Chapter 10

Digital Search Trees Average case analysis of digital search trees and tries

Nicolai Baron von Hoyningen-Huene

10.1 Introduction

A very common problem in computer science is to search for the appearance of a string inside a text. There exist algorithms that use suffix trees, which are often implemented as digital trees to optimize the performance of the search. In this article some properties of those data structures are analyzed, which can be used to calculate the average performance and space complexity of algorithms using digital trees. For an introduction to Rice's method (see Chapter 9) is recommended, but the mathematical derivation in this article is largely based on [FS86b].
In the first part basic terms and structures are defined. Subsequently there will be a detailed analysis of the internal path length and external internal nodes for digital search trees. The following derivation for properties of tries are just sketched because of similarity to the precedent part. Then the binary trees are generalized to M-ary trees and examined. Concluding, a general framework for analysis of properties of digital trees is presented.

10.2 Trees

First of all we look at rudimentary definitions for the sake of completeness and later usage, the definitions are taken from [CLR90]. In computer science data has to be stored in an intelligent way. Often there exist keys which are related to data, thus a special data structure is needed.

Definition 10.1. A **dictionary** is defined as an abstract data type storing items, or keys, associated with values. Basic operations are insert, find, and delete.

The operations new(), insert(i, v, D), and find(i, D) may be defined as follows:

- *new* () returns a dictionary

- $find\,(i, insert\,(i, v, D)) = v$
  $find\,(i, insert\,(j, v, D)) = find\,(i, D)$ if $i \neq j$ where i and j are items or keys, v is a value, and D is a dictionary. The operation $find\,(i, new\,())$ is not defined.
- The modifier function $delete\,(i, D)$ may be defined as follows.
  $delete\,(i, new\,()) = new\,()$
  $delete\,(i, insert\,(i, v, D)) = delete\,(i, D)$
  $delete\,(i, insert\,(j, v, D)) = insert\,(j, v, delete\,(i, D))$ if $i \neq j$

We can define $find\,(i, new\,())$ using a special value: $fail$. This only changes the return type of $find$. $find\,(i, new\,()) = fail$
A tree is a commonly used structure for implementation of dictionaries:

Definition 10.2. A **tree** is defined as a data structure accessed beginning at the root node. Each node is either a leaf or an internal node. An internal node has one or more child nodes and is called the parent of its child nodes. All children of the same node are siblings. Contrary to a physical tree, the root is usually depicted at the top of the structure, and the leaves are depicted at the bottom.

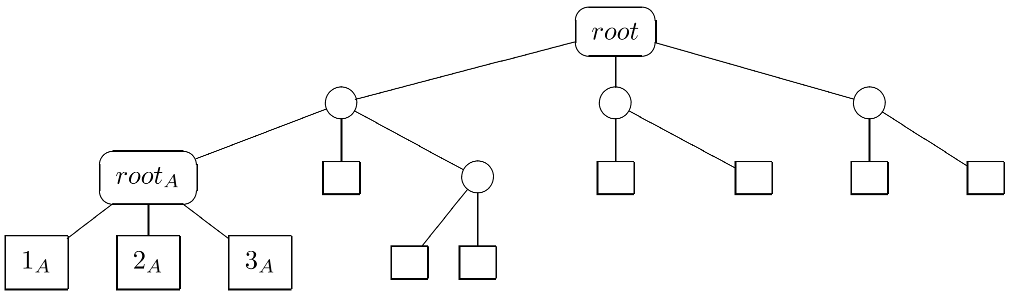

Figure 10.1: Example for a trinary tree

And we have a special property of a tree:

Definition 10.3. The **depth** of a node in a tree is the distance from this node to the root of the tree.

We can constrain trees to optimize performance for some purpose:

Definition 10.4. A **search tree** is a tree where every subtree of a node has values less than any other subtree of the node to its right. The values in a node are conceptually between subtrees and are greater than any values in subtrees to its left and less than any values in subtrees to its right.

In the binary world of a computer, trees with binary properties are widely supported:

Definition 10.5. A **binary tree** is either empty (no nodes), or has a root node, a left binary tree, and a right binary tree.

10.3 Digital Search Trees

10.3.1 Data Structure of Binary Search Trees

Definition 10.6. A **binary search tree** is a binary tree and also a search tree. A new node is added as a leaf.

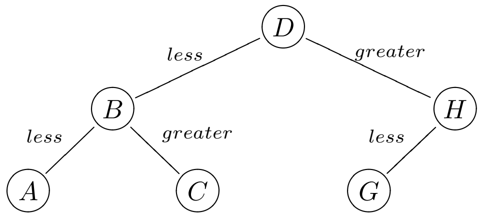

Figure 10.2: Example for a binary search tree with lexical ordering

The worst case for search is in the order of the number of keys N stored in a binary search tree, since the tree can be degenerated to a linear list. This arise, when keys are inserted in a a- or descending order. Instead the average case for successful search in a binary search tree is logarithmic, because the worst case is very unlikely. It evaluates to

$$2\left(1+\frac{1}{N}\right)H_N-3=(2\ln 2)\lg N+2\gamma-3+O\left(\frac{\log N}{N}\right).$$

10.3.2 Data Structure of Digital Search Trees

Keys are always handled by a computer as binary data. Therefore we can use the digital properties of the keys.

Definition 10.7. A **digital tree** is a tree for storing a set of strings where nodes are organized by substrings common to two or more strings.

The ordering of the keys is intuitively: we follow the tree by the bits descending from the first bit of the key represented as a binary number until we come to a leaf, a zero directs us to the left, a one to the right. In Figure 10.3 and 10.4 you can see an example tree, the binary coding of each letter is written next to it. Note, that the structure of the digital search tree depends of the order of the input of the keys. We define N as the number of keys stored in this dictionary. The number of nodes is limited by the number of bits in the keys and larger than $\lg N$ but likely less than a constant factor for many natural situations.

Definition 10.8. A **digital search tree** is a dictionary implemented as a digital tree which stores keys in internal nodes, so there is no need for extra leaf nodes to store the keys.

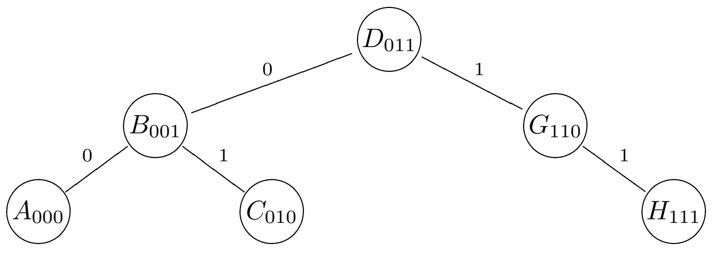

Figure 10.3: Example of a digital search tree with internal path length = 8

The worst case is the same as for binary search trees in a similar pathological case. But the average case for successful search in a digital search tree is improved:

$$\lg N+\frac{\gamma-1}{\ln 2}+\frac{3}{2}-\alpha+\delta(N)+O\left(\frac{\log N}{N}\right).$$

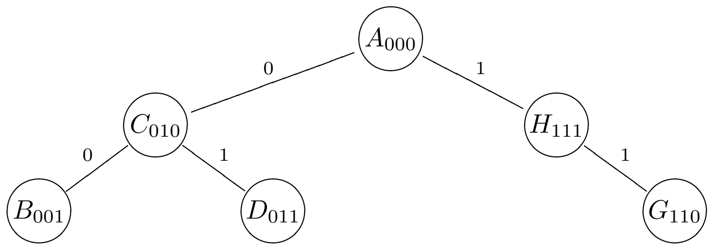

Figure 10.4: Another example of a digital search tree with same keys

10.3.3 Internal Path Length

In this chapter the first property of a digital search tree is analyzed.

Definition 10.9. The internal path length of a tree is the sum of the depth of every node of the tree.

This property is directly related to the complexity of the data structure: The number of nodes examined during a successful search in a search tree with N nodes is the path length of this node and in average case this counts as one plus the internal path length normalized through division by N.

Let A_N be the average internal path length of a digital search tree built from N (sufficiently long) keys comprised of random bits. Then we have the fundamental recurrence relation

$$A_N = N - 1 + \sum_{k=0}^{\infty} \frac{1}{2^{N-1}} \binom{N-1}{k} \left(A_k + A_{N-1-k}\right), \qquad N \geq 1 \tag{10.1}$$

with $A_0 := 0$.

The internal path length of any tree of N nodes is the sum of the internal path lengths of the subtrees of the node plus $N-1$, that are the ones missing for the distance from each node to the root of the whole tree. We count for each possible partition of the nodes to both subtrees and weight the sum by the number of all possibilities. The subtrees are randomly built.

We strike now for the goal to approximate A_N to get an explicit useful term. By symmetry A_k is equal to A_{N-1-k} and we get

$$A_N = N - 1 + \sum_{k=0}^{\infty} \frac{1}{2^{N-1}} \binom{N-1}{k} \left(A_k + A_k\right)$$

This equation is transformed into a functional one on the exponential generating function $A(z) = \sum_{N=0}^{\infty} A_N \frac{z^N}{N!}$ with $A'(z) = \sum_{N=1}^{\infty} A_N \frac{z^{N-1}}{(N-1)!}$ by multiplying both sides by $\frac{z^{N-1}}{(N-1)!}$ and summing for $N \geq 1$:

$$\begin{aligned}
\sum_{N=1}^{\infty} \frac{A_N z^{N-1}}{(N-1)!} &= \sum_{N=1}^{\infty} \frac{(N-1)\, z^{N-1}}{(N-1)!} + 2 \sum_{N=1}^{\infty} \sum_{k=0}^{\infty} \frac{1}{2^{N-1}} \binom{N-1}{k} A_k \frac{z^{N-1}}{(N-1)!} \\
&= z \sum_{N=2}^{\infty} \frac{z^{N-2}}{(N-2)!} + 2 \sum_{k=0}^{\infty} \sum_{N=k+1}^{\infty} \frac{A_k}{2^{N-1}} \frac{(N-1)!}{k!\,(N-1-k)!} \frac{z^{N-1}}{(N-1)!} \\
&= z \sum_{t=0}^{\infty} \frac{z^t}{t!} + 2 \sum_{k=0}^{\infty} \frac{A_k}{k!} \sum_{N=k+1}^{\infty} \frac{z^{N-1}}{2^{N-1}\,(N-k-1)!} \\
&= ze^z + 2 \sum_{k=0}^{\infty} \frac{A_k}{k!} \sum_{N=k+1}^{\infty} \left(\frac{z}{2}\right)^{N-1} \frac{1}{(N-k-1)!} \\
&= ze^z + 2 \sum_{k=0}^{\infty} \frac{A_k}{k!} \sum_{N=0}^{\infty} \left(\frac{z}{2}\right)^{(N+k+1)-1} \frac{1}{((N+k+1)-k-1)!} \\
&= ze^z + 2 \sum_{k=0}^{\infty} \frac{A_k}{k!} \left(\frac{z}{2}\right)^{k} \sum_{N=0}^{\infty} \left(\frac{z}{2}\right)^{N} \frac{1}{N!} \\
&= ze^z + 2 \sum_{k=0}^{\infty} \frac{A_k}{k!} \left(\frac{z}{2}\right)^{k} e^{\frac{z}{2}} \\
A'(z) &= ze^z + 2A\left(\frac{z}{2}\right) e^{\frac{z}{2}}
\end{aligned}$$

We simplify this by substituting $e^z B(z)$ for $A(z)$ with

$$B(z) = \sum_{N=0}^{\infty} B_N \frac{z^N}{N!}$$

and

$$B'(z) = \sum_{N=1}^{\infty} B_N \frac{z^{N-1}}{(N-1)!}$$

That is, $A(z) = e^z B(z)$ and $A'(z) = e^z B'(z) + e^z B(z)$. One can say that $B(z)$ is the expectation of the internal path length, if the number of keys is Poisson with parameter z. So the equation reads in terms of $B(z)$ as

$$e^z B'(z) + e^z B(z) = ze^z + 2B\left(\frac{z}{2}\right) e^{\frac{z}{2}} e^{\frac{z}{2}}$$

$$B'(z) + B(z) = z + 2B\left(\frac{z}{2}\right)$$

$$\sum_{N=1}^{\infty} B_N \frac{z^{N-1}}{(N-1)!} + \sum_{N=0}^{\infty} B_N \frac{z^N}{N!} = z + 2 \sum_{N=0}^{\infty} B_N \frac{\left(\frac{z}{2}\right)^N}{N!}$$

This corresponds to a simple recurrence on the coefficients

$$B_N + B_{N-1} = \frac{1}{2^{N-2}} B_{N-1}$$

$$B_N = -\left(1 - \frac{1}{2^{N-2}}\right) B_{N-1}$$

for $N \geq 3$ with $B_2 = 1$.
And leads us to an explicit formula for B_N:

$$B_N = (-1)^N \prod_{j=1}^{N-2} \left(1 - \frac{1}{2^j}\right)$$

So we can get an explicit formula for A_N:

$$\begin{aligned} A(z) &= e^z B(z) \\ &= e^z \sum_{N=0}^{\infty} B_N \frac{z^N}{N!} \\ &= \left(\sum_{N=0}^{\infty} \frac{z^N}{N!}\right)\left(\sum_{N=0}^{\infty} B_N \frac{z^N}{N!}\right) \\ &= \sum_{N=0}^{\infty} \left(\sum_{k=0}^{N} \frac{1}{(N-k)!}\frac{B_k}{k!}\right) z^N \\ &= \sum_{N=0}^{\infty} \left(\sum_{k=0}^{N} B_k \binom{N}{k}\right) \frac{z^N}{N!} \end{aligned}$$

$$A_N = \sum_{k=0}^{N} \binom{N}{k} B_k \tag{10.2}$$

We want to analyse this sum by Rice's integral resp. a theorem for meromorphic function, which is derived in 9:

Theorem 10.1. *Let ϕ be a function holomorphic in a domain that contains the half-line $[n_0, \infty[$. If n is big enough we have:*
If ϕ is meromorphic on the half-plane defined by $\Re(s) \geq d$ for some $d < n_0$ and of polynomial growth in this set, then

$$\sum_{k=n_0}^{n} \binom{n}{k} (-1)^k \phi(k) = -\sum_{s} (-1)^n Res_s \left(\phi(s) \frac{n!}{s(s-1)\dots(s-n)}\right) + O(n^d) \tag{10.3}$$

where the sum is taken over all poles but $n_0, n_0+1, \dots, n$.

Therefore we introduce a new series Q_N to come closer to the preceding equation:

$$Q_N = \prod_{j=1}^{N} \left(1 - \frac{1}{2^j}\right)$$

So we can get $B_N = (-1)^N Q_{N-2}$and by substitution:

$$A_N = \sum_{k=2}^{N} \binom{N}{k} (-1)^k Q_{k-2} \tag{10.4}$$

Q_N is defined only for integers, so we have to find a meromorphic function to extend Q_N to the complex plane. We choose the following function

$$Q(x) = \prod_{j=1}^{\infty} \left(1 - \frac{x}{2^j}\right)$$

with obviously $Q(1) = Q_\infty$ and you can see clearly that $\frac{Q(1)}{Q(2^{-N})}$ is a correct extension:

$$Q_N = \prod_{j=1}^{N} \left(1 - \frac{1}{2^j}\right) = \frac{\prod_{j=1}^{\infty} \left(1 - \frac{1}{2^j}\right)}{\prod_{j=N}^{\infty} \left(1 - \frac{1}{2^j}\right)} = \frac{Q(1)}{\prod_{j=1}^{\infty} \left(1 - \frac{1}{2^{j+N}}\right)} =$$

$$\frac{Q(1)}{\prod_{j=1}^{\infty} \left(1 - \frac{2^{-N}}{2^j}\right)} = \frac{Q(1)}{Q(2^{-N})}$$

Our equation now has the form:

$$A_N = \sum_{k=2}^{N} \binom{N}{k} (-1)^k \frac{Q(1)}{Q(2^{-k+2})}$$

$Q(x)$ is obviously meromorphic on the half-plane defined by $\Re(s) \geq d$ with $d := \frac{1}{2}$ and the function is also polynomial. So we get the following equation by theorem 10.1:

$$A_N = -\sum_z (-1)^N Res_z \left(B(N+1,-z) \frac{Q(1)}{Q(2^{-z+2})} \right) + O\left(N^{\frac{1}{2}}\right) \tag{10.5}$$

We proceed to evaluate the poles at $z \leq 1$ for $B(N+1,-z) \frac{Q(1)}{Q(2^{-z+2})}$:

- $B(N+1,-z)$ is singular at $z = 0,1$ because the Γ function is zero in the denominator.
- $z = j \pm \frac{2\pi ik}{\ln 2}$ for $j = 1, 0, -1, \ldots$ and all $k \geq 0$ are poles since at these points $2^{-z+j} = 1$ which causes one of the factors of $Q\left(2^{-z+2}\right)$ to vanish.

Only the poles for $z \geq \frac{1}{2}$ are within the region of interest.
So we can approximate the residues at the poles. Beginning with $z = 1$:

$$-B(N+1,-z) \frac{Q(1)}{Q(2^{-z+2})} = -B(N+1,-z) \frac{1}{1-2^{-z+1}} \frac{Q(1)}{Q(2^{-z+1})}$$

First analyze $-B(N+1,-z)$:

$$\begin{aligned}
-B(N+1,-z) &= -\frac{\Gamma(N+1)\Gamma(-z)}{\Gamma(N+1-z)} \\
&= -\frac{N!(-z-1)!}{(N-z)!} \\
&= (-1)^N \frac{N!}{\prod_{k=0}^{N}(z-k)} \\
&= (-1)^N \frac{N!}{z\prod_{k=1}^{N} k\left(\frac{z}{k}-1\right)} \\
&= (-1)^N \frac{N!}{zN!\prod_{k=1}^{N}\left(\frac{z}{k}-1\right)} \\
&= \left((-1)^N z \prod_{k=1}^{N} -\left(1-\frac{z}{k}\right)\right)^{-1} \\
&= -\left(z \prod_{k=1}^{N}\left(1-\frac{z}{k}\right)\right)^{-1} \\
&= \left(z(z-1)\prod_{k=2}^{N}\left(1-\frac{z}{k}\right)\right)^{-1}
\end{aligned}$$

We want to approximate this by a Taylor series expansion. To do this the following lemma is helpful:

Lemma 10.1. *If $F(z) = \prod_{j\in R} \frac{1}{1-f_j(z)}$ for some index set R, then the Taylor series expansion of F at a, if it exists, is given by*

$$F(z) = F(a)\left(1 + \sum_{j\in R} \frac{f_j'(a)}{1-f_j(a)}(z-a) + O\left((z-a)^2\right)\right)$$

Proof.

$$G(z) = \prod_{j\in R} g_j(z)$$
$$G'(z) = \sum_{j\in R} g'_j(z) \prod_{k\in R\neq j} g_k(z)$$
$$\frac{G'(z)}{G(z)} = \frac{\sum_{j\in R} g'_j(z) \prod_{k\in R\neq j} g_k(z)}{\prod_{j\in R} g_j(z)} = \sum_{j\in R} \frac{g'_j(z)}{g_j(z)}$$

$$F(z) = F(a)\left(1 + \frac{F'(a)}{F(a)}(z-a) + O\left((z-a)^2\right)\right) = F(a) + F'(a)(z-a) + O\left((z-a)^2\right)$$

□

At $a = 1$ we have the expansion for the Beta function:

$$\begin{aligned} -B(N+1,-z) &= \frac{1}{1-z} z^{-1} \prod_{j=2}^{N} \left(1 - \frac{z}{j}\right)^{-1} \\ &= \frac{1}{1-z} N \left(1 + (H_{N-1} - 1)(z-1) + O\left((z-1)^2\right)\right) \\ &= \frac{N}{1-z} - N(H_{N-1} - 1) + O(z-1) \\ &= -\frac{N}{z-1} - N(H_{N-1} - 1) + O(z-1) \end{aligned}$$

with $H_{N-1} = \gamma + \ln N - O\left(\frac{1}{N}\right)$. So we get

$$-B(N+1,-z) = -\frac{N}{z-1} - N(\gamma + \ln N - 1) + O(z-1)$$

Approximation of $\frac{1}{1-2^{-z+1}}$ with the series expansion for $\frac{1}{e^x-1}$ leads to:

$$\begin{aligned} \frac{1}{1-2^{-z+1}} &= \frac{1}{1-e^{\ln 2(-z+1)}} = -\frac{1}{(-z+1)\ln 2} + \frac{1}{2} - \frac{-z+1}{12} + O\left((-z+1)^3\right) \\ &= \frac{1}{(z-1)\ln 2} + \frac{1}{2} + O(z-1) \end{aligned}$$

The missing part $\frac{Q(1)}{Q\left(2^{-z+1}\right)}$ is analyzed similarly to $-B(N+1,-z)$ by using the Taylor expansion for the Q function:

$$\begin{aligned} \frac{Q(1)}{Q(2^{-z+1})} &= Q(1) \prod_{j<1} \left(1 - 2^{-z+j}\right)^{-1} \\ &= 1 - \ln 2 \sum_{j<1} \frac{2^{j-1}}{1-2^{j-1}} (z-1) + O\left((z-1)^2\right) \\ &= 1 - \alpha \ln 2 (z-1) + O\left((z-1)^2\right) \end{aligned}$$

with $\alpha = 1 + \frac{1}{3} + \frac{1}{7} + \frac{1}{15} + ... \approx 1,606695$.
Connecting the analyzed parts, the integrand is approximately:

$$\begin{aligned} -B(N+1,-z)\frac{Q(1)}{Q(2^{-z+2})} &= \left(-\frac{N}{z-1} - N(H_{N-1} - 1) + O(z-1)\right) \\ &\times \left(\frac{1}{(z-1)\ln 2} + \frac{1}{2} + O(z-1)\right) \\ &\times \left(1 - \alpha \ln 2 (z-1) + O\left((z-1)^2\right)\right) \end{aligned}$$

The residue at $z = 1$ is the Laurent coefficient a_{-1} of $\frac{1}{z-1}$ in this product:

$$\begin{aligned} Res_{z=1} &= -\frac{N}{\ln 2}\left(H_{N-1}-1\right)+N\left(\alpha-\frac{1}{2}\right) \\ &= -N\lg N - N\left(\frac{\gamma-1}{\ln 2}-\alpha+\frac{1}{2}\right)+O(1) \end{aligned}$$

Then approximate the residues at the other poles $z = 1 \pm \frac{2\pi ik}{\ln 2}$:

$$\begin{aligned} Res_{z=1\pm\frac{2\pi ik}{\ln 2}} &= -\frac{1}{\ln 2}\sum_{k\neq 0} B\left(N+1,-1-\frac{2\pi ik}{\ln 2}\right) \\ B\left(N+1,-1-\frac{2\pi ik}{\ln 2}\right) &= \frac{\Gamma(N+1)\,\Gamma\left(-1-\frac{2\pi ik}{\ln 2}\right)}{\Gamma\left(N-\frac{2\pi ik}{\ln 2}\right)} \\ &= N\Gamma\left(-1-\frac{2\pi ik}{\ln 2}\right)\frac{\Gamma(N)}{\Gamma\left(N-\frac{2\pi ik}{\ln 2}\right)} \end{aligned}$$

Now the standard approximation formula for the Γ function is used to simplify this term.

$$\begin{aligned} \frac{\Gamma(N+1)}{\Gamma(N+1-\alpha)} &= N^{\alpha}\left(1+O\left(\frac{1}{N}\right)\right) \\ Res_{z=1\pm\frac{2\pi ik}{\ln 2}} &= N\Gamma\left(-1-\frac{2\pi ik}{\ln 2}\right)(N-1)^{\frac{2\pi ik}{\ln 2}}\left(1+O\left(\frac{1}{N}\right)\right) \\ &= N\Gamma\left(-1-\frac{2\pi ik}{\ln 2}\right)N^{\frac{2\pi ik}{\ln 2}}\left(1+O\left(\frac{1}{N}\right)\right) \\ &= N\Gamma\left(-1-\frac{2\pi ik}{\ln 2}\right)e^{\frac{2\pi ik\lg N}{\ln 2}}\left(1+O\left(\frac{1}{N}\right)\right) \end{aligned}$$

The sum of residues at the points $z = z = 1 \pm \frac{2\pi Ik}{\ln 2}$ is found to be

$$Res_{z=1\pm\frac{2\pi Ik}{\ln 2}} = -N\delta(N)+O(1)$$

where

$$\delta(N) = \frac{1}{\ln 2}\sum_{k\neq 0}\Gamma\left(-1-\frac{2\pi ik}{\ln 2}\right)e^{2\pi ik\lg N}$$

is a small oscillatory term. So finally we insert these results in (10.5) and get the following theorem:

Theorem 10.2. *The average internal path length of a digital search tree built from N records with keys from random bit stream is*

$$A_N = N\lg N + N\left(\frac{\gamma-1}{\ln 2}-\alpha+\frac{1}{2}+\delta(N)\right)+O\left(N^{\frac{1}{2}}\right)$$

Proof. This follows from the derivation above. □

10.3.4 External Internal Nodes

A property of trees of some interest is the number of internal nodes which have both links null. An alternate storage representation could be used for such nodes to save space.

Theorem 10.3. *The average number of nodes with both links null in a digital search tree built from N records with keys from random bit streams is*

$$N\left(\beta+1-\frac{1}{Q_\infty}\left(\frac{1}{\ln 2}+\alpha^2-\alpha\right)+\delta^*(N)\right)+O\left(N^{\frac{1}{2}}\right)$$

where the constants involved have the values
$\alpha = 1+\frac{1}{3}+\frac{1}{7}+\frac{1}{15}+\ldots\approx 1.606695\ldots,$
$Q_\infty = \frac{1}{2}\cdot\frac{3}{4}\cdot\frac{7}{8}+\ldots\approx .288788\ldots$ *and*
$\beta = \frac{1\cdot 2^2}{1}\left(\frac{1}{1}\right)+\frac{2\cdot 2^3}{1\cdot 3}\left(\frac{1}{1}+\frac{1}{3}\right)+\frac{3\cdot 2^4}{1\cdot 3\cdot 7}\left(\frac{1}{1}+\frac{1}{3}+\frac{1}{7}\right)+\ldots\approx 7.74313\ldots$
The function $\delta^*(N)$ is a periodic function in $\lg N$, with $|\delta^*(N)|<10^{-6}$. The approximate value of the coefficient of the leading term is $0.372046812\ldots$*.*

Proof. As before, we use a simple transform with generating functions to derive an explicit sum, then use Rice's method to evaluate this sum. The number of external internal nodes is

$$C_N=\sum_{k=0}^{\infty}\frac{1}{2^{N-1}}\binom{N-1}{k}\left(C_k+C_{N-1-k}\right),\qquad N\geq 2 \tag{10.6}$$

with $C_1=1$ and $C_0=0$. This follows from the fact that the number of nodes with both links null in a tree is exactly the sum of the numbers of such nodes in the two subtrees of the root while the tree has more than one node.
In terms of the exponential generating function $C(z)=\sum_{N=0}^{\infty}\frac{C_N z^N}{N!}$ by multiplying both sides by $\frac{z^{N-1}}{(N-1)!}$ and summing for $N\geq 1$, we have:

$$\begin{aligned}
C_1+\sum_{N=2}^{\infty}\frac{C_N z^{N-1}}{(N-1)!} &= 1+\sum_{N=2}^{\infty}\left(2\sum_{k=0}^{\infty}\frac{1}{2^{N-1}}\binom{N-1}{k}C_k\frac{z^{N-1}}{(N-1)!}\right)\\
\sum_{N=1}^{\infty}\frac{C_N z^{N-1}}{(N-1)!} &= 1+\sum_{N=2}^{\infty}\left(2\sum_{k=0}^{\infty}\frac{1}{2^{N-1}}\binom{N-1}{k}C_k\frac{z^{N-1}}{(N-1)!}\right)
\end{aligned}$$

This leads, similar to A_N, to the equation

$$C'(z)=1+2C\left(\frac{z}{2}\right)e^{\frac{z}{2}} \tag{10.7}$$

Again we introduce a new generating function $D(z)=\sum_{N=0}^{\infty}\frac{D_N z^N}{N!}$ defined by $D(z)=e^{-z}C(z)$ to get a somewhat more manageable form:

$$D'(z)+D(z)=e^{-z}+2D\left(\frac{z}{2}\right)$$

By the recurrence on the coefficients we get:

$$D_N+D_{N-1}=(-1)^{N-1}+\frac{1}{2^{N-2}}D_{N-1}$$

$$D_N=(-1)^{N-1}-\left(1-q^{N-2}\right)D_{N-1},\qquad N\geq 2 \tag{10.8}$$

with $D_1=1$, $D_0=0$.
We define the constant $q:=\frac{1}{2}$ (we will see that only this constant changes for M-ary trees). This recurrence is inhomogeneous, so we get a somewhat more complicated explicit form:

$$D_N=(-1)^{N-1}\sum_{i=1}^{N-1}\prod_{j=i}^{N-2}\left(1-q^j\right)$$

Rewriting this in terms of

$$\begin{aligned}
R_N &= \sum_{i=1}^{N} \prod_{j=i}^{N} \left(1 - \frac{1}{2^j}\right) \\
&= \sum_{i=1}^{N} \frac{\prod_{j=1}^{N} \left(1 - \frac{1}{2^j}\right)}{\prod_{j=1}^{i} \left(1 - \frac{1}{2^j}\right)} \\
&= Q_N \sum_{i=2}^{N} \frac{\prod_{j=1}^{N} \left(1 - \frac{1}{2^j}\right)}{\prod_{j=1}^{i} \left(1 - \frac{1}{2^j}\right)} \\
&= Q_N + \sum_{i=1}^{N} \frac{Q_N}{Q_i} \\
&= Q_N \left(1 + \sum_{k=1}^{N} \frac{1}{Q_k}\right)
\end{aligned}$$

and transforming like (10.2) back:

$$\begin{aligned}
D_N &= (-1)^N R_{N-2} C_N \\
&= N - \sum_{k=2}^{N} \binom{N}{k} D_N
\end{aligned}$$

We have the following explicit sum for the desired quantity:

$$C_N = N - \sum_{k=2}^{\infty} \binom{N}{k} (-1)^k R_{k-2} \tag{10.9}$$

This sum is similar to (10.4) but more difficult to evaluate because R_N is more complicated than Q_N. Because $R(z)$ does not extend R_N for positive integers, we get by Taylor expansion that $R_N = N + 1 - \alpha + R_N^*$. R_N^* satisfies a simple recurrence, converges very quickly and is polynomial bounded, so we extend R_N^* by the meromorphic function $R^*(z)$

$$\begin{aligned}
R_N^* &= \frac{(N + 1 - \alpha) q^{N+1}}{1 - q^{N+1}} + \frac{1}{1 - q^{N+1}} R_{N+1}^* \\
R^*(z) &= \sum_{i=2}^{\infty} \frac{(z + 1 + i - \alpha) q^{z+1+i}}{\prod_{j=0}^{i} (1 - q^{z+1+j})}
\end{aligned}$$

Substituting, we have

$$C_N = N - \sum_{k=2}^{\infty} \binom{N}{k} (-1)^k (R_{k-2}^* + k + 1 - \alpha)$$

After applying the elementary identities of Pascal's triangle

$$\sum_{k=0}^{\infty} \binom{N}{k} (-1)^k = \sum_{k=0}^{\infty} \binom{N}{k} \alpha (-1)^k = 0,$$

we have the simplified result

$$C_N = (N - 1)(\alpha + 1) - \sum_{k=2}^{\infty} \binom{N}{k} (-1)^k R_{k-2}^* \tag{10.10}$$

Now, by Theorem 10.1 and again looking only at the half-plane to the right of the line $z = \frac{1}{2}$, we know that

$$C_N - (N-1)(\alpha+1) = -\sum_z (-1)^N Res_z \left(B(N+1,-z) R^*(z-2)\right) + O\left(N^{\frac{1}{2}}\right) \tag{10.11}$$

In this case we look at the poles for $R^*(z-2)$ at $1 \pm \frac{2\pi i k}{\ln 2}$ and see that they are all single poles. The main term is given by $N \lim_{z\to 1} R^*(z-2)$; the poles for $k \neq 0$ add a small oscillatory term.

Lemma 10.2.

$$\sum_{n=0}^{\infty} \frac{u^n}{\prod_{k=1}^{n}(1-q^k)} = \frac{1}{\prod_{k=0}^{\infty}(1-q^k u)}$$

Proof. The coefficient of $u^n q^m$ on both sides is the number of ways to write n as the sum of m nonnegative integers. □

The method of calculating $R^*(-1)$ is to express $R^*(z)$ in terms of generating functions, which generalizes the function of Lemma 10.2, then to expand that function and exploit certain properties of its derivatives. Specifically, we define

$$F(u,v) = \sum_{j=1}^{\infty} \frac{q^j u^j}{\prod_{i=1}^{j}(1-q^i v)}$$

This is the generating function for restricted partitions, the coefficients of $u^n v^m q^k$ is the number of ways to partition k into m parts not exceeding n. By Lemma 10.2 we get:

$$F(u,1) = \sum_{n=0}^{\infty} \frac{q^n u^n}{\prod_{k=1}^{n}(1-q^k)}$$

$$F(u,1) + 1 = \sum_{n=0}^{\infty} \frac{u^n}{\prod_{k=1}^{n}(1-q^k)}$$

$$F(u,1) = \frac{1}{\prod_{k=0}^{\infty}(1-q^k u)} - 1$$

$$F(1,1) = Q_\infty^{-1} - 1.$$

Also we have

$$F_1'(u,1) = \frac{1}{\prod_{i=1}^{\infty}(1-q^i u)} \sum_{(k=1)}^{\infty} \frac{q^k}{(1-q^k u)}$$

so that $F_1'(1,1) = \frac{\alpha}{Q_\infty}$. Furthermore, we have

$$F\left(1, q^{z+1}\right) = \sum_{j=1}^{\infty} \frac{q^j}{\prod_{i=1}^{j}(1-q^{z+1+i})}$$

$$F_1'\left(1, q^{z+1}\right) = \sum_{j=1}^{\infty} \frac{j q^j}{\prod_{i=1}^{j}(1-q^{z+1+i})},$$

which finally gives the following expansion:

$$R^*(z) = \frac{q^{z+1}}{1-q^{z+1}} \left((z+1-\alpha)\left(F\left(1,q^{z+1}\right)+1\right) + F_1'\left(1,q^{z+1}\right)\right)$$

From this formulation, a Taylor expansion around $z=-1$ is straightforward:

$$\frac{q^{z+1}}{1-q^{z+1}} = \frac{1}{(z+1)\ln q} - \frac{1}{2} + O(z+1)$$

$$F\left(1,q^{z+1}\right) = F(1,1) + (z+1)\ln q F_2'(1,1) + O(z+1)^2$$
$$F_1'\left(1,q^{z+1}\right) = F_1'(1,1) + (z+1)\ln q F_{12}'(1,1) + O\left((z+1)^2\right),$$

so that

$$R(z) = -\frac{F(1,1)+1}{\ln q} + \alpha F_2'(1,1) - F_{12}''(1,1) + O(z+1)$$

$$F_2'(1,1) = \sum_{j=1}^{\infty}\left(\frac{q^j}{\prod_{i=1}^{j}(1-q^i)}\left(\sum_{k=1}^{j}\frac{q^k}{1-q^k}\right)\right) \tag{10.12}$$

$$F_{12}''(1,1) = \sum_{j=1}^{\infty}\left(\frac{jq^j}{\prod_{i=1}^{j}(1-q^i)}\left(\sum_{k=1}^{j}\frac{q^k}{1-q^k}\right)\right) \tag{10.13}$$

Actually, we can relate $F_2'(1,1)$ to α and Q_∞. Because

$$F_1'(1,1) = F_2'(1,1) + F(1,1)$$

there is $F_2'(1,1) = \frac{\alpha-1}{Q_\infty}+1$. There does not seem to be an easy way to express $F_{12}''(1,1)$ in terms of α and Q_∞, so we denote that constant simply by β. Collecting terms, we have shown that the residue of the integrand at $z=1$ is

$$N\left(\beta + 1 - \frac{1}{Q_\infty}\left(\alpha^2 - \alpha - \frac{1}{\ln q}\right)\right)$$

It remains to calculate the residues of the integrand at the other singularities.
This calculation is straightforward: the residue of $\frac{1}{1-q^{z+1}}$ at $z = -1 \pm \frac{2\pi ik}{\ln q}$ is $-\frac{1}{\ln q}$, and the other terms in $R^*(z)$ contribute a factor of $\frac{2\pi ik}{Q_\infty \ln q}$. The factor for $B(N+1,-z)$ is expanded exactly as in the preceding Taylor expansion for the internal path length; thus we have the oscillatory term

$$\delta^*(N) = \frac{1}{Q_\infty \ln q}\sum_{k\neq 0}\frac{2\pi ik}{\ln q}\Gamma\left(-1-\frac{2\pi ik}{\ln q}\right)e^{2\pi ik \lg N}$$

This completes the calculation of the coefficient of the linear term. □

10.4 Digital Search Tries

10.4.1 Data Structure of Tries

Definition 10.10. A digital search trie is a digital tree for storing a set of strings in which there is one node for every prefix of every string in the set.

The name of this data structure comes from the word re<u>trie</u>val. The word retrieval is stressed, because a trie has a lookup time that is equivalent to the length of the string being looked up. Again we represent the strings as keys in binary form. It may be convenient to assume that the strings are all of same (binary) length, but the method is also appropriate for varying length strings, if no string is a prefix of another.
Digital search tries compared to trees have much improved worst case performance. Their average case performance is asymptotically optimal. If N records with keys from random bit streams are inserted into an initially empty trie, then the average number of nodes examined during successful search reads as

$$\lg N + \frac{\gamma}{\ln 2} + \frac{1}{2} + \delta(N) + O\left(\frac{1}{N}\right)$$

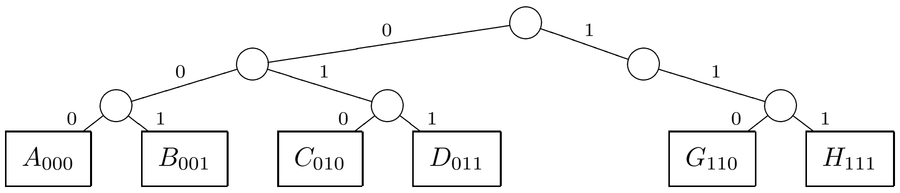

Figure 10.5: Example for a digital search trie with external path length = 18

10.4.2 Data Structure of Patricia Tries

You can optimize the performance of a trie constructed with N keys by ensuring that this trie has just $N-1$ internal nodes by collapsing one-way branches on internal nodes and get the so called Patricia tries:

Definition 10.11. A Patricia tree is defined as a compact representation of a digital search trie where all nodes with one child are merged with their parent.

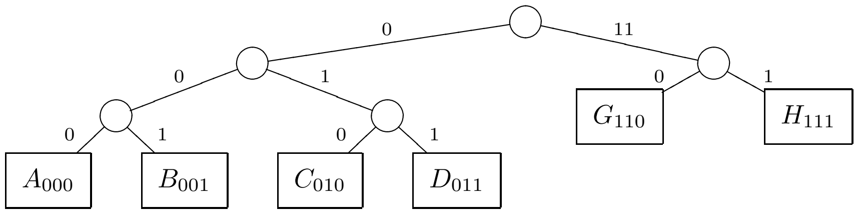

Figure 10.6: Example for a Patricia trie

The average number of nodes examined during successful search is one less than for standard tries.

10.4.3 External Path Length

Definition 10.12. The external path length of a tree is the sum of the depth of every leaf of the tree.

The fundamental recurrence for the average external path length of a binary trie is

$$A_N^{[T]} = N + \sum_{k=0}^{\infty} \frac{1}{2^N} \binom{N}{k} \left(A_k^{[T]} + A_{N-k}^{[T]} \right), \qquad N \geq 2 \tag{10.14}$$

with $A_0^{[T]} = A_1^{[T]} = 0$. This is the number of nodes examined during all successful searches. Note that since no key is stored at the root, the subtrees have a total of N keys. The resulting functional equation on the exponential generating function is not a difference-differential but simply a difference equation:

$$A^{[T]}(z) = z\left(e^z - 1\right) + 2A^{[T]}\left(\frac{z}{2}\right) e^{z-2}$$

It is still convenient to transform the equation with $A(z) = e^z B(z)$ to get the equation

$$B^{[T]}(z) = z\left(1 - e^{-z}\right) + 2B^{[T]}\left(\frac{z}{2}\right)$$

This yields directly

$$B^{[T]}(z) = \frac{N(-1)^N}{1 - \left(\frac{1}{2}\right)^{N-1}}$$

and

$$A_N^{[T]} = \sum_{k=2}^{\infty} \binom{N}{k} (-1)^k \frac{k}{1 - \left(\frac{1}{2}\right)^{k-1}}$$

This can be handled directly by Rice's method or Mellin transform techniques, as described in full detail in [Szp00].
The fundamental recurrence for the average external path length of a Patricia trie is

$$A_N^{[P]} = N \left(1 - \frac{1}{2^{N-1}}\right) + \sum_{k=0}^{\infty} \frac{1}{2^N} \binom{N}{k} \left(A_k^{[P]} + A_{N-k}^{[P]}\right), \qquad N \geq 1 \qquad (10.15)$$

with $A_0^{[P]} = 0$. The external path length is the sum of the ones of the subtries of the root plus the number of nodes in the subtries (N) unless one of the subtries is empty which has the probability $\frac{1}{2^{N-1}}$. The resulting functional equation on the exponential generating function is

$$A^{[P]}(z) = z\left(e^z - e^{\frac{z}{2}}\right) + 2A^{[P]}\left(\frac{z}{2}\right) e^{\frac{z}{2}}$$

with transformed version

$$B^{[P]}(z) = z\left(1 - e^{-\frac{z}{2}}\right) + 2B^{[P]}\left(\frac{z}{2}\right)$$

which yields directly

$$B^{[P]}(z) = \frac{N(-1)^N}{2^{N-1} - 1}$$

and

$$A_N^{[P]} = \sum_{k=2}^{\infty} \binom{N}{k} \frac{k(-1)^k}{2^{k-1} - 1} = A_N^{[T]} - N$$

Given the result for binary tries the average external path length for Patricia tries is obvious.

10.4.4 External Internal Nodes

The average number of internal nodes with both sons external are computed for Patricia tries. The derivation is similar to the one for digital search trees, so we only sketch it here. We start with the recurrence

$$C_N^{[P]} = \sum \frac{1}{2^N} \binom{N}{k} \left(C_k^{[P]} + C_{N-k}^{[P]}\right), \qquad N \geq 3 \qquad (10.16)$$

with $C_0^{[P]} = C_1^{[P]} = 0$ and $C_2^{[P]} = 1$. This corresponds to the functional equation

$$C^{[P]}(z) = \left(\frac{z}{2}\right)^2 + 2C^{[P]}\left(\frac{z}{2}\right) e^{\frac{z}{2}}$$

which transforms to

$$D^{[P]}(z) = \left(\frac{z}{2}\right)^2 e^{-z} + 2D^{[P]}\left(\frac{z}{2}\right)$$

and eventually gives the sum

$$C_N^{[P]} = \frac{1}{4} \sum_{k=2}^{N} \binom{N}{k} (-1)^k \frac{k(k-1)}{1 - \frac{1}{2}^{k-1}}$$

Knuth gives specific evaluations of such sums. The eventual result is that the proportion of nodes in Patricia tries with both sons external is $\frac{1}{4 \ln 2} = .3606...$ plus a small oscillatory term. Thus, according to this measure, digital search trees are (slightly) more balanced than Patricia tries.

10.5 Multiway Branching

Above only binary trees have been analyzed. But we can generalize the average analysis to M-ary trees, where each node contains M links to other nodes, numbered from 0 to $M-1$. It turns out that the analysis given above survives largely intact for the M-ary case. For example, to find the average number of nodes in a M-ary digital search tree with all links null, we begin with the fundamental recurrence:

$$C_N^{[M]} = \sum_{k_1+k_2+...+k_M=N-1} \frac{1}{M^{N-1}} \binom{N-1}{k_1,k_2,...,k_M} \left(C_{k_1}^{[M]} + C_{k_2}^{[M]} + ... + C_{k_M}^{[M]}\right), \quad N \geq 2 \qquad (10.17)$$

with $C_1^{[M]} = 1$ and $C_0^{[M]} = 0$. The argumentation is the same as for the digital search tree. The number of nodes with all links null in a tree is exactly the number of such nodes in all the subtrees of the root, unless the tree has just one node. Again the partitions of $N-1$ nodes without the root into M subtrees weighted by all possibilities are examined. All the subtrees are randomly built according to the same model.
By symmetry, (10.17) is equivalent to

$$C_N^{[M]} = M \sum_{k_1+k_2+...+k_M=N-1} \frac{1}{M^{N-1}} \binom{N-1}{k_1,k_2,...,k_M} C_{k_1}^{[M]}, \qquad N \geq 2$$

with $C_1^{[M]} = 1$ and $C_0^{[M]} = 0$. Now we introduce the exponential generating function $C^{[M]}(z) = \sum_{N=0}^{\infty} \frac{C_N^{[M]} z^N}{N!}$ and derive the following difference-differential equation:

$$C^{[M]\prime}(z) = 1 + MC^{[M]}\left(\frac{z}{M}\right)\left(e^{\frac{z}{M}}\right)^{M-1} \qquad = 1 + MC^{[M]}\left(\frac{z}{M}\right)\left(e^{\left(1-\frac{1}{M}\right)z}\right) \qquad (10.18)$$

For $M = 2$, this is exactly the equation derived from (10.7); moreover none of the manipulations used for solving it depend in an essential way on the value of that constant.

Corollary 10.1. *The average number of nodes with all links null in an M-ary search tree (for $M \geq 2$) built from N records with keys from random bit streams is*

$$N\left(\beta^{[M]} + 1 - \frac{1}{Q_\infty^{[M]}}\left(\frac{1}{\ln M} + {\alpha^{[M]}}^2 - \alpha^{[M]}\right) + \delta^{[M]}(N)\right) + O\left(N^{\frac{1}{2}}\right)$$

where the constants involved are given by
$\alpha^{[M]} = \sum_{k=1}^{\infty} \frac{1}{M^k-1}$,
$Q_\infty^{[M]} = \prod_{k=1}^{\infty}\left(1 - \frac{1}{M^k}\right)$,
$\beta^{[M]} = \sum_{k=1}^{\infty} \frac{kM^{k+1}}{\prod_{i=1}^{k}\left(M^i-1\right)} \sum_{j=1}^{k} \frac{1}{M^j-1}$
and the oscillatory term is
$\delta^{[M]}(N) = \frac{1}{Q_\infty \ln M} \sum_{k\neq 0} \frac{2\pi ik}{\ln M} \Gamma\left(-1 + \frac{2\pi ik}{\ln M}\right) e^{2\pi ik \lg N}$.

10.6 General Framework

The methods that we have used in the previous sections can be applied to study many other properties of digital trees. If $X(T)$ and $x(T)$ are parameters of trees satisfying

$$X(T) = \sum_{subtrees\, T_j\, of\, the\, root\, of\, T} X(T_j) + x(T) \qquad (10.19)$$

then the exponential generating functions for the expectations X_N and x_N for an M-ary digital search tree built from N records with keys from random bit streams satisfy

$$X'(z) = MX\left(\frac{z}{M}\right)e^{\left(1-\frac{1}{M}\right)z} + x(z)$$

This is derived in exactly the same manner as (10.18). Now in terms of the generating functions $Y(z) = e^{-z}X(z)$ and $y(z) = e^{-z}x(z)$ this becomes

$$Y'(z) + Y(z) = MY\left(\frac{z}{M}\right) + y'(z) + y(z) \tag{10.20}$$

This leads to a nonlinear recurrence like (10.8) satisfied by Y_N, with the solution sought given by $X_N = \sum_{k=0}^{N}\binom{N}{k}Y_k$. If the quantity $(-1)^k Y_k$ is sufficiently well behaved, we can study its asymptotics and find a function Y_k^* which

(i) is simply related to Y_k so that $\sum_{k=0}^{N}\binom{N}{k}\left(Y_k - (-1)^k Y_k^*\right)$ is easily evaluated,

(ii) satisfies a recurrence of the form $Y_{N+1}^* = (1 - g(M,N))\,Y_N^* + f(M,N)$,

(iii) goes to zero quickly as $N \to \infty$.

Depending on the nature of $g(M,N), f(M,N)$ and the speed of convergence, conditions (ii) and (iii) may allow the recurrence to be turned around to extend Y_N^* to the complex plane and so allow the desired expectation to be computed by evaluating the sum $\sum_{k=0}^{N}\binom{N}{k}\left(Y_k - (-1)^k Y_k^*\right)$ as detailed in the previous sections.

The same type of generalization applies to the study of tries (and Patricia tries), and the simpler nature of the recurrences follows through the generalization. For example, the exponential generating functions for the expectations X_N and x_N of parameters of trees satisfying (10.19) for a random trie built from N records from random bit stream is

$$X(z) = MX\left(\frac{z}{M}\right)e^{\frac{z}{M}} + x(z) \tag{10.21}$$

which is considerably easier to deal with. The equation can be solved by Rice's method and also by Mellin transform techniques.

This general framework allows quite full analysis of the types of trees considered, and they clearly expose the fundamental differences and similarities among the analyses.