Acta Informatica 24, 277-298 (1987)

Acta Informatica

Branching Processes in the Analysis of the Heights of Trees

L. Devroye*

School of Computer Science, McGill University, 805 Sherbrooke Street West,
Montreal PQ, Canada H3A 2K6

Summary. It is shown how the theory of branching processes can be applied in the analysis of the expected height of random trees. In particular, we will study the height of random binary search trees, random $k-d$ trees, quadtrees and union-find trees under various models of randomization. For example, for the random binary search tree constructed from a random permutation of $1, \ldots, n$, it is shown that $H_n/(c\log(n))$ tends to 1 in probability and in the mean as $n \to \infty$, where H_n is the height of the tree, and $c = 4.31107\ldots$ is a solution of the equation $c\log\left(\frac{2e}{c}\right) = 1$. In addition, we show that $H_n - c\log(n) = O(\sqrt{\log(n)\operatorname{loglog}(n)})$ in probability.

1. Introduction

The purpose of this note is to introduce a simple technique which can be used in the study of the height of some types of random trees. Heights of random trees can be studied by means of a detailed analysis of the generating function of the number of trees with height not exceeding a given value: see for example [9, 18, 34, 29] for several kinds of ordered trees, and the survey papers by [13] for other trees including binary trees. The height of random tries is studied in [11, 14, 15, 24-26, 28]. This approach has been particularly successful for trees in which randomness is defined by making each of the possible shapes equally likely.

For the random binary search tree, under the standard random permutation model for the data, a veriety of techniques have been used in the analysis of various quantities. For example, the result that the expected depth of the n-th node in an n-node tree is asymptotic to $2\log(n)$ can be found in most textbooks on data structures and algorithms [1, 2, 19, 20]. The expected

CR Categories: 3.74, 5.25, 5.5

* Research of the author was sponsored by NSERC Grant A3456 and by FCAC Grant EQ-1678

height is known to be asymptotic to $4.31107\ldots\log(n)$ [30, 11, 16]. In this note we would like to provide the readers with a simple general technique for studying the expected height of trees including the expected height of the random binary search tree. The method is based upon simple properties of branching processes. This is a situation in which the techniques seem more important than the results. We will briefly indicate how they can be applied to other trees too such as random multidimensional search trees and union-find trees, but we do not imply anything about the usefulness of these particular trees. Furthermore, there is a severe penalty for the added simplicity: we will only be able to obtain the first term in the asymptotic expansions. In addition, several new results will be proved. For example, if H_n is the height of a random tree on n nodes, then

$$\frac{H_n}{c\log(n)}\to 1 \text{ in probability}$$

and

$$E(H_n)\sim c\log(n),$$

where c is a given constant. Note that "in probability" is to be taken in the standard probability theoretical sense. In combinatorial jargon, this corresponds to "almost all". Thus, the first statement can be rephrased as follows: for all $\varepsilon>0$, almost all permutations of $1,\ldots,n$ yield trees of height H_n between $(c-\varepsilon)\log(n)$ and $(c+\varepsilon)\log(n)$ as $n\to\infty$.

Occasionally, there will be explicit probability bounds with various uses, but the method is generally not designed to yield unlimited asymptotic accuracy in expansions of quantities such as $E(H_n)$. It is also not designed to solve the problem of the asymptotic distribution of H_n, which in the case of the random binary search tree appears to be rather difficult to obtain by any method. Nevertheless, we will illustrate how some possibly suboptimal but useful tail estimates can be constructed for the difference $H_n-c\log(n)$.

2. Some Results from Branching Processes

In this section we will define some terms and recall fundamental properties of branching processes. For proofs, the reader is referred to [17]. The particular branching process of use to us is the Galton-Watson process, which can be defined as follows. Assume that we have a population at time (generation) 0 of one element. Each element in each generation is replaced in the next generation by a random number (say, X) of elements with given distribution:

$$P(X=i)=p_i \qquad (i=0,1,2,\ldots).$$

These replacements are independent of each other, and the probability vector $p_0, p_1, \ldots$ is fixed. Thus, at time 1, we have X elements, and at time $2\sum_{j\leq X} X_j$ where $X, X_1, \ldots$ are i.i.d. random variables. The total population size at time n is Z_n. The first result needed here concerns the extinction or survival of the species. The most important parameters governing the asymptotic behavior of Z_n are the mean μ of X and the variance σ^2 of X.

Theorem 1. *The extinction probability theorem. Let Z_n be a Galton-Watson process defined by a probability vector with mean μ. Then,*

$$P(\lim_{n\to\infty} Z_n=0)=1-P(\lim_{n\to\infty} Z_n=\infty)=q$$

where $q=1$ if $\mu<1$ or $\mu=1$, $p_0>0$ and $0\leqq q<1$ otherwise.

The parameter q, the probability of eventual extinction, also characterizes the branching process. It is 0 only if $p_0=0$. At a crucial point, we will need an inequality connecting q with the distribution of X. We have:

Theorem 2. *In a Galton-Watson process with parameters $\mu>1$ and $\sigma^2<\infty$, Z_n/μ^n tends in distribution to a nonnegative random variable W where*

$$E(W)=1,$$

$$\mathrm{Var}(W)=\frac{\sigma^2}{\mu^2-\mu},$$

$$P(W=0)=q.$$

Consequently,

$$1-q\geqq\frac{\mu^2-\mu}{\sigma^2+\mu^2-\mu}.$$

If only $p_0, p_1, \dots, p_M$ are positive, then

$$1-q\geqq\frac{\mu-1}{M}.$$

Proof of Theorem 2. Only the last two inequalities require a few words of explanation. By the one-sided Cantelli inequality (see e.g. [27], p. 145) or [32],

$$P(W=0)\leqq\frac{\mathrm{Var}(W)}{\mathrm{Var}(W)+E^2(W)}=\frac{\mathrm{Var}(W)}{\mathrm{Var}(W)+1}.$$

It is important to note that the process grows roughly speaking as $W\mu^n$ where W is a random variable, which is strictly positive when $p_0=0$, and which is positive with probability $1-q$ in all cases.

The last inequality can be obtained as follows. Let

$$f(s)=\sum_{i=0}^{M} p_i s^i$$

be the generating function for the branching process, where $s\in[0, 1]$. Then the obvious inequality

$$p_i s^i\leqq\left(1-\frac{i}{M}\right)p_i+\frac{i}{M}\,p_i s^M$$

implies

$$f(s) \leqq \sum_{i=0}^{M} \left(1-\frac{i}{M}\right) p_i + \sum_{i=0}^{M} \frac{i}{M} p_i s^M = 1-\frac{\mu}{M}+\frac{\mu}{M} s^M.$$

It is known that q is equal to the value of s for which $f(s)=s$. Thus the solution of

$$1-\frac{\mu}{M}+\frac{\mu}{M} s^M = s$$

provides an upper bound for q. To obtain a good approximation of this value, we first find the point u where the derivative of the upper bound is 1, i.e.

$$u = \mu^{\frac{-1}{M-1}}.$$

We certainly have $q \leqq u$, and in view of the convex nature of the polynomial in s, we also have $q \leqq 1-\frac{\mu}{M}+\frac{\mu}{M} u^M$, which gives us

$$1-q \geqq \frac{\mu}{M}(1-u^M) \geqq \frac{\mu-1}{M}. \quad \square$$

3. The Random Binary Search Tree

We define a random binary search tree as a binary search tree constructed from a random permutation of $1, \dots, n$, where all permutations are equally likely, and insertion is carried out in the standard manner. The height H_n is the number of nonempty levels minus one. In this section, we establish a crucial link between random binary search trees and trees of random variables. The root of a random binary search tree splits the $n-1$ remaining elements in two sets, one for each subtree, where the size of the left subtree is distributed as $\lfloor nU \rfloor$, and U is a uniform $[0,1]$ random variable. Ignoring the floor function for a second, the subtrees are roughly distributed as $(nU, n(1-U))$. Each of the subtrees can be split in a similar fashion, requiring this time two new uniform $[0,1]$ random variables. This process can be repeated at all levels. In this manner we obtain a tree of products of uniform random variables. More formally, let T_k be a complete binary tree with k full levels of edges. The total number of edges is $2^1+2^2+\dots+2^k=2^{k+1}-2$. We will use the symbol p for a path from root to leaf (there are 2^k such paths in T_k). Consider all edges pairwise in level order and from left to right, and associate with each pair an independent random vector distributed as $(U, 1-U)$ where U is uniformly distributed on $[0,1]$. For a path p in T_k, we define the path value $V(p)$ by

$$V(p) = \prod_{i \in p} X_i$$

where X_i is the uniform random variable associated with edge i on path p. Thus, there are 2^k different path values $V(p)$. The difficulty arises from the fact that the $V(p)$'s are heavily correlated. We are in particular interested in the maximal value of the $V(p)$'s. Let us define

$$Z_k \triangleq \begin{cases} 1 & (k=0) \\ \max\limits_p V(p) & (k>0) \end{cases}.$$

Theorem 3. *Let $n \geqq 1$, $k \geqq 1$ be given integers. Then the height H_n of a random binary search tree on n nodes is related to the random variables Z_k via the following inequalities:*

$$P\left(Z_k \geqq \frac{1+k}{n}\right) \leqq P(H_n \geqq k) \leqq P\left(Z_k \geqq \frac{1}{n}\right) \leqq 2^k P\left(\prod_{i=1}^{k} U_i \geqq \frac{1}{n}\right),$$

where $U_1, \ldots, U_k$ are i.i.d. uniform $[0, 1]$ *random variables.*

Theorem 3 is proved in [11] by induction on k. It suffices to note that sizes of the subtrees of the root of an n-node tree are at most nU, $n(1-U)$ and at least $nU-1$, $n(1-U)-1$, where U is a uniform $[0, 1]$ random variable, and to apply this observation in the induction proof. The rightmost inequality is a Bonferroni inequality (see e.g. [22]) for the union of events.

The height H_n can be studied via the inequalities of Theorem 3. In particular, probabilistic upper bounds for H_n are easily obtained because the product of k independent uniform random variables has a simple explicitly known distribution. Unfortunately, since k will typically be of the order of $\log(n)$, the inequalities of Theorem 3 are not strong enough to study the second term of any asymptotic theory for H_n, because about $\log\log(n)$ is lost in the process. This is the price paid for the transition from H_n to Z_k.

In Theorem 5 below, we obtain the announced upper bounds. We are particularly careful in our estimation because the inequalities obtained there will be useful to obtain at least a rough idea of the size of the second (i.e. the $\log\log(n)$) term. Lower bounds for H_n follow by the careful application of Theorems 1–2 on branching processes. They are presented in Theorems 6 and 7. Collecting all of this, we will prove the following result:

Theorem 4. *Let H_n be the height of a random binary search tree. Then*

$$\frac{H_n}{c\log(n)} \to 1 \text{ in probability}$$

and

$$E(H_n) \sim c\log(n)$$

where $c = 4.31107\ldots$ is the solution of $c\log\left(\frac{2e}{c}\right) = 1$; $c \geqq 2$.

Theorem 5. *Let H_n be the height of a random binary search tree on n nodes. For $k \geqq \log(n)$, $n \geqq 1$,*

$$P(H_n \geqq k) \leqq \frac{(2\log(n))^k}{nk!\left(1-\frac{\log(n)}{k+1}\right)} \sim \left(\frac{2e\log(n)}{k}\right)^k \frac{1}{n\sqrt{2\pi k}\left(1-\frac{\log(n)}{k+1}\right)}.$$

Let c be as in Theorem 4, *and let d be defined by* $d=\frac{c}{2(1-c)}$. *Then,*

$$\lim_{M\to\infty}\limsup_{n\to\infty} P(H_n \geqq c\log(n)+d\operatorname{loglog}(n)+M)=0.$$

In particular, for all $\varepsilon>0$,

$$\lim_{n\to\infty} P(H_n \geqq (c+\varepsilon)\log(n))=0,$$

$$\lim_{n\to\infty} P\left(\frac{H_n-c\log(n)}{\operatorname{loglog}(n)}\geqq d+\varepsilon\right)=0,$$

$$\limsup_{n\to\infty}\frac{E(H_n)-c\log(n)}{\operatorname{loglog}(n)}\leqq d.$$

Proof of Theorem 5. From the upper bound in Theorem 3, and the fact that a product of k independent uniform random variables is gamma (k) distributed, we see that

$$P(H_n \geqq k)\leqq 2^k P(G_k \leqq \log(n)),$$

where G_k is a gamma (k) random variable. By partial integration of the gamma density, it is not difficult to verify that for $y>0$,

$$P(G_k\leqq y)=\left(\frac{y^k}{k!}e^{-y}\right)\left(1+\frac{y}{k+1}+\frac{y^2}{(k+1)(k+2)}+\frac{y^3}{(k+1)(k+2)(k+3)}+\dots\right).$$

The right-hand-side is at most equal to

$$\frac{\left(\frac{y^k}{k!}e^{-y}\right)}{1-\frac{y}{k+1}}$$

at least when $y<k+1$. This upper bound is asymptotically optimal when $y=O(k)$, and $k\to\infty$. This is precisely the domain we are interested in, and it is thus not necessary to look any further for better bounds. The first inequality of Theorem 5 follows by replacing y by $\log(n)$, and by applying Stirling's approximation to $k!$ $\left(\text{i.e., } k!\sim\left(\frac{k}{e}\right)^k\sqrt{2\pi k}\right)$.

Next, replace k by $c\log(n)+d\operatorname{loglog}(n)+x$ where x is a large number at least equal to M. Then, in view of

$$k!\geqq\left(\frac{k}{e}\right)^k\sqrt{2\pi k},$$

it is clear that, if we use C for some finite constant,

$$\begin{aligned}
\log(P(H_n \geqq k)) &\leqq k \log\left(\frac{2e\log(n)}{k}\right) - \log(n) - \tfrac{1}{2}\log(k) + C \\
&= k \log\left(\frac{2e}{c}\right) - k \log\left(1 + \frac{d \operatorname{loglog}(n) + x}{c \log(n)}\right) - \log(n) - \tfrac{1}{2}\log(k) + C \\
&= (\log(n))\left(c \log\left(\frac{2e}{c}\right) - 1\right) + (d \operatorname{loglog}(n) + x)\left(\log\left(\frac{2e}{c}\right) - 1\right) \\
&\quad - \tfrac{1}{2}\operatorname{loglog}(n) + C + o(1) \\
&= (\operatorname{loglog}(n))\left(d\,\frac{1-c}{c} - \frac{1}{2}\right) + x\,\frac{1-c}{c} + C + o(1) \\
&= x\,\frac{1-c}{c} + C + o(1) \\
&\leqq M\,\frac{1-c}{c} + C + o(1).
\end{aligned}$$

The right-hand-side of this chain tends to $-\infty$ if we first let n tend to ∞, and then M. This concludes the second statement of Theorem 5. The only troublesome part left in the theorem concerns the statement about $E(H_n)$. Let us rewrite the first inequality of the theorem as follows:

$$P(H_n \geqq k) \leqq \phi_k.$$

Note that for $k \geqq \log(n)$,

$$\phi_{k+1} \leqq \left(\frac{2\log(n)}{k+1}\right)\phi_k.$$

Define $K = c\log(n) + (d+\varepsilon)\operatorname{loglog}(n)$ where $\varepsilon > 0$. Then,

$$\begin{aligned}
E(H_n) &= \sum_{k=1}^{\infty} P(H_n \geqq k) \\
&\leqq K + \sum_{k=K+1}^{\infty} P(H_n \geqq k) \\
&\leqq K + \sum_{k=K+1}^{\infty} \phi_k \\
&\leqq K + \phi_{K+1} \sum_{k=0}^{\infty} \left(\frac{2\log(n)}{K+1}\right)^k \\
&= K + \frac{\phi_{K+1}}{1 - \left(\frac{2\log(n)}{K+1}\right)} \\
&= K + o(1)
\end{aligned}$$

by the first part of the theorem. □

Let us now get to work on the most difficult part, a lower bound for H_n. First, we will prove the simplest form, namely that H_n must be at least $c\log(n)$. Later, in Theorem 7, a more refined argument will be presented which will give us an idea of how close H_n is to $c\log(n)$. In the proof of Theorem 7, the different parameters found in the proof of Theorem 6 will be optimized, and without new ingredients in the proof, no better estimates of $H_n - c\log(n)$ can be obtained. Since this balancing of parameters is tedious and uninspiring, Theorem 7 and its proof are put in a separate section which can be skipped without loss. If the reader wants to see how Theorems 1, 2 are applied in the derivation of a lower bound, he must of course take a close look at the proof of Theorem 6, which is based in part upon a method developed in [6, 7].

Theorem 6. *Let c be as in Theorem 4, and let H_n be the height of a random binary search tree on n nodes. Then, for all $\varepsilon > 0$,*

$$\lim_{n\to\infty} P(H_n < (c-\varepsilon)\log(n)) = 0.$$

Also,

$$\liminf_{n\to\infty} \frac{E(H_n)}{c\log(n)} \geqq 1.$$

Proof of Theorem 6. We will use the shorthand notation k for $\lfloor (c-\varepsilon)\log(n)\rfloor$. This integer k needs to be decomposed as follows:

$$k = r + l\left(\frac{k-r}{l}\right).$$

Here r and l are integers, which in Theorem 6 take fixed values. The value of $\left(\frac{k-r}{l}\right)$ must also be integer. In Theorem 7, the values of r and l will depend upon n. The least crucial integer is r, so after having fixed l, the integer r can be adjusted to make $k-r$ a multiple of l. For the time being, we can thus take $r \in [l, 2l)$. The complete tree T_k of Theorem 3 is split up as follows: consider a complete top tree T_r having 2^r leaves. Each leaf is the root of a complete subtree T_{k-r}. From Theorem 3, we recall that

$$P(H_n \geqq k) \geqq P\left(Z_k \geqq \frac{1+k}{n}\right)$$

which in turn is at least equal to the probability that at least one of the subtrees rooted at those leaves of T_r for which $V(p) \geqq 4^{-r}$ has $Z_{k-r} \geqq 4^r\left(\frac{1+k}{n}\right)$. Assume for the time being that we have shown that the probability that for one fixed subtree T_{k-r}, $Z_{k-r} \geqq 4^r\left(\frac{1+k}{n}\right)$ is at least equal to $1-q$. Then, if N is the number of paths in T_r for which $V(p) \geqq 4^{-r}$, it is clear that for any integer s,

$$P(H_n \geqq k) \geqq E(1-q^N) \geqq (1-q^s)\,P(N \geqq s).$$

Consider now the following branching process: start with a population of size 1. The number of children of each element is 1 or 2 with equal probability. Then, $\mu=\frac{3}{2}$, and $\sigma^2<\infty$. Also, the extinction probability of this branching process is 0. Thus, by Theorem 2, there exists a positive random variable W such that the population size N after n iterations divided by $(3/2)^n$ tends in distribution to W. In particular, for every $\delta>0$, we can find a number $c_\delta>0$ such that $P(N\geqq c_\delta(3/2)^n)$ is at least $1-\delta$ for all n large enough. The branching process is identified with our tree T_r as follows. The root node has two outgoing edges, with associated with it the pair of random variables $(U,\ 1-U)$ where U is uniformly distributed on $[0,1]$. We "keep" the two substrees if both U and $1-U$ are at least $\frac{1}{4}$, and keep only the subtree corresponding to $\max(U,\ 1-U)$ otherwise. Both events have equal probability $\frac{1}{2}$. This pruning process is repeated for all nodes. After having considered r levels of nodes, we end up with N surviving paths, each having the property that $V(p)\geqq 4^{-r}$. As seen above, for all $\delta>0$, there exists $c_\delta>0$ such that

$$P(H_n\geqq k)\geqq(1-q^{c_\delta(\frac{3}{2})^r})(1-\delta).$$

If $q<1$ is fixed (not dependent upon n), then the lower bound is at least $1-2\delta$ for all r large enough. Thus, it suffices to prove that for the first subtree T_{k-r}, there exists a fixed q smaller than 1.

To prove the existence of such a value of q, we apply the theory of branching processes once again. Start with a population of size 1, the root of T_{k-r}. This element has between 0 and 2^l children, according to the following rule: include all paths p of length l starting at the root, for which $V(p)\geqq e^{-l/(c-\frac{\varepsilon}{2})}$. For each child, repeat the same process. If this branching process survives, then it surely survives after $\frac{k-r}{l}$ generations, and the products of the values of $V(p)$ along paths of length $l\frac{k-r}{l}=k-r$ in the original tree are at least

$$\begin{aligned}\left(e^{-\frac{l}{\left(c-\frac{\varepsilon}{2}\right)}}\right)^{\frac{k-r}{l}} &= e^{-\frac{k-r}{\left(c-\frac{\varepsilon}{2}\right)}}\\ &\geqq e^{-\frac{(c-\varepsilon)\log(n)}{\left(c-\frac{\varepsilon}{2}\right)}}\\ &=\frac{n^{\frac{\varepsilon}{2c-\varepsilon}}}{n}\\ &\geqq 4^r\left(\frac{1+k}{n}\right)\end{aligned}$$

for all n large enough. Thus, $P\left(Z_{k-r}\geqq 4^r\left(\frac{1+k}{n}\right)\right)\geqq P$ (Branching process survives). From Theorem 1, we see that it if we can verify that for this branching process, $\mu>1$, then the probability of survival is a positive constant, which in our case will not depend upon n. This would then conclude the proof of the first part of the theorem. But clearly, if U_1, $U_2, \ldots$ are i.i.d. uniform $[0, 1]$ random variables, and G_l is a gamma (l) random variable, then

$$\mu = 2^l P\left(\prod_{i=1}^{l} U_i \geqq e^{-\frac{l}{c-\frac{\varepsilon}{2}}}\right)$$

$$= 2^l P\left(G_l \leqq \frac{l}{c-\frac{\varepsilon}{2}}\right)$$

$$\geqq 2^l \frac{e^{-\frac{l}{c-\frac{\varepsilon}{2}}}\left(\dfrac{l}{c-\frac{\varepsilon}{2}}\right)^l}{l!}$$

$$\geqq \left(\frac{2e e^{-\frac{1}{c-\frac{\varepsilon}{2}}}}{c-\frac{\varepsilon}{2}}\right)^l \frac{1}{\sqrt{2\pi l}\, e^{\frac{1}{12l}}}.$$

Here we used Stirling's approximation. By definition of c, the exponential term is greater than $(1+a)^l$ for some $\alpha>0$ depending upon ε. Thus, for all l large enough, $\mu>1$. Choose such a fixed value for l.

This leaves us with the statement about $E(H_n)$. Obviously,

$$E(H_n) \geqq (c-\varepsilon)\log(n)\, P(H_n \geqq (c-\varepsilon)\log(n)) \sim (c-\varepsilon)\log(n)$$

for all $\varepsilon>0$. This concludes the proof of Theorem 6. □

4. A More Precise Lower Bound

The purpose of this section is to prove that for a random binary search tree, $H_n - c\log(n)$ is $O(\sqrt{\log(n)\,\mathrm{loglog}(n)})$ in probability. This is essentially done by a careful adjustment of the parameters in the proof of Theorem 6. We do not claim that this is optimal. For example, an improvement of the inequality of Theorem 2 for branching processes could possibly affect the size of the second term in the lower bound.

Theorem 7. *Let c be as in Theorem 5, and let $\varepsilon>0$ be arbitrary. Then, for a random binary search tree with n nodes,*

$$\lim_{n\to\infty} P(H_n \leqq c\log(n) - (1+\varepsilon) K \sqrt{\log(n)\,\mathrm{loglog}(n)}) = 0,$$

where

$$K = 2c\sqrt{\frac{(\log(4)-1/c)\log(3)}{4(1-1/c)\log(3/2)}}.$$

Also,

$$\liminf_{n\to\infty} \frac{E(H_n)-c\log(n)}{\sqrt{\log(n)\operatorname{loglog}(n)}} \geqq -K.$$

Proof of Theorem 7. We will use the notation of the proof of Theorem 6. The constant L is defined by

$$L=\sqrt{\frac{\log(3/2)}{4(1-1/c)(\log(4)-1/c)\log(3)}}.$$

We will work with the following values for l and k:

$$l=\lfloor L\sqrt{\log(n)\operatorname{loglog}(n)}\rfloor,$$
$$k=\lfloor c\log(n)-(1+\varepsilon)K\sqrt{\log(n)\operatorname{loglog}(n)}\rfloor.$$

Furthermore, r is the unique integer in $[r_0, r_0+l)$ for which $\frac{k-r}{l}$ is integer, where

$$r_0=\left\lfloor l\frac{\log(2)}{\log(\frac{3}{2})}\right\rfloor.$$

Let us recall first that if $1-q$ is the survival probability of the first of the 2^r branching processes started at the leaves of T_r, then

$$P(H_n\geqq k)\geqq(1-q^{c_\delta(\frac{3}{2})^r})(1-\delta).$$

We recall that $\delta>0$ is arbitrary, and that $c_\delta>0$ is a given function of δ only. The inequality is valid for all r large enough, i.e. for all n large enough. The lower bound is $1-\delta-o(1)$ if

$$(\tfrac{3}{2})^r\log(q)\to-\infty$$

as $n\to\infty$. This is in turn satisfied if

$$2^l\log(q)\to-\infty.$$

The branching processes in the 2^r subtrees T_{k-r} are now defined in the following manner: each element has between 0 and 2^l descendants where a descendant corresponds to a path of length l having

$$V(p)\geqq\left(\frac{1+k}{n}4^r\right)^{\frac{l}{k-r}}.$$

We will call this event E. It is easy to see that for paths of length k in the pruned tree T_k, we have

$$V(p)\geqq\frac{1+k}{n}$$

as required. For one of the individual branching processes in a subtree T_{k-r}, we apply the inequality of Theorem 2, which states that the probability of

eventual survival $1-q$ is at least
$$\frac{2^l p-1}{2^l}$$

where p is the probability of the event E. In particular, we see that
$$1-q \geqq (1+o(1))\,p$$

when $2^l p \to \infty$. But because $\log(q) \leqq -(1-q)$, this would also imply $2^l \log(q) \to -\infty$ as required above. The remainder of the proof boils quite simply down to a verification of the condition $2^l p \to \infty$. Note that
$$\begin{aligned} p &= P\left(G_l \leqq l\,\frac{\log(n)-\log(1+k)-r\log(4)}{k-r}\right) \\ &\triangleq P\left(G_l \leqq \frac{l}{c}(1+a_n)\right) \end{aligned}$$

where G_l is a gamma (l) random variable, and a_n is a sequence of numbers tending to 0 (this will be verified below). By an inequality for gamma random variables, we have
$$\begin{aligned} 2^l p &\geqq \left(\frac{l(1+a_n)}{c}\right)^l \frac{2^l}{l!}\, e^{-\frac{l}{c}(1+a_n)} \\ &\sim \left(\frac{e(1+a_n)}{c}\right)^l \frac{2^l}{\sqrt{2\pi l}}\, e^{-\frac{l}{c}(1+a_n)} \\ &= \left(\frac{2e}{c}\, e^{-\frac{1}{c}}\right)^l \frac{1}{\sqrt{2\pi l}}\, e^{-\frac{l}{c}a_n}(1+a_n)^l \\ &= \frac{1}{\sqrt{2\pi}}\, e^{l\left(\log(1+a_n)-\frac{1}{c}a_n\right)-\frac{1}{2}\log(l)} \end{aligned}$$

The exponent in the last expression is
$$l a_n\left(1-\frac{1}{c}\right)-\frac{1}{4}\operatorname{loglog}(n)+O(l a_n^2)+O(\operatorname{logloglog}(n)).$$

After resubstitution of our value of l, we see that the exponent tends to ∞ if
$$\liminf \frac{l a_n}{\operatorname{loglog}(n)} > \frac{1}{4(1-1/c)},$$

and
$$a_n = O\left(\sqrt{\frac{\operatorname{loglog}(n)}{\log(n)}}\right).$$

We can verify this quite easily: we start from

$$\frac{la_n}{\log\log(n)} \sim La_n\sqrt{\frac{\log(n)}{\log\log(n)}}$$

$$= L\sqrt{\frac{\log(n)}{\log\log(n)}} \times \left(\frac{1-\frac{log(1+k)+r\log(4)}{\log(n)}}{\frac{k-r}{c\log(n)}} - 1 \right)$$

$$= L\sqrt{\frac{\log(n)}{\log\log(n)}} \times \left(\frac{\log(n)-\log(1+k)-r\log(4)-\left(\frac{k-r}{c}\right)}{\log(n)} \right)$$

$$\geqq L \times \left(\frac{(1+\varepsilon)K}{c} - \frac{r}{\sqrt{\log(n)\log\log(n)}}\left(\log(4)-\frac{1}{c}\right) + o(1) \right)$$

$$\geqq L \times \left(\frac{(1+\varepsilon)K}{c} - \frac{L\log(3)}{\log(3/2)}\left(\log(4)-\frac{1}{c}\right) + o(1) \right)$$

$$\left(\text{since } r \leqq L\sqrt{\log(n)\log\log(n)}\,\frac{\log(3)}{\log(3/2)}\right)$$

$$\sim L \times \left(\frac{\varepsilon K}{c} + \frac{L\log(3)}{\log(3/2)}\left(\log(4)-\frac{1}{c}\right) \right)$$

$$\left(\text{since } \frac{K}{c} = \frac{\log(3)}{\log(3/2)}\left(\log(4)-\frac{1}{c}\right) \times 2L\right)$$

$$> \frac{1}{4(1-1/c)}$$

when L is as indicated at the outset of the proof. Similarly, it is trivial to show that $a_n = O(\sqrt{\log\log(n)/\log(n)})$. This concludes the proof of the theorem. □

5. Multidimensional Trees

The results of this paper can be used in the analysis of the expected height of some multidimensional data structures. The $k-d$ tree [5] is a straightforward generalization of the binary search tree for d-dimensional data. The difference is that at each level of the tree a different coordinate is used for splitting a subset of R^d in two parts. For example, at even levels of a $2-d$ tree, cuts perpendicular to the x-axis are made, and at odd levels, cuts perpendicular to the y-axis are made. In general, we have a sequence $d_1, d_2, \ldots$ where d_i is the index of the axis to be cut at the i-th level. Although this sequence is typically 1, 2, ..., d, 1, 2, ..., d, 1, 2, ..., d, 1, 2, ..., there is no restriction whatever on it for our analysis. One could thus have 1, 1, 1, 1, ... in which case a binary search tree is obtained. Assume that the data consists of an i.i.d. sequence of d-dimensional random vectors with the property that all marginal distributions are nonatomic (this is to avoid duplicates among some coordinate values). Then for the first split, the random permutation model holds, as for the binary search tree. For subsequent splits, one can verify that this random permutation model is still valid regardless of which coordinate is cut. Thus, the height H_n has precisely the same distribution as in the one-dimensional case, and all results given for the binary search tree remain valid. The same can be said if the data consists of a randomly permuted sequence of n points in R^d having the property that there are no duplicates among the coordinate values.

The point quadtree in R^d [12] generalizes the binary search tree in a different way. Each data point is a node in a tree having 2^d subtrees corresponding to the quadrants formed by considering this data point as the new origin. The process of inserting into point quadtrees is analogous to that used in binary search trees. Search operations using point quadtrees are analyzed by [4] and [23]. Because of the large branching factor, point quadtrees seem ideally suited for parallel (multiprocessor) environments. For a survey of quadtrees, see [31]. Under the i.i.d. model given above for the $k-d$ tree, the height H_n of the point quadtree has a distribution which depends upon the distribution of the data points. For example, if the data points are uniformly distributed on the diagonal of the unit hypercube of R^d, then H_n is distributed as for the binary search tree. In a sense this is the worst case, since the branching factor (2^d) is of no help in making the tree flatter. For the sake of simplicity, we will assume that the data points are i.i.d. random vectors with independent components, and that each component has a nonatomic distribution. But as far as the distribution of the depths of the nodes is concerned, this is equivalent to assuming that the data points are uniformly distributed on the unit hypercube of R^d. Thus, the root point $(U_1, \ldots, U_d)$ splits the space into 2^d quadrants, and the number of data points ending up in these quadrants is multinomially distributed with parameter $n-1$ (for the total cardinality) and probabilities $U_1 U_2 \ldots U_d$, $U_1 U_2 \ldots U_{d-1}(1-U_d)$, etcetera. In other words, the fundamental inequality of Theorem 3 remains valid because it was based upon precisely this observation. However, in the definitions of $V(p)$ and Z_k, every occurrence of one uniform random variable has to replaced by a product of d independent uniform random vectors. For example, we have in the notation of Theorem 3,

$$P\left(Z_k \geqq \frac{1+k}{n}\right) \leqq P(H_n \geqq k) \leqq P\left(Z_k \geqq \frac{1}{n}\right) \leqq 2^{dk} P\left(\prod_{i=1}^{dk} U_i \geqq \frac{1}{n}\right),$$

where $U_1, \ldots$ are i.i.d. uniform $[0, 1]$ random variables. In the proofs of Theorems 6 and 7, the parameters k, l, r can all formally be replaced by dk, dl, dr. In particular, we see that for the quadtree with the independent component model,

$$\frac{H_n}{c \log(n)} \to \frac{1}{d} \text{ in probability,}$$

and

$$\lim_{n\to\infty} \frac{E(H_n)}{c \log(n)} = \frac{1}{d}.$$

In other words, the quadtree is about d times flatter than the binary search tree. If flatness is measured in terms of the expected value of the depth of a randomly picked node, computations much simpler than the ones given in this paper show that the same is true, i.e. the expected depth is asymptotic to $\frac{2}{d}\log(n)$ (versus $2\log(n)$ for the random binary search tree). Thus, if searching is done by applying d binary decisions for each node, we have a status quo. On the other hand, if d parallel processors are used when moving down the tree, a speed-up of about d can be expected over the random binary search tree, at least if communication time between the parallel processors is ignored.

6. The Union-Find Tree

In this section we consider the height of a random tree which occurs in the well-known tree-structured set-union algorithm (see e.g. [2], pp. 184–189, or [33]). The algorithm is sometimes called the union-find algorithm and the corresponding problem the equivalence or set union problem. In this data structure, each element in $1, \ldots, n$ has a parentpointer $parent[i]$, $1 \leqq i \leqq n$. Initially, each element defines one singleton set, which is indicated by setting $parent[i] := i$. Sets with more than one element are organized as trees, and the name or label of a set is the name of the root of the tree. The two operations of interest are

```
Find(i):     (returns the name of the set to which i belongs)
             $j := i;$
             WHILE $parent[j] \neq j$ DO $j := parent[j]$;
             RETURN($j$)
Union(i, j): (union of sets rooted at i and j respectively)
             $parent[j] := i.$
```

The cost of a union is constant. Its contribution to the total time taken by a sequence of k finds and m unions is proportional to m. The more interesting contribution to the total time comes from the find operations. Here we equate time with the number of comparisons "$parent[j] \neq j$" in the **WHILE** statement. The number of sets in the partition of $1, \ldots, n$ is nonincreasing. Thus, it makes sense to analyze the time taken by a sequence union(find(x_i), find(y_i)), $1 \leqq i \leqq n$, such that at the end of this sequence all points belong to the same set. A sequence (x_i, y_i) is said to be legal if after $i-1$ iterations, find$(x_i) \neq$ find(y_i). We look at the shape of the final tree. This is important because the time required for one find operation is bounded by the time required for one find operation after the elements have all been put in the same set. In the structure of the final tree, we distinguish between $L_{n1}, \ldots, L_{nn}$, the levels of elements $1, \ldots, n$ (the level of the root is defined as 0), and $H_n = \max_{1 \leqq i \leqq n} L_{ni}$, the height of the tree. Since these random variables are only determined by how v is distributed in the union (u, v) operations, all that will be said below will be valid for the following general situation:

At each iteration, choose (u, v) in such a way that $u \neq v$ and that both u and v are uniformly distributed over all possible sets. (Note that we do not specify how x_i and y_i are picked within the selected sets.)

It is instructive to see how large the level of an individual element is, before looking at the height of the tree. To do this, consider B_p, $0 \leqq p \leqq 1$, a Bernoulli (p) random variable:

$$P(B_p=1)=1-P(B_p=0)=p.$$

It is clear that $L_{n1}, \ldots, L_{nn}$ are identically distributed random variables with distribution equal to

$$\sum_{i=2}^{n} B_{1/i}, \tag{1}$$

where the Bernoulli random variables are independent. This follows from the observation that whenever element 1 is in $\operatorname{find}(y_i)$, its level increases by 1. The level of element 1 remains unchanged when it is in $\operatorname{find}(x_i)$.

Theorem 8. 1. $E(L_{n1})=\sum_{i=2}^{n} 1/i \sim \log n$.

2. $\operatorname{Var}(L_{n1}) \sim \log n$.
3. $L_{n1}/\log n \to 1$ *in probability as* $n \to \infty$.
4. $(L_{n1}-\log n)/\sqrt{\log n}$ *tends in distribution to a normal* $(0, 1)$ *random variable.*

Proof of Theorem 8. Part 1 is trivially true. For part 2, we observe that $\operatorname{Var}(L_{n1})$ is equal to

$$\sum_{i=2}^{n} (1/i-1/i^2).$$

This in turn is equal to $\log(n)+O(1)$. For part 3, we need only show that

$$L_{n1} \Big/ \sum_{i=2}^{n} 1/i \to 1 \text{ in probability.}$$

But by Chebyshev's inequality, we have for all $\varepsilon>0$,

$$\begin{aligned} P\left(\left|L_{n1}-\sum_{i=2}^{n} 1/i\right| \geqq \varepsilon \sum_{i=2}^{n} 1/i\right) & \\ \leqq \operatorname{Var}(L_{n1})/(\varepsilon^2(\log n)^2) & \\ =o(1). & \end{aligned}$$

Part 4 follows without work from parts 1 and 2 by Lindeberg's version of the central limit theorem (see e.g. [10], pp. 290–291). This concludes the proof of Theorem 8. □

Let us now turn to the study of the height of the final tree. We will begin with the following upper bound, which is once again the easy half of the analysis.

Theorem 9. 1. *For* $\varepsilon>0$, $P(H_n>(1+\varepsilon)\log n) \leqq c^{\varepsilon} n/n^{(1+\varepsilon)\log(1+\varepsilon)-\varepsilon}$, *where* $c = e^{\gamma+\frac{1}{2n}}$ *and* γ *is Euler's constant.*

2. *If ε_n is any sequence of positive numbers tending to ∞ as $n \to \infty$, then $P(H_n \geqq e \log n + \varepsilon_n) \to 0$ as $n \to \infty$.*

3. $E(H_n) \leqq e \log n + O(1)$.

Proof of Theorem 9. For $x > 0$,

$$\begin{aligned} P(L_{n1} \geqq x) &\leqq E(e^{tL_{n1}} e^{-tr}) \qquad (t>0) \text{ (Jensen's inequality)} \\ &= \prod_{i=2}^{n} \left(\frac{1}{i} e^t + 1 - \frac{1}{i}\right) e^{-tx} \\ &\leqq e^{-tx + \sum_{i=1}^{n} \frac{1}{i}(e^t - 1)} \\ &\leqq e^{-tx + \left(\log n + \gamma + \frac{1}{2n}\right)(e^t - 1)} \\ &= c^{e^t - 1} e^{-tx + (\log n)(e^t - 1)}. \end{aligned}$$

Here we used a well-known inequality for the harmonic series (see e.g. [19, 20]). The second factor in the upper bound is minimized with respect to t for the value $t = \log(x/\log n)$. Thus, for $x \geqq \log n$,

$$P(L_{n1} \geqq x) \leqq c^{\frac{x}{\log n} - 1} e^{-\log n + x - x \log x + x \log\log n}.$$

In particular, if we replace x by $(1+\varepsilon)\log n$, then we obtain the inequality in part 1, after observing that

$$P(H_n \geqq x) \leqq n P(L_{n1} \geqq x).$$

Part 2 of Theorem 9 follows from part 1: take $\varepsilon = e - 1 + \delta_n$, where $\delta_n \downarrow 0$. Note that with this choice,

$$(1+\varepsilon)(\log(1+\varepsilon) - 1) \sim \delta_n.$$

Thus, using the notation of part 2, with $\varepsilon_n = \delta_n \log n$,

$$P(H_n \geqq e \log n + \varepsilon_n) \leqq \frac{c^{e-1+o(1)}}{e^{(1+o(1))\delta_n \log n}} = o(1).$$

Part 3 can be shown as follows:

$$\begin{aligned} E(H_n) &= \int_0^\infty P(H_n \geqq x)\, dx \\ &\leqq e \log n + \int_0^\infty P(H_n > e \log n + x)\, dx \\ &= e \log n + \int_0^\infty \log n\, P(H_n > (e+u)\log n)\, du \\ &\leqq e \log n + \int_0^\infty \log n \frac{c^{e+u-1}}{n^{(u+e)\left(\log\left(1+\frac{u}{e}\right)\right)}}\, du \end{aligned}$$

$$\begin{aligned}
&\leqq e\log n + c^{e-1}\log n \int_0^\infty c^u n^{\frac{-2u(u+e)}{2e+u}}\,du \qquad \left(\text{use } \log\left(1+\frac{u}{e}\right) \geqq \frac{2u}{2e+u}\right)\\
&\leqq e\log n + c^{e-1}\log n \int_0^\infty \left(\frac{c}{n}\right)^u du \qquad \left(\text{use } \frac{u+e}{u+2e} \geqq \frac{1}{2}\right)\\
&= e\log n + c^{e-1}\log n/\log(n/c)\\
&= e\log n + O(1),
\end{aligned}$$

which was to be shown. This concludes the proof of Theorem 9. □

In Theorem 9, we have shown that the expected height of the final tree is at most e times the expected depth. Also, the probability that the height is much larger than $e \log n$ is extremely small in view of the exponential bound given in part 1 of Theorem 9. We will now show that it is very unlikely that H_n is smaller than $(e-\varepsilon)\log(n)$ for any $\varepsilon > 0$:

Theorem 10. *For a random union-find tree with n elements, we have*

$$\frac{H_n}{e\log(n)} \to 1 \text{ in probability}$$

and

$$E(H_n) \sim e\log(n).$$

Proof of Theorem 10. In view of Theorem 9, we need only be concerned with lower bounds here. Also, the lower bound for $E(H_n)$ follows from the convergence in probability. It will be convenient to work with the associated binary tree (abbreviated to ABT below). This tree is constructed as follows: define all n elements as leaves. Internal nodes represent sets of more than one element. When two sets with roots at u and v are merged, with parent v, then a new internal node is defined having as left subtree the tree which represents the set containing u, and as right subtree the tree containing v. This tree has obviously $n-1$ internal nodes, one for each union operation. The tree formed by the internal nodes is called the ABT.

Let us define the left-depth of a node as the number of left edges encountered on the path from the node to the root, and let the left-height of the tree be the maximal left-depth. It is clear that if L_{n-1} is the left-height of the ABT, and H_n is the height of the original union-find tree, then

$$H_n = 1 + L_{n-1} \qquad (n \geqq 2).$$

Furthermore, the ABT is distributed as a random binary search tree on $n-1$ nodes for the standard random permutation model. This can be seen inductively: If a new first element is added, then it is merged with a randomly picked external node in the ABT. Since this property characterizes random binary search trees too, we see that the ABT on $n-1$ nodes is distributed as a random binary search tree on $n-1$ nodes. This observation is due to [21].

In the study of L_n, it is convenient to work with an infinite complete binary tree T in which we associate with the edges, in level order and from left to right, random variables $U_1, (1-U_1)$, U_2, $(1-U_2), \ldots$ where the U_i's are i.i.d.

uniform [0, 1] random variables. For a path p, we define its value $V(p)$ as the product of the uniform random variables encountered on the path. Often, we will use this notation for the value of a node. It is understood that in such cases, we mean the value of the path from the node to the root. Thus, p can be used for a path from a node to the root, and for the node itself. We will also use $LH(p)$, $H(p)$ for the left-height and the height of p. The starting point of the analysis is the following fundamental inequality:

$$P(L_n \geqq h) \geqq P\left(\max_{\substack{p:\, LH(p) \geqq h \\ p:\, H(p) \leqq h^2}} V(p) \geqq \frac{1+h^2}{n} \right).$$

To see this, note that the sizes of the subtrees of the root are distributed as $\lfloor nU \rfloor$ and $\lfloor n(1-U) \rfloor$ where U is a uniform [0, 1] random variable. These are stochastically greater than $nU-1$ and $n(1-U)-1$. If we define $V(p)=1$ for the root, then the latter quantities can be rewritten as $nUV(p)-1$ and $n(1-U)\,V(p)-1$. If u, v are the children of the root, then

$$nUV(p)-1 = nV(u)-1, \qquad n(1-U)\,V(p)-1 = nV(v)-1.$$

By induction, we obtain that jointly for all 2^d nodes p at distance d from the root, the sizes of the subtrees rooted at these nodes are stochastically greater than $nV(p)-d$. Thus, the event $[L_n \geqq h]$ is implied by the event that one of the nodes at leftheight h is a root of a subtree of size at least equal to one. This event is in turn implied by the event that one of the nodes p of leftheight h has $V(p) \geqq \dfrac{1+H(p)}{n}$. Since it is inconvenient to have $H(p)$ in this condition, we get rid of it by observing that the last event is implied by the event that $\max V(p) \geqq (1+h^2)/n$ where the maximum is taken over all nodes p of leftheight h and $H(p) \leqq h^2$. This concludes the proof of our starting inequality.

From here onwards, the proof follows the lines of the proof of Theorem 6. Let r be a large but fixed integer, let $\varepsilon > 0$ be an arbitrarily small constant, let l be another large fixed integer, and let h be defined as the largest multiple of l not exceeding $(e-\varepsilon)\log(n)$. Let T_r be the complete binary tree of Theorem 3. Of its 2^r leaves, we keep only those for which $V(p) \geqq 4^{-r}$. The infinite complete binary trees rooted at these leaves will be called $T(1), \ldots, T(N)$. Assume furthermore that we have shown that the probability that $T(1)$ has a node of leftheight h, height $\leqq (h-r)^2$ and $V(p) \geqq \dfrac{1+h^2}{n}\, 4^r$ is at least $1-q>0$ where q does not depend upon n and V, LH, H are all relative to the root of $T(1)$. In this case, we call $T(1)$ "good". Then, for arbitrary $\delta > 0$, there exists $c_\delta > 0$ such that for r large enough,

$$\begin{aligned} P(L_n \geqq h) &\geqq P\left(\bigcup_{i=1}^{N} T(i) \text{ is good} \right) \\ &\geqq E(1-q^N) \\ &\geqq (1-\delta)(1-q^{c_\delta (3/2)^r}) \\ &\geqq 1-2\delta. \end{aligned}$$

We have implicitly used the fact that for some node in a good tree $T(i)$,

$$V(p) \geqq \frac{1+h^2}{n} 4^r 4^{-r} = \frac{1+h^2}{n},$$

$$LH(p) \geqq h,$$

$$H(p) \leqq r + (h-r)^2 \leqq h^2.$$

The tree $T(1)$ is associated with a branching process in the following manner: consider all nodes at leftheight l, and height $\leqq l^2$ for which

$$V(p) \geqq e^{-\frac{l}{e \times \frac{\varepsilon}{2}}}.$$

These nodes form the first generation in the branching process. Repeat this construction for each of these nodes in turn to construct the second generation, and so forth. If this branching process survives, then it certainly survives after h/l iterations, and $T(1)$ must therefore have at least one node p with the following properties:

$$V(p) \geqq \left(e^{-\frac{l}{\varepsilon - \frac{\varepsilon}{2}}}\right)^{\frac{h}{l}} \geqq e^{-\frac{h}{e-\varepsilon}(1+o(1))} = e^{\log(1+h^2) + r\log(4) - \log(n)} = \frac{1+h^2}{n} 4^r;$$

$$LH(p) \geqq \frac{h}{l}\, l = h;$$

$$H(p) \leqq \frac{h}{l}\, l^2 \leqq hl \leqq (h-r)^2.$$

We note in passing that all of this remains valid for n larger than some n_0 depending upon l, r, ε.

Let us next look at the survival probability of the branching process. At height H, leftheight l, we have

$$\binom{H}{l}$$

nodes in $T(1)$. Thus, the mean μ for the branching process is

$$\mu = \sum_{H=l}^{l^2} P\left(\prod_{j=1}^{H} U_j \geqq e^{-\frac{l}{e-\frac{\varepsilon}{2}}}\right)$$

where the U_j's are i.i.d. uniform $[0, 1]$ random variables. If we use the notation G_H for a gamma (H) random variable, then it is easily seen that

$$\mu = \sum_{H=l}^{l^2} \binom{H}{l} P\left(G_H \leqq \frac{l}{e - \frac{\varepsilon}{2}}\right)$$

$$= \sum_{H=l}^{l^2} \binom{H}{l} \int_0^{l/(e-\varepsilon/2)} \frac{y^{H-1}}{(H-1)!} e^{-y}\, dy$$

$$= \int_0^{l/(e-\varepsilon/2)} \frac{y^{l-1}}{l!} \sum_{H=l}^{l^2} \frac{H y^{H-l}}{(H-l)!} e^{-y}\, dy$$

$$\geq \int_0^{l/(e-\varepsilon/2)} \frac{y^{l-1}}{(l-1)!} \left(\sum_{H=l}^{\infty} \frac{y^{H-l}}{(H-l)!} - \sum_{H=l^2+1}^{\infty} \frac{y^{H-l}}{(H-l)!} \right) e^{-y}\, dy$$

$$\geq \int_0^{l/(e-\varepsilon/2)} \frac{y^{l-1}}{(l-1)!} \left(1 - \frac{y^{l^2+1-l}}{(l^2+1-l)!} \right) dy$$

$$= \left(\frac{l}{e-\frac{\varepsilon}{2}} \right)^{l} \frac{1}{l!} - \left(\frac{l}{e-\frac{\varepsilon}{2}} \right)^{l^2+2-l} \frac{1}{(l^2+2-l)!}$$

$$\sim \left(\frac{e}{e-\frac{\varepsilon}{2}} \right)^{l} \frac{1}{\sqrt{2\pi l}}$$

$$\rightarrow \infty$$

as $l \rightarrow \infty$. Now, choose l large enough so that $\mu > 1$. By Theorem 2, this fixes the probability of survival $1-q>0$. Going back over the proof, we see that this allows us to fix a value of r. After having fixed r, we do not need to re-adjust our original choice of l, because all the estimates are valid for $n \geqq n_0$ where only n_0 is affected by the choices of r, l. This concludes the proof of Theorem 10. □

Acknowledgements. I would like to thank all the referees for pointing out improvements in the presentation and in one inequality in the proof of Theorem 7.

References

1. Aho, A.V., Hopcroft, J.E., Ullman, J.D.: The design and analysis of computer algorithms. Reading, Mass.: Addison-Wesley 1975
2. Aho, A.V., Hopcroft, J.E., Ullman, J.D.: Data structures and algorithms. Reading, Mass.: Addison-Wesley 1983
3. Athreya, K.B., Ney, P.E.: Branching processes. Berlin-Heidelberg-New York: Springer 1972
4. Bentley, J.L., Stanat, D.F.: Analysis of range searches in quad trees. Inf. Proc. Lett. **3**, 170–173 (1975)
5. Bentley, J.L.: Multidimensional binary search trees used for associative searching. Commun. ACM **18**, 509–517 (1975)
6. Biggins, J.D.: The first and last-birth problems for a multitype age-dependent branching process. Adv. Appl. Probab. **8**, 446–459 (1976)
7. Biggins, J.D.: Chernoff's theorem in the branching random walk. J. Appl. Probab. **14**, 630–636 (1977)
8. Brown, G.G., Shubert, B.O.: On random binary trees. Math. Oper. Res. **9**, 43–65 (1984)
9. de Bruijn, N.G., Knuth, D.E., Rice, S.O.: The average height of planted plane trees. In: Read, R.-C. (ed.) Graph Theory and Computing, pp. 15–22. New York: Academic Press 1972
10. Chow, Y.S., Teicher, H.: Probability theory. Berlin-Heidelberg-New York: Springer 1978
11. Devroye, L.: A note on the expected height of binary search trees. J. ACM **33**, 489–498 (1986)

12. Finkel, R.A., Bentley, J.L.: Quad trees: a data structure for retrieval on composite keys. Acta Inf. **4**, 1-9 (1974)
13. Flajolet, P., Odlyzko, A.: The average height of binary trees and other simple trees. J. Comput. Syst. Sci. **25**, 171-213 (1982)
14. Flajolet, P., Steyaert, J.M.: A branching process arising in dynamic hashing, trie searching and polynomial factorization. Lect. Notes Comput. Sci., vol. 140, pp. 239-251. Berlin-Heidelberg-New York: Springer 1982
15. Flajolet, P.: On the performance evaluation of extendible hashing and trie search. Acta Inf. **20**. 345-369 (1983)
16. Gonnet, G.H.: Unpublished handwritten letter. (1983)
17. Harris, T.E.: The theory of branching processes. Berlin-Heidelberg-New York: Springer 1963
18. Kemp, R.: On the stack size of regularly distributed binary trees. 6th ICALP Conference, held at Udine, Italy, 1979
19. Knuth, D.E.: The art of computer programming, vol. 1. Fundamental algorithms, 2nd Ed. Reading, Mass.: Addison-Wesley 1973
20. Knuth, D.E.: The art of computer programming, vol. 3. Sorting and searching. Reading, Mass.: Addison-Wesley 1973
21. Knuth, D.E., Schonhage, A.: The expected linearity of a simple equivalence algorithm. Theor. Comput. Sci. **6**, 281-315 (1978)
22. Kounias, E.G.: Bounds for the probability of a union, with applications. Ann. Math. Statist. **39**, 2154-2158 (1968)
23. Lee, D.T., Wong, C.K.: Worst-case analysis for region and partial region searches in multidimensional binary search trees and quad trees. Acta Inf. **9**, 23-29 (1977)
24. Mendelson, H.: Analysis of extendible hashing. IEEE Trans. Software Eng. **8**, 611-619 (1982)
25. Pittel, B.: Asymptotical growth of a class of random trees. Ann. Probab. **13**, 414-427 (1985)
26. Pittel, B.: Paths in a random digital tree: limiting distributions. Adv. Appl. Probab. **18**, 139-155 (1986)
27. Rao, C.R.: Linear statistical inference and its applications. New York: John Wiley 1973
28. Regnier, M.: On the average height of trees in digital search and dynamic hashing. Inf. Process. Lett. **13**, 64-66 (1982)
29. Renyi, A., Szekeres, G.: On the height of trees. J. Aust. Math. Soc. **7**, 497-507 (1967)
30. Robson, J.M.: The height of binary search trees. Aust. Comput. J. **11**, 151-153 (1979)
34. Samet, H.: The quadtree and related hierarchical data structures. Comput. Surv. **16**, 187-260 (1984)
32. Savage, I.R.: Probability inequalities of the Tchebycheff type. Res. Nat. Bur. Stand. **65**, 211-222 (1961)
33. Standish, T.A.: Data structure techniques. Reading, Mass.: Addison-Wesley 1980
34. Stepanov, V.E.: On the distribution of the number of vertices in strata of a random tree. Theory Probab. Appl. **14**, 65-78 (1969)

Received May 10, 1985/January 8, 1987