t \geq 0 \ .$$

Taking $t = n/2$, we are thus done if we can show that $\mathrm{E}\{M_n\} = o(n)$.

Let S be the (infinite) set of buckets in our collection, and for $s \in S$, define $p(s) = \mathbf{P}\{X \in s\}$, where X is a length d string consisting of i.i.d. symbols $Y_1, \ldots, Y_d$ drawn from $\{p_i\}$. Define

$$Z = \prod_{i=1}^{d} p_{Y_i}.$$

Note that for fixed constant $\delta > 0$,

$$|\{s : knp(s) \geq 1/\delta\}| \leq \sum_{s \in S} \delta knp(s) = \delta kn,$$

where we used Markov's inequality. Thus

$$\begin{aligned}
\mathbf{E}\{M_n\} &= \sum_{s \in S} \left(1 - (1 - p(s))^{kn}\right) \\
&\leq \sum_{s \in S : knp(s) \leq 1/\delta} knp(s) + \delta kn \\
&\leq kn\mathbf{P}\left\{Z \leq \frac{1}{\delta kn}\right\} + \delta kn \\
&= kn\mathbf{P}\left\{\sum_{i=1}^{d} \log(1/p_{X_i}) \geq \log(\delta kn)\right\} + \delta kn \\
&= o(kn) + \delta kn,
\end{aligned}$$

where we used the law of large numbers, as $\mathbf{E}\{\log(1/p_{Y_1})\} = H$, and thus

$$\frac{\sum_{i=1}^{d} \log(1/p_{Y_i})}{dH} \to 1$$

in probability. Since $\delta > 0$ was arbitrary, we have $\mathbf{E}\{M_n\} = o(n)$. □

3.2 The entropy upper bound.

The bound of Theorem 5 is tight in the following sense:

THEOREM 6. *Assume $H < \infty$. In the notation of Theorem 5, we have for all $\epsilon > 0$, there exists k large enough such that*

$$\lim_{n \to \infty} \mathbf{P}\left\{H_n^*(k) \geq \frac{(1+\epsilon)\log n}{H}\right\} = 0.$$

PROOF. Fix $\epsilon > 0$ and k. Consider the trie formed by the kn input strings. For the j-th string of the i-th datum, let $D_{i,j}$ be the maximal common prefix length with any string belonging to a datum not equal to i (the other strings for datum i are thus ignored). Note that $H_n^*(k) \leq \max_i \min_j D_{i,j}$. It suffices to show that

$$\mathbf{P}\left\{\max_i \min_j D_{i,j} \geq \frac{(1+\epsilon)\log n}{H}\right\} = o(1),$$

which is implied by

$$\mathbf{P}\left\{\min_j D_{1,j} \geq \frac{(1+\epsilon)\log n}{H}\right\} = o\left(\frac{1}{n}\right).$$

Let

$$d = \left\lceil \frac{(1+\epsilon)\log n}{H} \right\rceil .$$

Let $s_1, \ldots, s_k$ be the strings of length d corresponding to datum 1 (note that duplicates are allowed), and let $N_1, \ldots, N_k$ be the number of strings among the remaining $(n-1)k$ strings whose length d prefix agree with $s_1, \ldots, s_k$, respectively. Then

$$\mathrm{P}\left\{ \min_j D_{1,j} \geq d \right\} = \mathrm{P}\left\{ N_1 > 0, \ldots, N_k > 0 \right\}.$$

Let A be the event that the cardinality $C = |\{s_1, \ldots, s_k\}|$ is $\leq k/2$, where we assume k to be even. Set $\theta = \max_\ell p_\ell$. We have

$$\mathrm{P}\{A\} \leq \binom{k}{k/2} ((k/2)\theta^d)^{k/2} = o(1/n)$$

by choice of k. We use the notation $a_1 \neq a_2 \neq \cdots \neq a_k$ to denote the event that $a_1, a_2, \ldots, a_k$ are all different. Let B be the event $[s_1 \neq \cdots \neq s_{k/2}]$. We have $\mathrm{P}\{B^c\} \leq \binom{k/2}{2}\theta^d = o(1)$. Given $s_1 \neq \cdots \neq s_k$, the N_i's are part of multinomial random vector and thus negatively associated (Mallows, 1968). Therefore, if S denotes the number of different values occurring among the s_i's, and $a_1, \ldots, a_S$ are the indices of the first occurrences of these S values, in order of occurrence among $s_1, \ldots, s_k$,

$$\begin{aligned}
\mathrm{P}\left\{ N_1 > 0, \ldots, N_k > 0 \right\} &= \mathrm{E}\left\{ \mathrm{P}\left\{ N_1 > 0, \ldots, N_k > 0 | s_1, \ldots, s_k \right\} \right\} \\
&\leq \mathrm{P}\{A\} + \mathrm{E}\left\{ 1_{A^c} \mathrm{P}\left\{ N_1 > 0, \ldots, N_k > 0 | s_1, \ldots, s_k \right\} \right\} \\
&\leq \mathrm{P}\{A\} + \mathrm{E}\left\{ 1_{A^c} \prod_{i=1}^{S} \mathrm{P}\left\{ N_{a_i} > 0 | s_{a_i} \right\} \right\} \\
&\qquad \text{(by negative association of multinomial random variables)} \\
&\leq \mathrm{P}\{A\} + \mathrm{E}\left\{ 1_{A^c} \prod_{i=1}^{k/2} \mathrm{P}\left\{ N_{a_i} > 0 | s_{a_i} \right\} \right\} \\
&\leq \mathrm{P}\{A\} + \mathrm{P}\{A^c\} \mathrm{E}\left\{ \prod_{i=1}^{k/2} \mathrm{P}\left\{ N_{a_i} > 0 | s_{a_i} \right\} | A^c \right\} \\
&= \mathrm{P}\{A\} + \mathrm{P}\{A^c\} \mathrm{E}\left\{ \prod_{i=1}^{k/2} \mathrm{P}\left\{ N_i > 0 | s_i \right\} | s_1 \neq \cdots \neq s_{k/2} \right\} \\
&\leq \mathrm{P}\{A\} + \frac{\mathrm{P}\{A^c\} \mathrm{E}\left\{ \prod_{i=1}^{k/2} \mathrm{P}\left\{ N_i > 0 | s_i \right\} \right\}}{\mathrm{P}\{s_1 \neq \cdots \neq s_{k/2}\}} \\
&= \mathrm{P}\{A\} + (1+o(1)) \mathrm{E}\left\{ \prod_{i=1}^{k/2} \mathrm{P}\left\{ N_i > 0 | s_i \right\} \right\} \\
&= \mathrm{P}\{A\} + (1+o(1)) \prod_{i=1}^{k/2} \mathrm{P}\left\{ N_i > 0 \right\} \\
&\qquad \text{(by independence of the } s_i\text{'s)} \\
&= \mathrm{P}\{A\} + (1+o(1)) \left(\mathrm{P}\left\{ N_i > 0 \right\} \right)^{k/2}.
\end{aligned}$$

Let $Z_1, \ldots, Z_d$ be the symbols occurring in s_1. Then

$$\begin{aligned}
\mathrm{P}\{N_1 > 0\} &= \mathrm{E}\left\{1 - \left(1 - \prod_{i=1}^{d} p_{Z_i}\right)^{(n-1)k}\right\} \\
&\le \mathrm{E}\left\{1 - \left(1 - r^d\right)^{(n-1)k}\right\} + \mathrm{P}\left\{\prod_{i=1}^{d} p_{Z_i} > r^d\right\} \\
&\le nkr^d + \mathrm{P}\left\{\sum_{i=1}^{d} \log(1/p_{Z_i}) < d\log(1/r)\right\} \\
&= nkr^d + \exp\left(-d(o(1) + \phi(\log(1/r)))\right)
\end{aligned}$$

as $d \to \infty$ and $r \in (0,1)$ remains fixed, by Cramér's large deviation theorem (see, e.g., Dembo and Zeitouni, 1998). The function $\phi(u)$ is positive for all $u < \mathrm{E}\{\log(1/p_{Z_1})\} = H$. Thus, for all fixed $1 > r > e^{-H}$, there exists a constant $\delta = \delta(r) \in [0,1)$ such that for all d,

$$\mathrm{P}\{N_1 > 0\} \le nkr^d + \delta^d.$$

Observe that if we set $r = \exp(-(1+\epsilon/2)H/(1+\epsilon))$, then $nr^d \le n^{-\epsilon/2}$. Combining all our bounds, we see that

$$\mathrm{P}\left\{\min_j D_{1,j} \ge d\right\} \le o(1/n) + (1+o(1))\left(kn^{-\epsilon/2} + \delta^d\right)^{k/2} = o(1/n) + O\left(n^{-k\epsilon/4} + \delta^{dk/2}\right) = o(1/n)$$

by choice of k. □

Remark 2. One might ask for the exact constant in the weak limit of $H_n^*(k)/\log n$. For $k = 2$, it is $1/Q$ as we showed earlier. However, for $k > 2$, the precise constant is harder to determine. The proof of Theorem 6, suitably extended, yields the following upper bound for k fixed. Assume a finite entropy of second order: $H_2 \stackrel{\mathrm{def}}{=} \sum_i p_i \log^2 p_i < \infty$. Then, for all $\eta > 0$, and all $k > k^*(\eta)$,

$$\lim_{n\to\infty} \mathrm{P}\left\{H_n^*(k) \ge \frac{(1 + (1+\eta)\psi(k))\log n}{H}\right\} = 0,$$

where

$$\psi(k) = \max\left(\frac{4}{k}, \frac{4}{\sqrt{k}}\sqrt{\frac{H_2 - H^2}{H}}\right).$$

A brief sketch follows. It suffices to verify for which ϵ the last estimate in the proof of Theorem 6 is $o(1/n)$. Trivially, $\epsilon > 4/k$ is needed to make the first term $o(1/n)$. So, let us verify under what condition the last term is $o(1/n)$. In other words, with the given choice of r, when is

$$\mathrm{P}\left\{\sum_{i=1}^{d} \log(1/p_{Z_i}) < d\log(1/r)\right\} = o\left(n^{-2/k}\right)?$$

Under the condition $H_2 < \infty$, we can easily verify from Taylor's series with remainder that, as $t \downarrow 0$,

$$\log\left(\sum_i p_i^{t+1}\right) = -tH + (t^2/2)(H_2 - H^2) + o(t^2).$$

By Chernoff's bound, for $t > 0$,

$$\begin{aligned}
\mathrm{P}\left\{\sum_{i=1}^{d} \log(1/p_{Z_i}) < d\log(1/r)\right\} &\leq r^{-td}\left(\sum_i p_i^{t+1}\right)^d \\
&= \exp\left(-d\left(t\log r + tH - (t^2/2)(H_2 - H^2) + o(t^2)\right)\right) \\
&= \exp\left(-d\left(\frac{(H+\log r)^2}{2(H_2-H^2)} + o(t^2)\right)\right) \\
&\qquad \text{(upon putting } t = (H+\log r)/(H_2-H^2) = H(\epsilon/2)/(1+\epsilon)(H_2-H^2)\text{)} \\
&= \exp\left(-d\left(\frac{H^2(\epsilon/2)^2}{2(1+\epsilon)^2(H_2-H^2)} + o(\epsilon^2)\right)\right) \quad (\text{as } \epsilon \downarrow 0) \\
&\leq n^{-\frac{H(\epsilon/2)^2}{2(1+\epsilon)(H_2-H^2)} + o(\epsilon^2)} \\
&= n^{-\frac{H(\epsilon/2)^2}{2(H_2-H^2)} + o(\epsilon^2)}.
\end{aligned}$$

The statement follows if

$$\frac{H(\epsilon/2)^2}{2(H_2-H^2)} > \frac{2}{k}.$$

In other words,

$$\epsilon^2 > \frac{16(H_2-H^2)}{kH}.$$

4. Distributed hash tables and ID management.

Tries occur naturally, but not under the name "trie", in the area of ID management in distributed hash tables. In this area of network research, n hosts are assigned one ID in the unit interval $[0, 1)$. The unit interval is wrapped around to form a ring. At any time, the set of ID's partition $[0, 1)$ into n intervals on the ring. The hosts are organized in the ring, linking to the next smaller and next larger host, and are in general quite unaware of the ID's of the other hosts, except what can be gleaned from traversing the ring in either direction. Hosts are added and deleted in some way, but ID choices are up to the system. Each interval is "owned" by the host to its left. Two parameters are of some importance here. The first one is the balance B_n in the partition, as determined by the ratio of the lengths of the largest to the smallest interval. Secondly, one must be able to determine quickly which host owns an interval in which a given ID x falls. The latter is the equivalent of a search operation. Other complexity parameters for updates are also important, but we will limit the discussion to B_n and the maximal search time.

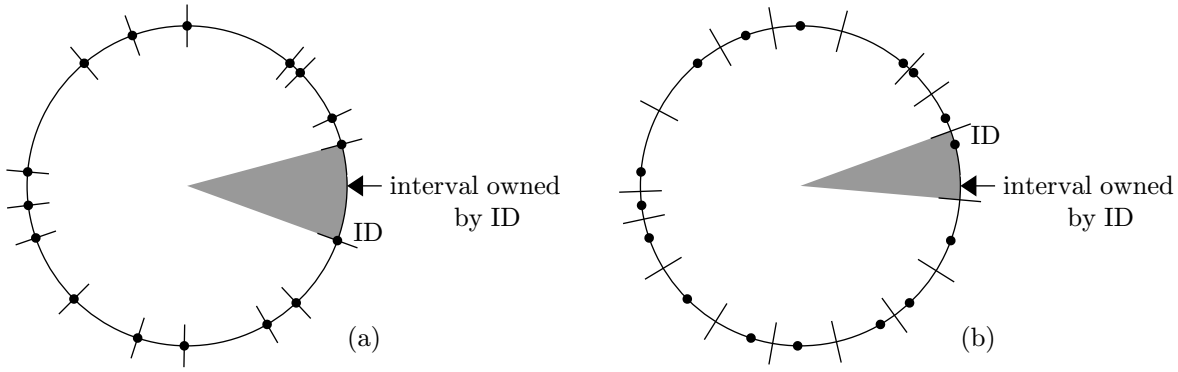

Figure 5. Two ways of partitioning the space. In both (a) and (b), IDs are randomly generated on the perimeter. In (a), IDs own the intervals to their left in clockwise order. In (b), IDs own an interval whose boundaries are determined in some manner, e.g., by virtue of a trie or digital search tree. A downloader or user generates a random number X on the perimeter and picks the owner of the interval of X for its job. The objective is to make all intervals of about equal length, so that all hosts receive about equal traffic.

ID's are represented by their (infinite) binary expansions. Assume, for example, that the n ID's are i.i.d. and uniformly distributed on $[0, 1)$. See, e.g., Ratnasamy et al (2001), Malkhi et al (2002) or Manku et al (2003). Then it is a routine exercise in probability to show that the largest spacing defined by the ID's is asymptotic to $\log n/n$ in probability and that the smallest spacing is $\Theta(1/n^2)$ in probability (see Lévy (1939) or Pyke (1965)). Thus, B_n is asymptotic to $n \log n$ in probability. Adler et al (2003) and Naor and Wieder (2003) implicitly suggest a digital search tree approach. Consider first a trie approach (not considered in those papers): the ID's are considered as strings in a binary trie (with $p_1 = p_2 = 1/2$), and ID's are inserted sequentially as in a digital search tree. The binary expansions of the nodes that are associated with the ID's are the actual ID's used (so, each ID is mapped to another one). In the case of a trie, each leaf is associated with an ID. In the case of a digital serarch tree, each internal node is mapped to an ID. That means that the ID's used have only a finite number of ones in their expansions. If the height of the trie is H_n, and the fill-up level (the number of full levels in the trie) is F_n, then $B_n = 2^{H_n - F_n + O(1)}$. It is known that $H_n = 2 \log_2 n + O(1)$ in probability (Pittel, 1985) and that $F_n = \log_2 n - \log_2 \log_2 n + O(1)$ in probability, so that $B_n = \Theta(n \log n)$ in probability. However,

for a digital search tree, we have $H_n = \log_2 n + O(1)$ in probability, while F_n is basically as for tries. Thus, $B_n = O(\log n)$ in probability. This is essentially the result Adler et al (2003) and Naor and Wieder (2003) were after.

In an attempt to improve this, Dabek et al (2001) proposed attaching $b = \log n$ randomly generated ID's to each host, swamping the interval, and then making the ID assignments to guarantee that $B_n = O(1)$ in probability. However, some discard this solution as too expensive in terms of resources for maintenance.

Abraham et al (2003), Karger and Ruhl (2003) and Naor and Wieder (2003) achieve $B_n = O(1)$ in probability while restricting hosts to one ID. In another approach, Abraham et al (2003) and Naor and Wieder (2003) pick $\log n$ i.i.d. uniform random numbers per host, and assign an ID based on the largest interval these fall into: the largest interval is split in half. A little thought shows that this corresponds to a digital search tree in which $\log n$ independent strings are considered, and only one is selected for insertion, namely the one that would yield a leaf nearest to the root. The fact that both H_n and F_n are now $\log_2 n + O(1)$ in probability yields the result.

Manku (2004) proposed a digital search tree with perfect balancing of all subtrees of size $\log_2 n$ (or something virtually equivalent to that). It also has $B_n = O(1)$ in probability. A similar (but different) view can also be taken for the trie version: start with an ordinary binary trie with the modification (see, e.g., Pittel, 1985) that leaf nodes are mapped to their highest ancestors that have subtrees with $b = \log_2 n$ or fewer leaves. These ancestors are the leaves of the so-called ancestor trie. Construct such a binary b-trie and its ancestor trie from n i.i.d. uniform $[0, 1)$ random numbers. Now, for each ancestor, partition its interval equally by spreading the leaves in its subtree out evenly when associating ID's. We know from the Erdős-Rényi law of large numbers that the maximal k-spacing (with $k = c \log n$) determined by n i.i.d. uniform $[0, 1)$ random numbers is $\Theta(\log n/n)$ in probability (Erdős and Rényi, 1970; Deheuvels, 1985; see also Novak, 1995). This would imply that all ancestors are at level $\log_2 n - \log_2 \log_2 n + O(1)$ in probability, and that all intervals owned by the ID's are $\Theta(1/n)$ in probability, from which $B_n = O(1)$ in probability.

The power-of-two choices can be explored in the present context. If we make an ordinary trie by taking the best of two ID's as described in this paper, and then map ID's to the strings that correspond to the corresponding leaf values, then $H_n = \log_2 n + O(1)$ in probability. However, it is easy to verify that F_n is as for the standard binary trie, so that $B_n = O(\log n)$ in probability. Indeed, the largest gap defined by $2n$ uniform strings on $[0, 1]$ is still $\Theta(\log n/n)$ in probability.

However, the trick suggested by Abraham et al. (2003) and Naor and Wieder (2003) may make $H_n - F_n \leq 2$ with probability tending to one. However, we have two modifications: first, we insist on using tries instead of digital search trees; and secondly, because of the use of tries, we have to modify the selection heuristic as picking the largest interval is not good enough for tries. The next section contains details of the construction of a trie that has both small height and large fill-up level, i.e., a trie that is nearly perfectly balanced.

Finally a bibliographic remark. Tries have been explicitly used in several other ways for distributed hash tables. Balakrishnan et al (2003) use them in the Pastry network. Ramabhadran et al. (2004) define a kind of trie called a prefix hash tree. More recent work on tries in this context was done by Balakrishnan et al. (2005).

4.1 A well-balanced trie for ID management.

Consider the interval $[0,1]$ and let $X_1,\ldots,X_n$ be n independent vectors of $k=\lceil c\log n\rceil$ i.i.d. uniform $[0,1]$ random variables $X_{i,j}$, $1\le i\le n, 1\le j\le k$, where $c>0$ is a constant. We first show that we can pick $X_{1,i_1},\ldots,X_{n,i_n}$ such that the n spacings defined by these random variables on the circular interval $[0,1]$ (the interval wrapped to a unit perimeter circle) are close to $1/n$ with high probability. Denote by M_n the maximal spacing defined by a given selection $X_{1,Z_1},\ldots,X_{n,Z_n}$ where $Z_1,\ldots,Z_n$ are random indices defined in some manner. Let m_n denote the minimal spacing.

LEMMA 6. *Let $\alpha\in(0,1)$ be fixed. Let $c\ge 2/\alpha$. Then there exists a selection $Z_1,\ldots,Z_n$ such that $X_{1,Z_1},\ldots,X_{n,Z_n}$ satisfies, for $n\ge 8$,*

$$\mathrm{P}\left\{\frac{1-\alpha}{n}<m_n\le M_n<\frac{1+\alpha}{n}\right\}\ge 1-\frac{3}{n}.$$

PROOF. Partition $[0,1]$ into n equal intervals I_i, $1\le i\le n$, and let J_i be an interval of length α/n centered within I_i. If our selection is such that each J_i receives exactly one random variable from the n selected, then $(1-\alpha)/n<m_n\le M_n<(1+\alpha)/n$. Thus, we need only be concerned with the event A that there exists a selection vector that guarantees that each J_i is occupied. We use Hall's theorem (1935) (see also Bondy and Murty, 1976). We recall that in a bipartite graph with independent sets A and B, there exists a perfect matching of all elements of A to different elements of B if and only if for all sets $S\subseteq A$, $|N(S)|\ge|S|$, where $N(S)$ is the neighborhood of S in B. Consider a bipartite graph in which the X_i's form one part and the J_j's form the other part. For each $X_{i,\ell}\in J_j$, draw an edge from X_i to J_j. Let N_i be the outdegree of X_i, a binomial $(\lceil c\log n\rceil,\alpha)$ random variable. By Hall's theorem, there does not exist a perfect matching between X_i's and J_j's if and only if for some ℓ, there exists a set of ℓ X_i's that have all their edges end up in a set S with $|S|<\ell$. By the union bound and conditioning on the N_i's,

$$\mathrm{P}\{A^c\}=\mathrm{E}\left\{\mathrm{P}\{A^c|N_1,\ldots,N_n\}\right\}\le\mathrm{E}\left\{\sum_{\ell=1}^{n}\binom{n}{\ell}\binom{n}{\ell-1}\left(\frac{\ell-1}{n}\right)^{N_1+\cdots+N_n}\right\}.$$

For a binomial (ℓ,p) random variable B, we have $\mathrm{E}\{s^B\}=(1-p+ps)^\ell\le\exp(-p(1-s)\ell)$, $s\in(0,1)$. Thus, using the fact that $N_1+\cdots+N_n$ is binomial $(n\lceil c\log n\rceil,\alpha)$,

$$\mathrm{E}\left\{\sum_{\ell=1}^{n/2}\binom{n}{\ell}\binom{n}{\ell-1}\left(\frac{\ell-1}{n}\right)^{N_1+\cdots+N_n}\right\}\le 4^n e^{-\frac{\alpha cn\log n}{2}}\le\left(\frac{4}{n}\right)^n\le\frac{1}{n},n\ge 8.$$

Also,

$$\begin{aligned}\mathrm{E}\left\{\sum_{\ell=n/2}^{n}\binom{n}{\ell}\binom{n}{\ell-1}\left(\frac{\ell-1}{n}\right)^{N_1+\cdots+N_n}\right\}&=\sum_{\ell=0}^{n/2}\binom{n}{\ell}\binom{n}{\ell+1}\mathrm{E}\left\{\left(1-\frac{\ell+1}{n}\right)^{N_1+\cdots+N_n}\right\}\\&\le\sum_{\ell=0}^{n/2}\frac{n^{2\ell+1}}{\ell!(\ell+1)!}\left(1-\frac{\alpha(\ell+1)}{n}\right)^{cn\log n}\\&\le\sum_{\ell=0}^{\infty}\frac{n^{2\ell+1}}{\ell!(\ell+1)!}e^{-\alpha c(\ell+1)\log n}\end{aligned}$$

$$\leq n^{1-\alpha c} \sum_{\ell=0}^{\infty} \frac{n^{\ell(2-\alpha c)}}{\ell!(\ell+1)!}$$
$$\leq \frac{1}{n} \sum_{\ell=0}^{\infty} \frac{1}{\ell!(\ell+1)!}$$
$$\leq \frac{2}{n}.$$

Taken together, we have $\mathbf{P}\{A^c\} \leq 3/n$. □

THEOREM 7. *Let $\alpha \in (0, 1/3)$ be fixed. Let $c = 2/\alpha$. Then there exists a selection $Z_1, \ldots, Z_n$ such that the height H_n and fillup level F_n of the associated trie for $X_{1,Z_1}, \ldots, X_{n,Z_n}$ satisfy, for $n \geq 8$,*

$$\mathbf{P}\{H_n - F_n \leq 2\} \geq 1 - \frac{3}{n}.$$

PROOF. Consider the binary trie formed by the selection $X_{1,Z_1}, \ldots, X_{n,Z_n}$ of Lemma 6. If a potential node at distance d from the root is not realized, then there is a leaf at distance less than d from the root. If that leaf is at distance $d-1$, then only one string in the selection falls in the corresponding interval, which has width $1/2^{d-1}$. Thus, the maximal spacing in the selection is at least half that, or $1/2^d$. Therefore, $1/2^d \leq M_n$. Let F_n be the fill-up level, the distance to the last full level of nodes. We have $F_n = d - 1$ if d is the first level with a missing node. Therefore

$$F_n \geq \log_2(1/M_n) - 1.$$

On the other hand, $H_n = h$, then at distance $h-1$, two strings in the selection visit the same node, and thus, two strings are at distance less than $1/2h - 1$ from each other. Thus, $m_n \leq 1/2^{h-1}$, or

$$H_n \leq \log_2(1/m_n) + 1.$$

If the selection is such that $(1-\alpha)/n < m_n \leq M_n < (1+\alpha)/n$, then we have

$$\lceil \log_2 n - \log_2(1+\alpha) \rceil - 1 \leq F_n \leq H_n \leq \lfloor \log_2 n - \log_2(1-\alpha) \rfloor + 1.$$

We conclude

$$H_n - F_n \leq 2 + \left\lfloor \log_2 \left(\frac{1+\alpha}{1-\alpha} \right) \right\rfloor.$$

If $\alpha < 1/3$, and $c = 2/\alpha$, then the upper bound is 2. □

4.2 An on-line algorithm for super-balancing.

Theorem 7 is an existence theorem, and is appealing since we did not even have to move or transform any of the IDs. Recall that most ID management algorithms (e.g., Manku (2004)) allow hosts to shift their ID's to obtain a better partition. The actual construction may be cumbersome, though. We have not established an algorithm, even for $b = 2$, that can achieve such a balance on-line while not sending many messages in the network.

So, let us find an on-line algorithm that achieves the super-balancing predicted by Theorem 7. Only, the spacings referred to in the definition of B_n now refer to the ID's mapped to the leftmost parts of the intervals in the trie (see below). For each of n hosts, $k = \lceil c \log n \rceil$ i.i.d. uniform $[0, 1]$ potential ID's are generated, $U_i(1), \ldots, U_i(k)$, $1 \le i \le n$. We build an ordinary trie T_n for the binary expansions of $U_1(Z_1), \ldots, U_n(Z_n)$, where the Z_i's are the selections. To define Z_{n+1}, consider the tries $T_{n,j}$ for T_n with $U_{n+1}(j)$ added, $1 \le j \le k$. This is easily done by trying to insert each ID separately. If $D_{n+1}(j)$ is the depth of the leaf of $U_{n+1}(j)$ after insertion into T_n, then

$$Z_{n+1} = \arg\min D_{n+1}(j),$$

where ties are broken randomly. In other words, we pick the ID that at the moment of its birth has the shortest distance to the root of the trie. The actual ID assigned is implicit in the path to the root: it has a common prefix with (the binary expansion of) $U_{n+1}(j)$. This scheme generalizes the best of two-strings greedy algorithm studied earlier in the paper. Unlike with two choices, with $k = c \log n$ choices one does not lose optimality by a greedy construction. We first consider the height H_n.

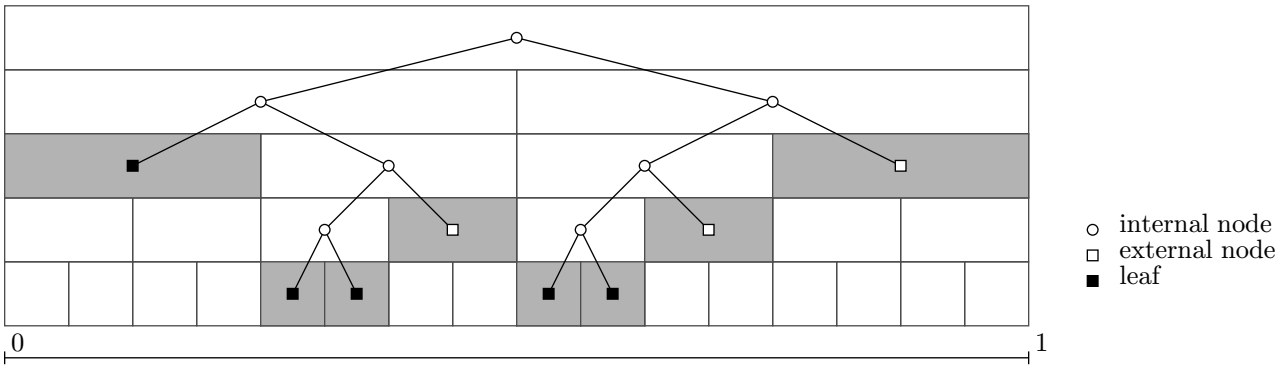

Figure 6. A standard trie for five strings. The correspondence between nodes and intervals in a dyadic partition of the unit interval is shown. The leaf ID assigned is read from the path to the root (0 for left, 1 for right). The external nodes, not normally part of the trie, are shown as well. Together, external nodes and leaves define a partition of the unit interval (shaded boxes). The fill-up level of this tree is one, while the height is four.

THEOREM 8. *Let $c > 1/\log 2$ and $k = \lceil c \log n \rceil$ in the greedy heuristic for assigning IDs. Then*

$$\mathrm{P}\{H_n \ge \log_2 n + 3\} \le n^{1 - c \log 2}.$$

PROOF. Given that $H_{n-1} < h$, we have $H_n \geq h$ if and only if $U_n(Z_n)$ lands within $1/2^{h-1}$ of one of $U_1(Z_1), \ldots, U_{n-1}(Z_{n-1})$. The probability of this is conservatively bounded by

$$\left(\frac{2(n-1)}{2^{h-1}}\right)^k,$$

for if one $U_n(j)$ is further than $1/2^{h-1}$ away from each $U_1(Z_1), \ldots, U_{n-1}(Z_{n-1})$, it will result in a leaf that is less than distance h away from the root. Thus, with $h = \lceil \log_2 n + 3 \rceil$,

$$\mathrm{P}\{H_n \geq h\} \leq \sum_{i=1}^{n} \mathrm{P}\{H_i \geq h | H_{i-1} < h\} \leq n\left(\frac{1}{2}\right)^k \leq n^{1-c\log 2}. \square$$

It is a bit harder to deal with the fill-up level, F_n.

THEOREM 9. *Let $c > (8/5)\log 2$ and $k = \lceil c \log n \rceil$ in the greedy heuristic for assigning IDs. Then*

$$\mathrm{P}\{F_n \leq \log_2 n - 4\} = O(1/\log^2 n).$$

PROOF. We consider vertices (and intervals) in the infinite binary tree level by level, where level refers to distance from the root. Level h thus has 2^h vertices (intervals). We consider two sets of vertices in the trie, leaves (the set is denoted by L), and external nodes that are child nodes of non-leaf nodes in the trie (the set is denoted by E). The nodes E are thus not part of the trie. The intervals that the nodes in L and E together represent form a partition of $[0, 1]$. When a string is inserted in the trie, it visits only one of these intervals. If it belongs to E, then that node in E joins L, and the string is associated with that new leaf. If it belongs to L, then the string occupying that leaf and the new string find the first bit on which they disagree. If this is bit h, then the old leaf node is deleted, and two new leaf nodes are created at level h. In addition, a number of new external nodes may have been created. In particular, if that leaf interval hit by the new string is at level ℓ, then $h = \ell + 1$ with probability $1/2$. Let t_h be the first time that all nodes of E and L are at level h or higher. Let s_h be the first time $\geq t_h$ such that all nodes of E (if any) are at level strictly higher than h. If E is empty at time t_h, then $s_h = t_h$. Let $\mathcal{F}_h$ denote the σ-algebra generated by all $U_i(j)$, $i \leq t_h$. Let $\mathcal{G}_h$ denote the σ-algebra generated by all $U_i(j)$, $i \leq s_h$. We will find upper bounds for $s_h - t_h$ and $t_{h+1} - s_h$ for all h, by bounding the time needed to clear level h, first from leaves, and then from external nodes. In what follows, we will use the duality

$$\mathrm{P}\{F_n \leq h\} = \mathrm{P}\{s_h > n\}.$$

To bound $s_h - t_h$, we start with a number of external node intervals that is anywhere between 0 and 2^h. Consider the worst scenario, where at time t_h, there are exactly 2^ℓ such nodes. The objective, as in the coupon collector problem, is to hit each interval with a string. Given that there are ℓ external nodes left, the probability of eliminating one of them by our greedy algorithm (when inserting one new node) is

$$q_{\ell,h} \stackrel{\text{def}}{=} 1 - \left(1 - \frac{\ell}{2^h}\right)^k.$$

If the waiting time until this happens is $T_{\ell,h}$, then we have

$$\mathrm{P}\{T_{\ell,h} \geq t\} = (1 - q_{\ell,h})^{t-1} \leq e^{1-q_{\ell,h}t}, t \geq 1.$$

Thus, $T_{\ell,h}$ is stochastically dominated ($\overset{(\mathrm{st})}{\leq}$) by $1+Z_{\ell,h}/q_{\ell,h}$, where $Z_{\ell,h}$ is an exponential random variable. Thus, given $\mathcal{F}_h$, and letting all Z's in the sum below be i.i.d. exponential random variables,

$$s_h - t_h \overset{(\mathrm{st})}{\leq} \sum_{\ell=1}^{2^h} \left(1 + \frac{Z_{\ell,h}}{q_{\ell,h}}\right) = 2^h + \sum_{\ell=1}^{2^h} \frac{Z_{\ell,h}}{q_{\ell,h}}.$$

Having cleared level h of all external nodes, we proceed to clear it from leaves. Note that a leaf is eliminated if one of the leaf intervals gets hit in a way that would induce the creation of two leaves at level $h+1$ (the conditional probability of this is $1/2$). Arguing as for the external nodes, we see that given ℓ leaf intervals, elimination happens with probability at least

$$p_{\ell,h} \overset{\mathrm{def}}{=} 1 - \left(1 - \frac{\ell}{2^{h+1}}\right)^k.$$

If the waiting time until this happens is $T'_{\ell,h}$, then we have

$$\mathrm{P}\{T'_{\ell,h} \geq t\} = (1-p_{\ell,h})^{t-1} \leq e^{1-p_{\ell,h}t}, t \geq 1.$$

Thus, $T'_{\ell,h}$ is stochastically dominated by $1 + Z'_{\ell,h}/p_{\ell,h}$, where $Z'_{\ell,h}$ is an exponential random variable. Thus, given $\mathcal{G}_h$, and letting all Z''s in the sum below be i.i.d. exponential random variables,

$$t_{h+1} - s_h \overset{(\mathrm{st})}{\leq} \sum_{\ell=1}^{2^h} \left(1 + \frac{Z'_{\ell,h}}{p_{\ell,h}}\right) = 2^h + \sum_{\ell=1}^{2^h} \frac{Z'_{\ell,h}}{p_{\ell,h}}.$$

We use duality. Setting $s_0 = 1$, we have, using the above stochastic domination bounds, and $p_{\ell,h-1} = q_{\ell,h}$, a simple coupling argument shows that

$$\sum_{r=1}^{h}(s_r - t_r + t_r - s_{r-1}) \overset{(\mathrm{st})}{\leq} \sum_{r=1}^{h} \left(2^r + 2^{r-1} + \sum_{\ell=1}^{2^r} \frac{Z_{\ell,r}}{q_{\ell,r}} + \sum_{\ell=1}^{2^{r-1}} \frac{Z'_{\ell,r-1}}{q_{\ell,r}}\right),$$

and therefore

$$\begin{aligned}
\mathrm{P}\{F_n \leq h\} &= \mathrm{P}\{s_h > n\} \\
&= \mathrm{P}\left\{\sum_{r=1}^{h}(s_r - t_r + t_r - s_{r-1}) > n-1\right\} \\
&\leq \mathrm{P}\left\{\sum_{r=1}^{h} \left(2^r + 2^{r-1} + \sum_{\ell=1}^{2^r} \frac{Z_{\ell,r}}{q_{\ell,r}} + \sum_{\ell=1}^{2^{r-1}} \frac{Z'_{\ell,r-1}}{q_{\ell,r}}\right) > n-1\right\} \\
&\leq \mathrm{P}\left\{2^{h+1} + 2^h - 3 + 2\sum_{r=1}^{h}\sum_{\ell=1}^{2^r} \frac{Z_{\ell,r}}{q_{\ell,r}} > n-1\right\} \\
&\leq \mathrm{P}\left\{3 \times 2^h + X > n+2\right\} \\
&\leq \mathrm{P}\left\{X - \mathrm{E}\{X\} > n + 2 - 3 \times 2^h - \mathrm{E}\{X\}\right\} \\
&\leq \frac{\mathrm{V}\{X\}}{\left(n + 2 - 3 \times 2^h - \mathrm{E}\{X\}\right)^2},
\end{aligned}$$

by Chebyshev's inequality, where all the Z's above are i.i.d.,

$$X = 2\sum_{r=1}^{h}\sum_{\ell=1}^{2^r}\frac{Z_{\ell,r}}{q_{\ell,r}},$$

and we assume that $n + 2 - 3 \times 2^h - \mathrm{E}\{X\} > 0$.

$$\mathrm{E}\{X\} = 2\sum_{r=1}^{h}\sum_{\ell=1}^{2^r}\frac{1}{q_{\ell,r}},$$

and

$$\mathrm{V}\{X\} = 4\sum_{r=1}^{h}\sum_{\ell=1}^{2^r}\frac{1}{q_{\ell,r}^2}.$$

We use the following lower bounds:

$$q_{\ell,h} \stackrel{\text{def}}{=} 1 - \left(1 - \frac{\ell}{2^h}\right)^k \geq 1 - e^{-k\ell/2^h} \geq \frac{e-1}{e} \times \min\left(1, \frac{k\ell}{2^h}\right).$$

Thus,

$$\begin{aligned}
\mathrm{E}\{X\} &= 2\sum_{r=1}^{h}\sum_{\ell=1}^{2^r}\frac{1}{q_{\ell,r}} \\
&\leq 2\sum_{r=1}^{h}\sum_{\max(1,2^r/k)\leq\ell\leq 2^r}\frac{1}{q_{\ell,r}} + 2\sum_{r=1}^{h}\sum_{1\leq\ell\leq 2^r/k}\frac{1}{q_{\ell,r}} \\
&\leq \sum_{r=1}^{h}\frac{2^{r+1}e}{e-1} + \frac{2e}{e-1}\sum_{r=1}^{h}\sum_{1\leq\ell\leq 2^r/k}\frac{2^r}{k\ell} \\
&\leq 2^{h+3} + \frac{2e}{e-1}\sum_{r=1}^{h}\frac{2^r(1 + r\log(2))}{k} \\
&\leq 2^{h+3} + \frac{2^{h+3}(1 + h\log(2))}{k}.
\end{aligned}$$

Similarly,

$$\begin{aligned}
\mathrm{V}\{X\} &\leq 4\sum_{r=1}^{h}\sum_{\max(1,2^r/k)\leq\ell\leq 2^r}\frac{1}{q_{\ell,r}^2} + 4\sum_{r=1}^{h}\sum_{1\leq\ell\leq 2^r/k}\frac{1}{q_{\ell,r}^2} \\
&\leq \sum_{r=1}^{h}\frac{2^{r+2}e^2}{(e-1)^2} + \frac{4e^2}{(e-1)^2}\sum_{r=1}^{h}\sum_{1\leq\ell\leq 2^r/k}\frac{2^{2r}}{k^2\ell^2} \\
&\leq 2^{h+5} + \frac{8\lambda^2}{3}\sum_{r=1}^{h}\frac{2^{2r}}{k^2} \\
&\leq 2^{h+5} + \frac{2^{2h+6}}{k^2}.
\end{aligned}$$

Putting everything together, we see that when $h = \lfloor \log_2 n - R \rfloor$, $R \in \mathcal{N}$, we have

$$\mathrm{P}\{F_n \leq h\} \leq \frac{2^{h+5} + \frac{2^{2h+6}}{k^2}}{\left(n + 2 - 3 \times 2^h - 2^{h+3} - \frac{2^{h+3}(1+h\log(2))}{k}\right)^2} = O\left(\frac{1}{k^2}\right)$$

if $(3 + 8 + 8\log(2)/c)/2^R < 1$. We can always find such a c when $R \geq 4$. In fact, for $R = 4$, it suffices to set $c > 8 \log 2/5$. □

In conclusion, $H_n - F_n \leq 7$ with probability tending to one when $c > 1/\log 2$. This implies that $B_n = O(1)$ in probability when spacings are defined with respect to leftmost points of leaf intervals. Finally, the trie scheme proposed here assumes that one knows n, while in distributed networks, n is unknown. However, one can estimate n by 2^D, where D is the depth of insertion of a random string in the trie: since all depths are within $O(1)$ of $\log_2 n$, the estimate is off by a constant factor only. One can thus extend the method in this manner by replacing $k = c \log n$ throughout by cD where D is the depth of insertion of a random string.

References.

I. Abraham, B. Awerbuch, Y. Azar, Y. Bartal, D. Malkhi, and E. Pavlov, "A generic scheme for building overlay networks in adversarial scenarios," in: *Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS 2003)*, 2003.

M. Adler, E. Halperin, R. M. Karp, and V. V. Vazirani, "A stochastic process on the hypercube with applications to peer-to-peer networks. ," in: *Proceedings of the 35th ACM Symposium on Theory of Computing (STOC 2003)*, pp. 575–584, 2003.

Y. Azar, A. Z. Broder, A. R. Karlin, and E. Upfal, "Balanced allocations (extended abstract)," in: *Proceedings of the 26th ACM Symposium on the Theory of Computing*, pp. 593–602, 1994.

Y. Azar, A. Z. Broder, A. R. Karlin, and E. Upfal, "Balanced allocations," *SIAM Journal on Computing*, vol. 29, pp. 180–200, 1999.

H. Balakrishnan, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica, "Looking up data in P2P systems," *Communications of the ACM*, vol. 1946, pp. 43–48, 2003.

H. Balakrishnan, S. Shenker, and M. Walfish, "Peering peer-to-peer providers," in: *Peer-to-Peer Systems IV, 4th International Workshop, IPTPS 2005, Ithaca, NY*, (edited by M. Castro, R. van Renesse (eds)), vol. 3640, pp. 104–114, Lecture Notes in Computer Science, Springer-Verlag, 2005.

J. A. Bondy and U. S. R. Murty, *Graph Theory with Applications*, North-Holland, Amsterdam, 1976.

S. Boucheron, G. Lugosi, and P. Massart, "A sharp concentration inequality with applications in random combinatorics and learning," *Random Structures and Algorithms*, vol. 16, pp. 277–292, 2000.

S. Boucheron, G. Lugosi, and P. Massart, "Concentration inequalities using the entropy method," *Annals of Probability*, vol. 31, pp. 1583–1614, 2003.

K. L. Chung and P. Erdös, "On the application of the Borel-Cantelli lemma," *Transactions of the American Mathematical Society*, vol. 72, pp. 179–186, 1952.

E. G. Coffman and J. Eve, "File structures using hashing functions," *Communications of the ACM*, vol. 13, pp. 427–436, 1970.

I. Csiszár, "On generalized entropy," *Studia Scientiarium Mathematicarum Hungarica*, vol. 4, pp. 401–419, 1969.

A. Czumaj and V. Stemann, "Randomized Allocation Processes," in: *Proceedings of the 38th IEEE Symposium on Foundations of Computer Science (FOCS'97), October 19-22, 1997, Miami Beach, FL*, pp. 194–203, 1997.

F. Dabek, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica, "Wide-area cooperative storage with CFS," in: *Proceedings of the 18th ACM Symposium on Operating Systems Principles (SOSP 2001)*, pp. 202–215, 2001.

P. Deheuvels, "On the Erdös-Rényi theorem for random fields and sequences and its relationships with the theory of runs and spacings," *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, vol. 70, pp. 91–115, 1985.

A. Dembo and O. Zeitouni, *Large Deviations Techniques and Applications (Second Edition)*, Springer-Verlag, New York, 1998.

L. Devroye, "Laws of large numbers and tail inequalities for random tries and Patricia trees," *Journal of Computational and Applied Mathematics*, vol. 142, pp. 27–37, 2002.

P. Erdös and A. Rényi, "On a new law of large numbers," *J. Anal. Math.*, vol. 22, pp. 103–111, 1970.

E. H. Fredkin, "Trie memory," *Communications of the ACM*, vol. 3, pp. 490–500, 1960.

P. Hall, "On representatives of subsets," *Journal of the London Mathematical Society*, vol. 10, pp. 26–30, 1935.

P. Jacquet and M. Régnier, "Trie partitioning process: limiting distributions," in: *CAAP 86*, (edited by P. Franchi-Zannettacci), vol. 214, pp. 196–210, Lecture Notes in Computer Science, Springer-Verlag, Berlin, 1986.

S. Janson, T. Luczak, and A. Rucinski, *Random Graphs*, Wiley-Interscience, New York, 2000.

D. R. Karger and M. Ruhl, "New algorithms for load balancing in peer-to-peer systems," IRIS Student Workshop, 2003.

D. E. Knuth, *The Art of Computer Programming, Vol. 3 : Sorting and Searching*, Addison-Wesley, Reading, Mass., 1973.

A. G. Konheim and D. J. Newman, "A note on growing binary trees," *Discrete Mathematics*, vol. 4, pp. 57–63, 1973.

P. Levy, "Sur la division d'un segment par des points choisis au hasard," *Comptes Rendus Acad. Sci . Paris*, vol. 208, pp. 147–149,

D. Malkhi, M. Naor, and D. Ratajczak, "Viceroy: A scalable and dynamic emulation of the butterfly," in: *Proceedings of the 21st ACM Symposium on Principles of Distributed Computing (PODC 2002)*, pp. 183–192, 2002.

C. L. Mallows, "An inequality involving multinomial probabilities," *Biometrika*, vol. 55, pp. 422–424, 1968.

G. S. Manku, M. Bawa, and P. Raghavan, "Symphony: Distributed hashing in a small world," in: *Proceedings of the Fourth USENIX Symposium on Internet Technologies and Systems (USITS 2003)*, pp. 127–140, 2003.

G. S. Manku, "Balanced binary trees for ID management and load balance in distributed hash tables," in: *23rd ACM Symposium on Principles of Distributed Computing (PODC 2004)*, pp. 197–205, 2004.

D. R. Morrison, "PATRICIA — Practical Algorithm To Retrieve Information Coded in Alphanumeric," *Journal of the ACM*, vol. 15, pp. 514–534, 1968.

M. Naor and U. Wieder, "Novel architectures for P2P applications: The continuous-discrete approach," in: *Proceedings of the 15th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA 2003)*, pp. 50–59, 2003.

S. Yu. Novak, "On the Erdös-Rényi maximum of partial sums," *Theory of Probability and its Applications*, vol. 42, pp. 254–270, 1995.

R. Pagh and F. F. Rodler, "Cuckoo hashing," BRICS Report Series RS-01-32, Department of Computer Science, University of Aarhus, 2001.

B. Pittel, "Asymptotical growth of a class of random trees," *Annals of Probability*, vol. 13, pp. 414–427, 1985.

B. Pittel, "Path in a random digital tree: limiting distributions," *Advances in Applied Probability*, vol. 18, pp. 139–155, 1986.

R. Pyke, "Spacings," *Journal of the Royal Statistical Society*, vol. 27, pp. 395–436, 1965.

S. Ramabhadran, S. Ratnasamy, J. M. Hellerstein, and S. Shenker, "Prefix hash tree: An indexing data structure over distributed hash tables," IRB Technical Report , 2004.

S. Ratnasamy, P. Francis, M. Handley, and R. M. Karp, "A scalable content-addressable network," in: *Proceedings of the ACM SIGCOMM 2001*, pp. 161–172, 2001.

M. Régnier and P. Jacquet, "New results on the size of tries," *IEEE Transactions on Information Theory*, vol. IT-35, pp. 203–205, 1989.

A. Rényi, "On the dimension and entropy of probability distributions," *Acta Mathematica Academiae Sci. Hungarica*, vol. 10, pp. 193–215, 1959.

W. Szpankowski, "Some results on V-ary asymmetric tries," *Journal of Algorithms*, vol. 9, pp. 224–244, 1988.

W. Szpankowski, "A characterization of digital search trees from the successful search viewpoint," *Theoretical Computer Science*, vol. 85, pp. 117–134, 1991.

W. Szpankowski, "On the height of digital trees and related problems," *Algorithmica*, vol. 6, pp. 256–277, 1991.

W. Szpankowski, *Average Case Analysis of Algorithms on Sequences*, Springer-Verlag, New York, 2001.

R. E. Tarjan, *Data Structures and Network Algorithms*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1983.