SRC TR 87-214

Analysis of Tries that Store Prefixing Keys

by

Pilar de la Torre

Analysis of Tries that Store Prefixing Keys

Pilar de la Torre

Systems Research Center

The University of Maryland

College Park, MD 20742

ABSTRACT

Tries are data structures for representing sets of string keys. Although tries can be readily adapted to represent unrestricted sets of keys, the analysis of tries built from sets that include *prefixing keys* (which are keys that are prefixes of other keys in the set) appears to have remained untouched until now. The exact expectations of the time and space requirements of the retrieval algorithms of several trie varieties built from unrestricted finite sets of keys are computed in this paper. We present a unified approach to the derivation of these expectations which are computed with respect to a probabilistic model that takes into account sample sets containing prefixing keys.

1. Introduction

Tries are classical data structures (see, for example, [delaB59, Fre60, Sus63, Gwe68, Mor68, Knu73, HS76, RND77, GG78, Mel79, Nie81, Sta80, Sed83, AHU83, Gon84]) for representing sets of string keys, and their analysis (see, for example, [Knu73b, Fra77, Tra78, Reg81, Dev82, Fla83, Pla83, Gon84, Fla84, FRS85, FS86, delaT87a]) has been carried out with respect to several probabilistic models, among which we find the *Bernoulli model* $\mathcal{I}$ and the *finite identical length keys model* $\mathcal{F}$. The sample space of the model $\mathcal{I}$ consists of n infinitely long keys composed of characters that are assumed to be taken, uniformly and independently, from a fixed finite alphabet $\mathcal{A}$ (see, for example, [Knu73b, Dev82, FS86, Gon86, Pla84, delaT87a]). The sample space of the model $\mathcal{F}$ consists of the n element sets of keys of length h composed from $\mathcal{A}$ (see, for example, [Fra77, Tra78, Pla84, delaT87a]). All the sets within these two models, as well as all other models used for tries analysis in the past, satisfy the *no–prefixing–key* restriction: no key in a sample set is a prefix of another.

In recent work [Kno86], Knott has presented detailed algorithms for the handling of sets of string keys that may include *prefixing keys* (that is, keys that prefixes of other keys in the set) and has designed three trie structures for this purpose: *full doubly–chained prefix tries*, *compact doubly–chained prefix tries*, and *patrician doubly–chained prefix tries.*

Although a way of getting around the no–prefixing–key restriction has been part of the programming folklore (namely by ending each key with a special symbol, the *endmarker*, to mark the key's end) the analysis of tries built from sets that include prefixing keys seems to have received no attention so far. The present work addresses this question by computing the exact average time and space required by the retrieval algorithms of

the three above mentioned varieties of doubly–chained prefix tries. We also present three additional trie variants which are natural adaptations of classical trie constructions for the purpose of handling prefixing keys. These are *full endmarker tries*, *compact endmarker tries* and and *patrician endmarker tries.* We compute the exact average space and average time complexities for retrieval in each of the six above mentioned trie varieties with respect to a probabilistic model that takes into account sets of keys that contain prefixing keys. The sample space of this model, the *prefix model*, consists of the n element sets of string keys of length no greater than h that can be composed from a fixed finite alphabet; all such sets are assumed to be equally probable. We present a unified approach to the calculation of expectations of random variables belonging to a certain wide class with respect to the prefix model.

Our approach is based on recurrence equations. It shares some features with the approaches to trie analyses for the models $\mathcal{I}$ and $\mathcal{F}$ developed using recurrences by Knuth in [Knu73a], by Trabb Pardo in [Tra78], and by the author in [delaT87a], and also with the approach through generating functions by Flajolet, Regnier, and Sotteau in [FRS85]. The systematic generating function approach of [FRS85] can be also extended to the prefix model and we have done this work in [delaT87b].

In this paper, section §2 introduces the logical tree structure, the *endmarker prefix tree*, underlying the construction of the tries structures to be analyzed. Section §3 presents the different kinds of endmarker tries and doubly–chained prefix tries as implementations of the endmarker prefix tree. Section §4 describes the unified method of calculating the expectations of random variables of interest with respect to the prefix model. Section §5 computes the expectations of the space and time required by the retrieval algorithms of each of the six trie varieties; these expectations are recorded in tables II and V. Section §6 contains concluding remarks.

2. The endmarker prefix tree

We begin by discussing the logical tree that underlies the trie data structures. Let s be a finite set of strings composed from the totally ordered alphabet $\mathcal{A} = \{a_1, \ldots, a_m\}$, where the characters are ranked according to $rank(a_i) = i$, $1 \leq i \leq m$. The set $pref(s) \equiv \{x \mid xz \in s\}$ of prefixes of the elements of s supports a natural m–ary tree structure. This tree will be called the *prefix tree* of s and denoted by $t(s)$. The set of nodes of $t(s)$ is $pref(s)$, and the i–th subtree of the node $x \in pref(s)$, $1 \leq i \leq m$, consists of those strings of $pref(s)$ that begin with xa_i, i.e. $\{xa_iw \mid xa_iw \in s\}$ (compare Figure 1). The root of $t(s)$ is the length zero string which will be denoted by ε.

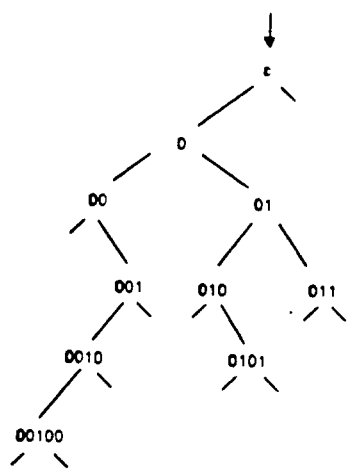


Figure 1. Prefix tree built from the set of keys $s = \{00100, 0101, 011\}$.

There is a correspondence between the paths on $t(s)$ and the strings with symbols in $\mathcal{A}$. To a length zero path corresponds the length zero string ε; to the (possibly infinitely long) path $p = v_1, v_2, \ldots$, where v_{i+1} is the

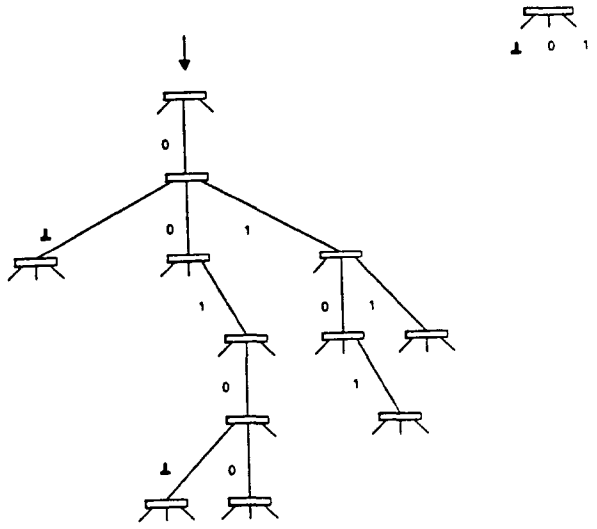


Figure 2. Endmarker prefix tree built from $s = \{00100, 0101, 011, 0010, 0\}$.

l_i–th son of v_i $(i > 1)$, corresponds $x = a_{l_1} a_{l_2} \ldots \in s$. This correspondence defines an injective mapping between the maximal paths of $t(s)$ and the keys of s. This mapping is bijective precisely when s satisfies the no–prefixing–key restriction. A *trie* is the generic name used for data structures that implement the prefix tree.

An unrestricted finite set s of strings composed from $\mathcal{A}$ can be easily modified enabling the prefix tree to yield a representation of s as the set of maximal paths of an $(m+1)$–ary tree, which is constructed as follows. Let $\mathit{prefixingkeys}(s) = \{x \in s \mid x \in \mathit{pref}(s - \{x\})\}$ be the set of prefixing keys of s, and let $\perp$ be a symbol not belonging to $\mathcal{A}$. In the set

$$s[\perp] \equiv (s - \mathit{prefixingkeys}(s)) \cup \{x\perp \mid x \in \mathit{prefixingkeys}(s)\}$$

no key is a prefix of another. The $(m+1)$–ary prefix tree $t(s[\perp])$ with respect to the extended alphabet $\mathcal{A}^{\perp} \equiv \{\perp\} \cup \mathcal{A}$, where the endmarker symbol $\perp$ is ranked according to $\mathit{rank}(\perp) = 0$, is called the *endmarker prefix tree* of s (Compare Figure 2).

The next section presents the six trie varieties analyzed in this paper as implementations of the endmarker prefix tree.

3. The trie structures

Let $r_1, \ldots, r_n$ be a collection of items where each item comprises a *key* part (which uniquely identifies it) and a *data* part. Let us further assume that the keys are strings composed from a finite ordered alphabet $\mathcal{A}$ which, after appropriate identifications, we can write as $\mathcal{A} = \{1, \ldots, m\}$. We shall construct six tree data structures for storing the above collection of items, all of which are based on the string properties of the keys. These trie structures are implementations of the endmarker prefix tree $t(s[\perp])$ built with the set s of the keys belonging to the items. They can be classified into two groups: *endmarker tries*, where an internal node v of $t(s[\perp])$ is represented as an array of pointers to v's subtrees, and *doubly-chained tries*, where such a node is represented by the linked list of pointers to v's non-empty subtrees. Within each of these two groups we shall consider three variants: *full, compact* and *patrician.*

3.1 *Endmarker tries*

Endmarker tries are natural adaptations of classical trie constructions for the purpose of storing sets that may include prefixing keys. For a set s such that $prefixingkeys(s) = \emptyset$, the full endmarker trie and the compact endmarker tries built from s revert (except for the additional null pointer field corresponding to the endmarker character $\perp$ at the internal nodes) to the original tries of de la Brandais [delaB59] and Fredkin [Fre60]; the patrician

endmarker trie reverts to Trabb Pardo's [Tra78] version of the PATRICIA storage–saving tries [Gwe68, Mor68].

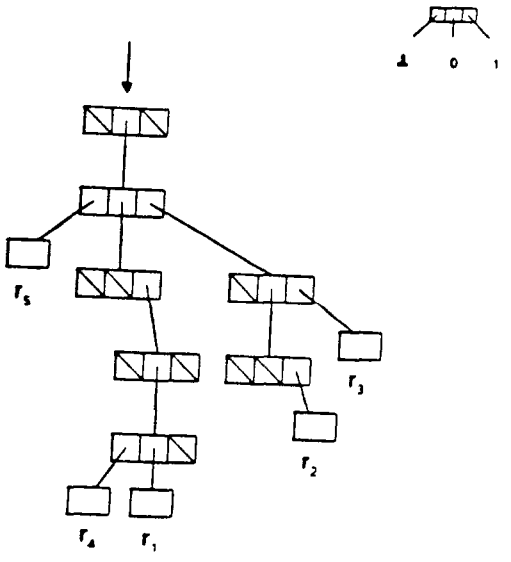
Full endmarker tries are suitable for storing finite length keys, while the compact and patrician versions can also accommodate infinitely long keys. Suppose that every key in s has finite length. Since the maximal paths of the endmarker prefix tree $t(s[\perp])$ are then of finite length, the terminal nodes of $t(s[\perp])$ bijectively correspond to the keys of s.

In the *full endmarker trie* built with s, denoted by $t^{fe}(s)$, a nonterminal node v of $t(s[\perp])$ is represented by an array $children(v)[1 : m+1]$ of pointers to its children. A terminal node v of $t(s[\perp])$, with corresponding key $k \in s$, is represented in $t^{fe}(s)$ by a pointer $data(v)$ to the data of the item whose key is k (compare Figure 3(a)).

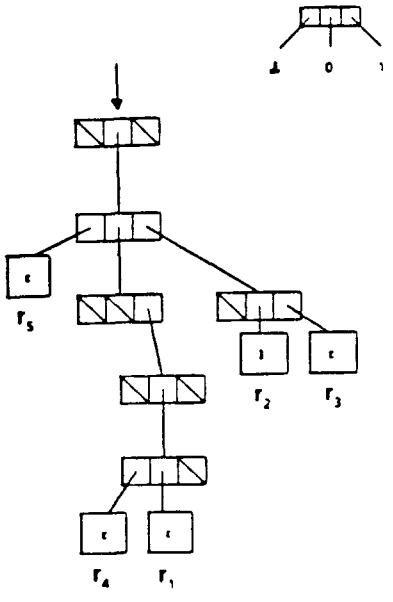
To search for a key k in $t^{fe}(s)$, we start at the root node and proceed recursively. If the root is a nonterminal node, we proceed as follows: if $k = \varepsilon$, we search for k in the first subtree; if $k = iz$, with $i \in \mathcal{A}$, we search for z in the $(i+1)$–th subtree. Otherwise, the search ends; it succeeds precisely when the root is a terminal node and $k = \varepsilon$.

The potentially wasteful use of space of this representation, for nodes with relatively few nonempty subtrees, motivates the following two pruning strategies (as well as the construction of the doubly–chained version of tries discussed later in §2.2).

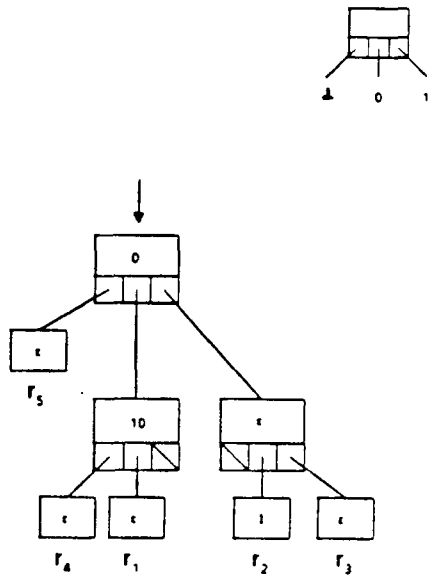
Now suppose that s is an arbitrary finite set of keys with characters in $\mathcal{A}$. The *compact endmarker trie* of s will be denoted by $t^{ce}(s)$. Its construction is similar to that of the full endmarker tries, except that now the branching within $t(s[\perp])$ is stopped at subtrees consisting of nodes with out–degree less than or equal to 1. Such a subtree T is collapsed into a single data node which takes as its label the string corresponding to the the unique maximal path of T.



(a) full



(b) compact



(c) patrician

Figure 3. Endmarker tries for the items $r_1, \ldots, r_5$ with respective keys $k_1 =$ 00100, $k_2 =$ 0101, $k_3 =$ 011, $k_4 =$ 0010, $k_5 =$ 0. The alphabet is $\{\perp, 0, 1\}$, with $\perp < 0 < 1$.

In the compact endmarker trie $t^{ce}(s)$, a node v of $t(s[\perp])$ with out–degree greater than 1 is represented as for full endmarker tries. On the other hand, suppose that T is a subtree of such a node v, and that every node of T has out–degree ≤ 1. Then T has a single maximal path; let y be the string corresponding to this path, and let $k \in s$ be the key corresponding to the maximal path of $t(s[\perp])$ that contains T. In $t^{ce}(s)$, T is collapsed into a single terminal node w which is represented by a structure consisting of two pointer fields: the field $suffix(w)$ points to y and the field $data(w)$ points to the data of the item whose key is k. The search algorithm for $t^{ce}(s)$ is analogous to that for full endmarkers tries except that, on reaching a terminal node w of $t^{ce}(s)$, the label of w must be compared to the current search key; only if they are equal is the search successful.

The *patrician endmarker trie* built from s is denoted by $t^{pe}(s)$ (compare Figure 3(c)). It results from the compact endmarker trie $t^{ce}(s)$ by collapsing each chain of nodes of out–degree 1 onto the only son, v say, of the node of highest depth in that chain (the root has depth 0). A pointer field $label(v)$, which points to the string corresponding to the collapsed chain, is added to the structure representing v.

The search for a key k in $t^{pe}(s)$ can be recursively described. We start at the root node, which we denote by r and its label by l. If r is a nonterminal node, we proceed as follows: if l is not a prefix of k, the search ends unsuccessfully; if $k = l$, we search for k in the first subtree; if $k = liz$, with $i \in \mathcal{A}$, we search for z in the $(i+1)$–th subtree. Otherwise, the search ends; it is successful precisely when the root is a terminal node and its label equals k.

Remark. Recursive structural definitions for each of the above endmarker tries will be provided later in §5.1.

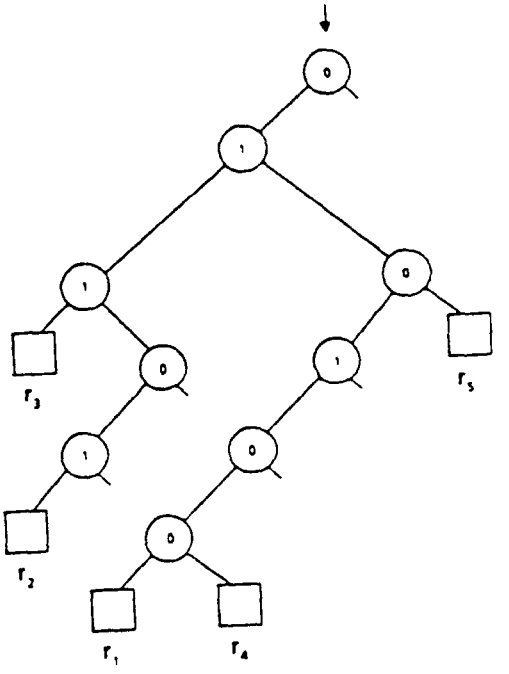
3.2 *Doubly–chained prefix tries*

We shall now consider full, compact and patrician doubly–chained prefix tries, which were presented in [Kno86] for storing a finite collection of items whose set of keys s may include prefixing keys. If no key of s is a prefix of another, the full and compact doubly–chained prefix tries built from s revert to the doubly–chained tries of de la Brandais [delaB59] and Sussenguth [Sus63]. The patrician doubly–chained prefix trie is Knott's doubly–chained version of PATRICIA tries [Gwe68, Mor68].

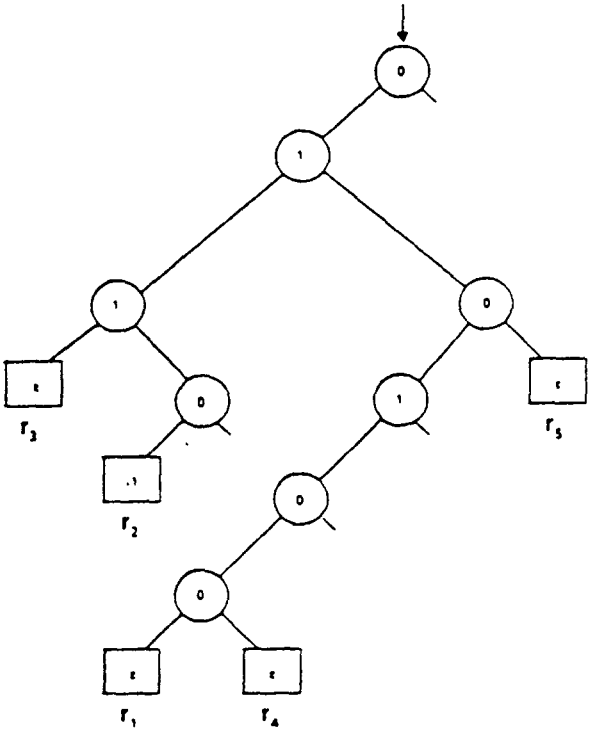
Doubly–chained prefix tries can be thought of as implementations of a modified version of the binary tree representation of the forest of subtrees of the endmarker prefix tree $t(s[\perp])$. This binary tree, denoted by $t^{bin}(s[\perp])$, is constructed as follows. A node v of $t(s[\perp])$ with non–empty subtrees $T_{l_1}, \ldots, T_{l_q}$ in order, $l_1 < \ldots < l_q$, is represented by a sequence $(v_q, l_q), \ldots, (v_1, l_1)$ of labeled nodes of $t^{bin}(s[\perp])$, where l_i is the label of node v_i. The right son of v_i is v_{i-1}, $l \geq i \geq 2$, and the right subtree of v_1 is empty. The left subtree of v_i is the binary tree representation of the forest of subtrees of T_{l_i}, $1 \leq i \leq q$.

Doubly–chained prefix tries are implementations of a pruned version of $t^{bin}(s[\perp])$ that will be denoted by $\bar{t}(s[\perp])$ and is constructed as follows. The tree $\bar{t}(s[\perp])$ results from $t^{bin}(s[\perp])$ by removing each node v with label $\perp$, while inserting v's only son (which is a terminal node) in its place. The bijective correspondence from maximal paths of $t(s[\perp])$ to keys in s translates as follows: Maximal paths of $\bar{t}(s[\perp])$ ending at right son terminal nodes bijectively correspond to keys in $\mathit{prefixingkeys}(s)$, while the remaining maximal paths bijectively correspond to keys in $s - \mathit{prefixingkeys}(s)$.

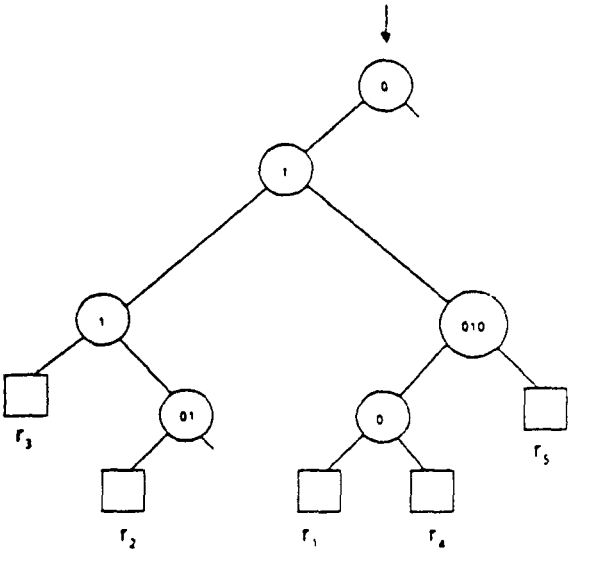
The *full doubly–chained prefix trie* built from a finite set of finite length keys s is denoted by $t^{fd}(s)$. In $t^{fd}(s)$, a nonterminal node v of $\bar{t}(s[\perp])$ is



(a) full



(b) compact



(c) patrician

Figure 4. Doubly–chained prefix tries for the items $r_1, \ldots, r_5$ with respective keys $k_1 = 00100$, $k_2 = 0101$, $k_3 = 011$, $k_4 = 0010$, $k_5 = 0$.

represented by a structure with a field $char(v)$ which holds v's label, and two additional pointer fields: $left(v)$ points to v's left subtree and $right(v)$ to its right subtree. A terminal node v of $\bar{t}(s[\perp])$, corresponding to a key $k \in s$, is represented by the pointer $data(v)$ to the data of the item whose key is k. (Compare Figure 4(b)).

In order to search for k in $t^{fd}(s)$, we start at the root node and proceed recursively. If the root is an nonterminal node, which we denote by r and its label c, we proceed as follows: if c is not a prefix of k, we search for k in the right subtree; if $k = cz$, we search for z in the left subtree. Otherwise, the search ends; it is successful precisely when the root is a terminal node and $k = \varepsilon$.

The *compact doubly-chained prefix trie* can be built from an arbitrary finite set s of keys over $\mathcal{A}$ and will be denoted by $t^{cd}(s)$ (compare Figure 4(b)). Its construction is similar to that of the full doubly-chained prefix trie, except that a left-link branching of $\bar{t}(s[\perp])$ is stopped as soon as a node is reached whose left subtree has only nodes of out-degree less than or equal to 1. In $t^{cd}(s)$, a node v of $\bar{t}(s[\perp])$ with out-degree greater than 1 is represented as for full doubly-chained prefix tries. On the other hand, let T be a left subtree of such node v and suppose that T has no node of out-degree greater than 1 (T is a path where all nodes other than the root are left sons). Let x be the concatenation of the labels of the nodes on the path connecting the root to v, and let y the concatenation of the labels of on the path T. Then $k \equiv xy \in s$, and T is represented by a single terminal node w of $t^{ce}(s)$. This node w consists of a structure with two pointer fields: $suffix(w)$ points to y, and $data(w)$ points the data of the item whose key is k.

To search for a key k in $t^{cd}(s)$, we proceed as we did for the full doubly-chained prefix tries except that, upon reaching a terminal node w, the label

of w must be compared to the current search ley; precisely when they are equal is the search successful.

The *patrician doubly–chained prefix trie*, denoted by $t^{pd}(s)$, results from the compact doubly–chained prefix trie $t^{cd}(s)$ by collapsing each maximal left–link chain into the parent of its node of lowest depth. This transformation can be effected by the following reiterated pruning of $t^{ce}(s)$: If a node v has an empty right subtree and v is the left son of its parent node $parent(v)$, then v is removed; the left subtree of v is then inserted as the left subtree of $parent(v)$, and the concatenation of the labels of $parent(v)$ and v becomes the new label of $parent(v)$. This process is reapplied to the resulting tree until no such transformation is possible.

To search for a key k in $t^{pd}(s)$, we start at the root node and proceed recursively. If the root is a nonterminal node, which we denote by r and its label by l, we proceed as follows: if l is not a prefix of k, the search ends unsuccessfully; if $k = l$, we search for k in the right subtree; if $k = lz$, we search for z in the left subtree. Otherwise, the search ends: it is successful precisely when the root is a terminal node and $k = \varepsilon$.

Remark. Recursive structural definitions for each of the above doubly–chained prefix tries will be provided later in §5.2.

4. Recurrences and averages for the prefix model

In this section we develop the main tool, *the root function method*, for computing expectations of certain random variables of interest with respect to the prefix model.

4.1 *The prefix model*

The sample space of the *prefix model* consists of a class of sets of keys described by three parameters: the common size n of the key sets, the maximum length h of the keys that are allowed in these sets, and the size m of the underlying alphabet $\mathcal{A} = \{1, \ldots, m\}$. For a nonnegative integer h, the set of all strings of length less than or equal to h composed from the alphabet $\mathcal{A} = \{1, \ldots, m\}$ will be denoted by $\mathcal{A}^{[h]}$, i.e. $\mathcal{A}^{[h]} \equiv \mathcal{A}^0 \cup \mathcal{A}^1 \cup \ldots \mathcal{A}^h$. (Note that $\mathcal{A}^{[h]}$ equals the set of prefixes of the strings in $\mathcal{A}^h$.) The sets of finite and infinite length strings over $\mathcal{A}$ will be respectively denoted by $\mathcal{A}^*$ and $\mathcal{A}^\infty$, and $\mathcal{A}^\circledast \equiv \mathcal{A}^* \cup \mathcal{A}^\infty$.

Given the integers h, n, m, with $h, n \geq 0$ and $m \geq 2$, the probability space for the prefix model consists of the n element subsets of $\mathcal{A}^{[h]}$, all of which are assumed to be equally probable.

If B is a finite set, $|B|$ will denote the number of elements in B, and $\mathcal{R}_n(B)$ the set of its n element subsets. It can be readily seen that

$$m^{[h]} \equiv |\mathcal{A}^{[h]}| = \frac{m^{h+1} - 1}{m - 1}, \qquad \text{and} \qquad |\mathcal{R}_n(\mathcal{A}^{[h]})| = \binom{m^{[h]}}{n}.$$

The expectation of a mapping X defined on the n element subsets $\mathcal{A}^{[h]}$, denoted by $E_{hn}[X]$, equals $E_{hn}[X] = \binom{m^{[h]}}{n}^{-1} \sum_{s \in \mathcal{R}_n(\mathcal{A}^{[h]})} X(s)$. The sum

$$N_{hn}[X] = \sum_{s \in \mathcal{R}_n(\mathcal{A}^{[h]})} X(s) = \binom{m^{[h]}}{n} E_{hn}[X],$$

will be called the *normalized expectation* of X.

4.2 *Expectations and root functions*

Throughout the remainder of section §4, $X(s)$ will denote the values of a real–valued function X defined on the finite subsets $s \subset \mathcal{A}^*$. To such a function X we associate the mapping

$$x(s) \equiv X(s) - \sum_{i \in \mathcal{A}} X(s_i), \tag{1}$$

which will be called the *root function* of X.

THEOREM 1. *Let x be the root function of X. If $X(\emptyset) = 0$, then $\sum_{z \in \mathcal{A}^*} x(s_z) = X(s)$. In general we have*

$$\frac{1}{1-m} x(\emptyset) + \sum_{z \in \mathcal{A}^*} \left[x(s_z) - x(\emptyset) \right] = X(s), \tag{2}$$

and x is the unique function with this property.

Proof: First we assume that $X(\emptyset) = 0$. This implies $x(\emptyset) = (1-m)X(\emptyset) = 0$ and $x(\{\varepsilon\}) = X(\{\varepsilon\})$. Making use of equation (1), the claimed expression of $X(s)$ in terms of $x(s)$ can be checked by induction on $l \equiv max(\{length(y) \mid y \in s\})$, the maximum length of a string in s. In the case of a general function X, (2) can be deduced from the case just dealt with. It suffices to take $Y(s) \equiv X(s) - X(\emptyset)$, and to notice the relation $y(s) = x(s) - (1-m)X(\emptyset)$, which follows from (1). Furthermore, if (2) is taken as the definition of $X(s)$, we can deduce that its root function must be equal to x. Thus, the root of X is the only function satisfying (2). $\square$

Our intention is to express $E_{hn}[X]$ in terms of the expectation $E_{hn}[x]$ of the root function x of X. The desired expression for $E_{hn}[X]$ will then be obtained by solving the recurrence relation for $E_{hn}[X]$ provided by the following theorem.

THEOREM 2. *Let x be the root function of X. Then $N_{0n}[X] = \delta_{0,n} X(\emptyset) + \delta_{1,n} X(\{\varepsilon\})$ and*

$$N_{hn}[X] = N_{hn}[x] + m \sum_{k \geq 0} \binom{m^h}{n-k} N_{h-1k}[X] \qquad (h \geq 1). \tag{3}$$

Proof: We first verify (3) under the assumption $X(\emptyset) = 0$, which implies $x(\emptyset) = 0$ and $x(\{\varepsilon\}) = X(\{\varepsilon\})$. This verification can be done by induction on $h \geq 0$. For $h = 0$ we have $\mathcal{A}^{[h]} = \{\varepsilon\}$ and $\mathcal{R}_n(\mathcal{A}^{[h]}) = \emptyset$ for $n \geq 2$, whereby

$$N_{hn}[X] = \sum_{s \in \mathcal{R}_n(\mathcal{A}^{[h]})} X(s) = \delta_{1,n} x(\{\varepsilon\}).$$

Assume $h \geq 1$. With the aid of Theorem 1 we can write

$$N_{hn}[X] = \sum_{s \in \mathcal{R}_n(\mathcal{A}^{[h]})} \sum_{w \in \mathcal{A}^*} x(s_w)$$

$$= N_{hn}[x] + \sum_{i \in \mathcal{A}} \sum_{0 \leq k \leq n} Q(i,k),$$

where

$$Q(i,k) = \sum_{\substack{s \in \mathcal{R}_n(\mathcal{A}^{[h]}) \\ |s_i| = k}} \sum_{z \in \mathcal{A}^*} x((s_i)_z).$$

For each $\tilde{s} \in \mathcal{R}_k(\mathcal{A}^{[h-1]})$, $h \geq 1$, the number of sets $s \in \mathcal{R}_n(\mathcal{A}^{[h]})$ such that $s_i = \tilde{s}$ equals $\binom{|\mathcal{A}^{[h]} - i\mathcal{A}^{[h-1]}|}{n-k}$. Making use of Lemma 1 we find

$$Q(i,k) = \binom{m^h}{n-k} \sum_{\tilde{s} \in \mathcal{R}_k(\mathcal{A}^{[h-1]})} \sum_{z \in \mathcal{A}^*} x(\tilde{s}_z) = \binom{m^h}{n-k} N_{h-1k}[X],$$

which concludes the proof in the case of $X(\{\emptyset\}) = 0$.

The case of a general function X can be reduced to the case just dealt applied to the function $Y(s) \equiv X(s) - X(\emptyset)$. It suffices to note that $x(\emptyset) = (1-m)X(\emptyset)$, $y(s) = x(s) - x(\emptyset)$, and to make use of the *Vandermonde* convolution

$$\sum_{r \geq k \geq 0} \binom{a}{r-k} \binom{b}{k} = \binom{a+b}{r}. \tag{4}$$

$\square$

Remark. The connection between the expectation of a cost function and the expectation of its root function, expressed either by means recurrences or through generating functions, also underlies the analyses of tries with respect to other models (see, for example, [Knu73, Tra78, FRS85, FS86, delaT87a]).

The following lemma furnishes the solution of recurrence (3) in terms of the values of the independent term $N_{hn}[x]$.

LEMMA 3. *The unique solution to the recurrence*

$$y(0,n) = f(0,n) \qquad (n \geq 0),$$
$$y(h,n) = f(h,n) + m \sum_{k \geq 0} \binom{g(h)}{n-k} y(h-1,k) \qquad (h \geq 1, n \geq 0),$$

where $f(x,y)$ and $g(x)$ are real-valued functions, can be calculated from the independent term $f(h,n)$ by

$$y(h,n) = \sum_{0 \leq i \leq h} m^i \sum_{k \geq 0} \binom{\sum_{j=0}^{i-1} g(h-j)}{n-k} f(h-i,k). \tag{5}$$

Proof: It can be verified by induction on $h \geq 1$ [delaT87a]. □

Solving recurrence (3) with the help of Lemma 3 we arrive at the following expression of $N_{hn}[X]$ in terms of $N_{hn}[x]$.

THEOREM 4. *If x is the root function of X then*

$$N_{hn}[X] = \sum_{\substack{0 \leq j \leq h \\ 0 \leq k}} m^j \binom{m^{[h]} - m^{[h-j]}}{n-k} N_{h-jk}[x], \tag{6}$$

where $N_{0k}[x] = \delta_{0,k}(1-m)X(\emptyset) + \delta_{1,k}[X(\{\varepsilon\}) - mX(\emptyset)]$.

Proof: The claimed expression of $N_{0k}[x]$ follows from (1). Applying Lemma 3, wherein we take $g(h) = m^h$ and $f(h,n) = N_{hn}[x]$, to the recurrence (3) yields

$$N_{hn}[X] = \sum_{0\le i\le h} m^i \sum_{k\ge 0} \binom{\sum_{j=0}^{i-1} g(h-j)}{n-k} N_{h-ik}[x].$$

Since $\sum_{0\le j\le i-1} g(h-j) = m^h + \cdots + m^{h-i+1} = m^{[h]} - m^{[h-i]}$, with $m^{[h]} = \frac{m^{h+1}-1}{m-1}$, the theorem follows. $\square$

Consider the function $Z(s) = |s|$, for instance. We have $Z(s) = z(s) + Z(s_1) + \ldots + Z(s_m)$ with

$$z(s) = \begin{cases} 1, & \text{if } \varepsilon \in s; \\ 0, & \text{otherwise.} \end{cases}$$

Substitution of $N_{hn}[z] = \binom{m^{[h]}-1}{n-1}$ in (6) followed by the application of (4) yields

$$\begin{aligned} N_{hn}[Z] &= \sum_{\substack{0\le j\le h \\ 0\le k}} m^j \binom{m^{[h]} - m^{[h-j]}}{n-k} \binom{m^{[h-j]}-1}{k-1} \\ &= m^{[h]} \binom{m^{[h]}-1}{n-1}, \end{aligned}$$

which is as expected.

A nontrivial application of Theorem 4 may be found in the following computation of the average value of $Tl(s) \equiv \sum_{x\in s} l(x)$, the total sum of the lengths of the strings in an n element set $s \subseteq \mathcal{A}^{[h]}$.

THEOREM 5. *The average value of $Tl(s)$ over the n element subsets $s \subseteq \mathcal{A}^{[h]}$, $0 \le n \le m^{[h]}$, equals*

$$E_{hn}[Tl] = n\Big[h - \frac{1}{m-1}\Big(1 - \frac{h+1}{m^{[h]}}\Big)\Big].$$

Proof: We have $Tl(s) = tl(s) + Tl(s_0) + \ldots + Tl(s_m)$ with $tl(s) = |s - \{\varepsilon\}|$. By direct counting we find

$$\begin{aligned} N_{hn}[tl] &= (n-1)\binom{m^{[h]}-1}{n-1} + n\binom{m^{[h]}-1}{n} \\ &= mm^{[h-1]}\binom{m^{[h]}-1}{n-1}. \end{aligned}$$

Since $Tl(\{\varepsilon\}) = Tl(\emptyset) = 0$, substituting $N_{hn}[tl]$ in (6) and making use of (4) we arrive at

$$\begin{aligned} E_{hn}[Tl] &= \frac{1}{\binom{m^{[h]}}{n}} \sum_{0 \le j < h} m^j \sum_{k \ge 0} \binom{m^{[h]} - m^{[h-j]}}{n-k} \binom{m^{[h-j]}-1}{k-1} mm^{[h-j-1]} \\ &= \frac{\binom{m^{[h]}-1}{n-1}}{\binom{m^{[h]}}{n}} \sum_{0 \le j < h} m^{j+1} m^{[h-j-1]}. \end{aligned}$$

The claimed expectation readily follows. □

4.3 *Limit of the expectations as $h \to \infty$ with n fixed*

It is possible to establish a relationship between the prefix model and the Bernoulli model $\mathcal{I}$. As described in §1, the sample space of $\mathcal{I}$ consists of sets of n infinitely long keys composed of characters that are assumed to be taken, uniformly and independently, from the fixed alphabet $\mathcal{A} = \{1, \ldots, m\}$.

Let X a function of the finite subsets of $\mathcal{A}^*$. Let Y be a function of finite subsets $s \subset \mathcal{A}^\infty$, and $y(s) = Y(s) - Y(s_1) - \ldots - Y(s_m)$ for all such s. We assume that $X(s) = 0$ and $Y(s) = 0$ when $|s| \le 1$. Also, let $E_{hn}[x]$ be the expectation of the root function x of X with respect to the prefix model, and $E_n[y]$ the expectation of y with respect to the model $\mathcal{I}$. We further assume that $E_n[y]$ is finite. Under these assumptions, the following theorem holds.

THEOREM 6. If $\lim_{h\to\infty} E_{hn}[x] = E_n[y]$ for every fixed n, then

$$\lim_{h\to\infty} E_{hn}[X] = E_n[Y]. \tag{7}$$

Proof: Since we assumed $X(s) = \emptyset$ for $|s| \leq 1$, Theorem 2 implies that $\alpha_{hn} \equiv E_{hn}[X]$ is the unique solution to the recurrence

$$\begin{aligned} \alpha_{0n} &= 0 \\ \alpha_{hn} &= E_{hn}[x] + m \sum_{m^{[h-1]} \geq k \geq 0} \chi(h,n,k)\alpha_{h-1k} \qquad (h \geq 1), \end{aligned} \tag{8}$$

$\chi(h,n,k) = \frac{\binom{m^h}{n-k}\binom{m^{[h-1]}}{k}}{\binom{m^{[h]}}{n}}$. By our assumptions, $E_{h1}[X] = 0$ forl all h. Letting $h \to \infty$ in recurrence (8), while n remains fixed, we obtain the recurrence

$$\begin{aligned} a_0 &= a_1 = 0 \\ a_n &= E_n[y] + m^{1-n} \sum_{k\geq 0} \binom{n}{k} (m-1)^{n-k} a_k \qquad (n \geq 2), \end{aligned} \tag{9}$$

which is thus satisfied by $a_n = \lim_{h\to\infty} E_{hn}[X]$. On the other hand, from Knuth's recurrences approach to trie analysis for the model I [Knu73b], we deduce that $a_n = E_n[Y]$ is the unique solution to (9). The two solutions must then coincide and the lemma follows. □

Remark. An analogous relationship holds between the model I and the model $\mathcal{F}$ (see, for example, [Tra78, delaT87a]).

4.4 *Expectations of suffix–cardinality dependent functions*

Theorem 4 reduces the computation of the expectation of X to determination of the expectation of its root function x. For some mappings (as seen earlier for Tl, for instance), the root function is simple enough so that its expectation can be easily determined by direct counting. A more systematic approach to the calculation the expectations of root functions is often possible.

For many cost functions of interest, the value $x(s) = X(s) - X(s_1) - \cdots - X(s_m)$ depends only on the cardinalities of the sets of suffixes $s \cap \{\varepsilon\}, s_1, \ldots, s_m$ (these sets are related to the partition $s = (s \cap \{\varepsilon\}) \cup 1s_1 \cup \ldots \cup ms_m$ induced by the root of the endmarker prefix tree $t(s[\perp])$). In other words, there is a function $\rho_X(n_0, \ldots, n_m)$ such that $x(s) = \rho_X(|s \cap \{\varepsilon\}|, |s_1|, \ldots, |s_m|)$. Theorem 1 then implies

$$X(s) = \frac{1}{1-m}\rho(0,0,\ldots,0) + \sum_{z \in \mathcal{A}^*} \left[\rho_X(|s_z \cap \{\varepsilon\}|, |s_{z1}|, \ldots, |s_{zm}|) - \rho_X(0,0,\ldots,0)\right].$$

We will say that such a mapping X is *suffix–cardinality dependent* and call ρ_X its *counting root function.* The mapping $Tl(s) \equiv \sum_{x \in s} l(x)$, for instance, is suffix–cardinality dependent and $\rho_{Tl}(n_0, n_1, \ldots, n_m) = n_1 + n_2 + \cdots + n_m$ is its counting root function. It may be noted that there is a bijective correspondence between the suffix–cardinality dependent functions X with a given value of $X(\emptyset)$, and the mappings $\rho(n_0, \ldots, n_m)$ of nonnegative integers n_i, $1 \le i \le m$, and $n_0 = 0$ or 1. This suggests introducing the operator

$$\mathcal{N}_{hn}[\rho] = \sum_{\substack{n_0 + \cdots + n_m = n \\ n_0 = 0,1}} \prod_{1 \le i \le m} \binom{m^{[h-1]}}{n_i} \rho(n_0, \ldots, n_m) \qquad (h \ge 1), \quad (9)$$

whose intuitive meaning emerges from the following lemma.

LEMMA 7. *Let $h \ge 1$ and $1 \le n \le m^{[h]}$. If $x(s) = \rho(|s_0|, \ldots, |s_m|)$ then $E_{hn}[x] = \mathcal{N}_{hn}[\rho]$.*

Proof: For each tuple $(n_0, n_1, \ldots, n_m)$ such that $n = n_0 + n_1 + \cdots + n_m$, where $n_1, \ldots, n_m$ nonnegative integers and $n_0 = 0$ or 1, there are exactly $\prod_{1 \le i \le m} \binom{m^{[h-1]}}{n_i}$ sets $s \subseteq \mathcal{A}^{[h]}$ such that $|s \cap \{\varepsilon\}| = n_0$ and $|s_i| = n_i$, $1 \le i \le m$. The lemma follows. $\square$

THEOREM 8. *If $\rho_X(n_0, n_1, \ldots, n_m)$ is the counting root function of X then*

$$E_{hn}[X] = m^h \Big[\Big(1 - \frac{n}{m^{[h]}}\Big) X(\emptyset) + \frac{n}{m^{[h]}} X(\{\varepsilon\})\Big] + \sum_{\substack{0 \le j < h \\ 0 \le k}} m^j \tau\big(m^{[h]}, m^{[h-j]}, n, k\big) \mathcal{N}_{h-jk}[\rho_X],$$

where $\tau(a,b,c,d) = \frac{\binom{a-b}{c-d}}{\binom{a}{c}}$ *and* $0 \le n \le m^{[h]}$.

Proof: Since $E_{hn} = \binom{m^{[h]}}{n}^{-1} N_{hn}[X]$, the claimed expectation can be deduced from Theorem 4 and Lemma 7. $\square$

Theorem 8 enables us to compute the expectation of X provided that we know the value of $\mathcal{N}_{hn}[\rho_X]$ as a function of h and n. The values of $\mathcal{N}_{hn}$ recorded in Table I follow immediately from (9), and so does the relation

$$\mathcal{N}_{hn}[f(n)\rho_1 + g(n)\rho_2] = f(n)\mathcal{N}_{hn}[\rho_1] + g(n)\mathcal{N}_{hn}[\rho_2].$$

This relation, and the elementary values recorded in Table I, will suffice to deduce the values of $\mathcal{N}_{hn}[\rho]$ for those ρ that will emerge from our calculations.

Table I: Elementary values of $\mathcal{N}_{hn}$

$\rho(n_0, \ldots, n_m)$	$\mathcal{N}_{hn}[\rho], \quad h \geq 1$
1	$\binom{m^{[h]}}{n}$
δ_{0,n_i}, $i \neq 0$	$\binom{m^h}{n}$
δ_{n,n_i}, $i \neq 0$	$\binom{m^{[h-1]}}{n}$
n_i, $i \neq 0$	$\binom{m^{[h]}-1}{n} m^{[h-1]}$
$n_j \delta_{0,n_i}$, $0 \neq i \neq j \neq 0$	$\binom{m^h-1}{n-1} m^{[h-1]}$
n_0	$\binom{m^{[h]}-1}{n-1}$
$n_0 \delta_{n,n_i}$, $i \neq 0$	0
$n_0 \delta_{0,n_i}$, $i \neq 0$	$\binom{m^h-1}{n-1}$
$f(n)\rho_1 + g(n)\rho_2$	$f(n)\mathcal{N}_{hn}[\rho_1] + g(n)\mathcal{N}_{hn}[\rho_2]$

where $f(n)$ and $g(n)$ are any two real–valued functions.

4.5 *Average number of proper prefixes in a random set*

We shall illustrate our approach by calculating the average value of $P(s) = |prefixingkeys(s)|$, the number of keys in s which are prefixes of other keys in that set. (Also, the functions $P(s)$ and $Tl(s)$ will be helpful for expressing the relationships among the different kinds of tries, and will add intuitive meaning to the expectations of the trie cost functions to be computed later.) If $ppref(s)$ denotes the set of proper prefixes of the elements of s, then $|ppref(s)| = |\{x \mid |s_x| \geq 2\}|$ and $P(s) = |s \cap ppref(s)|$. We have $k \in s \cap ppref(s)$ precisely when either $k = \varepsilon$, or $k = iz$ and $z \in s_i \cap ppref(s_i)|$ for some $i \in \mathcal{A}$. Thus,

$$s \cap ppref(s) = \begin{cases} \emptyset & \text{if } |s| \leq 1, \\ (\{\varepsilon\} \cap s) \cup \bigcup_{i \in \mathcal{A}} i(s_i \cap ppref(s_i)) & \text{otherwise,} \end{cases}$$

which implies $|\emptyset \cap ppref(s)| = 0$ and

$$|s \cap ppref(s)| = |\{\varepsilon\} \cap s|(1 - \delta_{|s|,1}) + \sum_{i \in \mathcal{A}} |s_i \cap ppref(s_i)|.$$

Therefore the counting root function of $P(s)$ is $\rho_P(n_0, \ldots, n_m) = n_0(1 - \delta_{n,1})$, with $n = \sum_{0 \leq j \leq m} n_j$.

THEOREM 9. *The average value of $P(s)$ over the n element subsets $s \subseteq \mathcal{A}^{[h]}$ equals*

$$E_{hn}[P] = n\frac{m^{[h-1]}}{m^{[h]}} - \sum_{0 \leq j < h} m^j \tau\big(m^{[h]}, m^{[h-j]}, n, 1\big),$$

where $\tau(a,b,c,d) = \frac{\binom{a-b}{c-d}}{\binom{a}{c}}$ *and* $0 \leq n \leq m^{[h]}$.

Proof: We have $P(\emptyset) = P(\{\varepsilon\}) = 0$. With the aid of Table I we deduce

$$\begin{aligned} \mathcal{N}_{hn}[\rho_P] &= \mathcal{N}_{hn}[n_0(1 - \delta_{n,1})] \\ &= (1 - \delta_{n,1})\mathcal{N}_{hn}[n_0] = (1 - \delta_{n,1})\binom{m^{[h]} - 1}{n - 1}. \end{aligned}$$

Substituting this expression in the expectation formula of Theorem 8, and after applying (4), we arrive at

$$E_{hn}[P] = \frac{1}{\binom{m^{[h]}}{n}} \sum_{0 \le j < h} m^j \sum_{k \ge 0} \binom{m^{[h]} - m^{[h-j]}}{n-k} \binom{m^{[h-j]} - 1}{k-1} (1 - \delta_{k,1})$$

$$= \frac{1}{\binom{m^{[h]}}{n}} \sum_{0 \le j < h} m^j \left[\binom{m^{[h]} - 1}{n-1} - \binom{m^{[h]} - m^{[h-j]}}{n-1} \right],$$

which implies the claimed value of $E_{hn}[P]$. □

5. The analysis of tries

Each of the tries defined in §3 is a tree, generically denoted by $g(s)$, with two kinds of nodes: *internal* nodes and *data* nodes. The internal nodes are the nonterminal nodes of $g(s)$ and hold the string matching information that drives the search algorithm. The data nodes are the terminal nodes of $g(s)$; each terminal node corresponds to a key of s and holds information about the location of the data of the item identified by this key. The retrieval algorithm traverses the unique path connecting the root and the data node corresponding to the search key. Thus, for a space and time cost analysis, the underlying structure of interest is that of a *t–ary tree with data nodes.*

DEFINITION 1: A t–ary tree with data nodes (t is a positive integer) is either empty (denoted Λ), a single 'data' node (denoted $\mathcal{D}$), or an 'internal' root node plus a sequence $g_1, \ldots, g_t$ of t–ary trees with data nodes, where g_i is called the i–th subtree, $1 \le i \le t$. For $t = 2$ we have a *binary tree with data nodes*, where g_1 is called the *left subtree* and g_2 the *right subtree.* The *total data node path length* of such a tree g is defined as the sum

$$tdpl(g) = \sum_{all\ data\ nodes\ d\ in\ g} depth(d)$$

where $depth(d)$ is the number of edges on the path connecting the root and d (the number of internal nodes on that path equals $depth(d)$).

The number of internal nodes in the full, compact and patrician endmarker tries $t^{fe}(s)$, $t^{ce}(s)$, and $t^{pe}(s)$ will be respectively denoted by $Sfe(s)$, $Sce(s)$, and $Spe(s)$; their respective total data node path length by $Tfe(s)$, $Tce(s)$, and $Tpe(s)$. Analogously, the number of internal nodes of the full, compact and patrician doubly–chained prefix tries $t^{fd}(s)$, $t^{cd}(s)$, and $t^{pd}(s)$ will be respectively denoted by $Sfd(s)$, $Scd(s)$, and $Spd(s)$; their respective total data node path length by $Tfd(s)$, $Tcd(s)$, and $Tpd(s)$.

For each one of these twelve cost functions we shall compute the exact expected value, over the n element subsets $s \subseteq \mathcal{A}^{[h]}$, as a function of n, h and the alphabet size m. Each of these computations will proceed as follows. We first formulate a recursive structural definition for the particular trie variety. This definition implies recursive expressions for the trie cost functions from which the counting root functions are readily apparent. The desired expectations of the cost functions are then derived from the general formula provided by Theorem 8.

Remark. Recursive structural definitions for tries have been presented in [Fra77], and in [FRS85] where they are also used in deriving root functions of the trie cost functions.

5.1 *Analysis of endmarker tries*

DEFINITION 2: The *full endmarker trie* built from a finite set of keys $s \subset \mathcal{A}^*$ is the $(m+1)$–ary tree with data nodes, denoted by $t^{fe}(s)$, which is recursively defined as follows:

(i) If s is empty, $t^{fe}(s)$ is equal to the empty tree Λ.

(ii) If $s = \{\varepsilon\}$, $t^{fe}(s)$ is equal to a single 'data' node $\mathcal{D}$.

(*iii*) Otherwise, $t^{fe}(s)$ is the $(m+1)$–ary tree with data nodes having an 'internal' root node whose subtrees are $t^{fe}(s \cap \{\varepsilon\}), t^{fe}(s_1), \ldots, t^{fe}(s_m)$ in order.

Recursive expressions for $Sfe(s)$ and $Tfe(s)$, which are the number of internal nodes and the total data node path length of $t^{fe}(s)$, follow:

$$Sfe(s) = 1 - \delta_{|s|,0} - \delta_{s,\{\varepsilon\}} + \sum_{i \in \mathcal{A}} Sfe(s_i),$$

$$Tfe(s) = |s|(1 - \delta_{s,\{\varepsilon\}}) + \sum_{i \in \mathcal{A}} Tfe(s_i).$$

The corresponding counting root functions are

$$\rho_{Sfe}(n_0, n_1, \ldots, n_m) = 1 - \delta_{n,0} - n_0 \delta_{n,1},$$
$$\rho_{Tfe}(n_0, n_1, \ldots, n_m) = n(1 - n_0 \delta_{n,1}),$$

where $n = n_0 + \ldots + n_m$. From the properties of $\mathcal{N}_{hn}$ recorded in Table 1 we deduce

$$\mathcal{N}_{hn}[\rho_{Sfe}] = \binom{m^{[h]}}{n}(1 - \delta_{n,0}) - \delta_{n,1},$$
$$\mathcal{N}_{hn}[\rho_{Tfe}] = m^{[h]}\binom{m^{[h]} - 1}{n - 1} - \delta_{n,1},$$

which substituted in the expectation formula of Theorem 8 yields

$$E_{hn}[Sfe] = \sum_{0 \le j < h} m^j \left[1 - \tau\left(m^{[h]}, m^{[h-j]}, n, 0\right) - \tau\left(m^{[h]}, m^{[h-j]}, n, 1\right)\right],$$

$$E_{hn}[Tfe] = \sum_{0 \le j < h} m^j \left[\frac{n}{m^{[h]}} m^{[h-j]} - \tau\left(m^{[h]}, m^{[h-j]}, n, 1\right)\right].$$

A recursive definition of the compact doubly–chained prefix trie $t^{cd}(s)$ can be obtained from Definition 2 by considering arbitrary finite subsets $s \subset \mathcal{A}^{\circledast}$ while replacing condition (*ii*) by the following 'compaction' condition:

Table II: Endmarker Tries

X	$\rho_X(n_0,\ldots,n_m)$	$\mathcal{N}_{hn}[\rho_X]$
Sfe	$1-\delta_{n,0}-n_0\delta_{n,1}$	$\binom{m^{[h]}}{n}(1-\delta_{n,0})-\delta_{n,1}$
Sce	$1-\delta_{n,0}-\delta_{n,1}$	$\binom{m^{[h]}}{n}(1-\delta_{n,0}-\delta_{n,1})$
Spe	$(1-\delta_{n,0}-\delta_{n,1})\big[1-\sum_{1\le i\le m}\delta_{n,n_i}\big]$	$\Big[\binom{m^{[h]}}{n}-m\binom{m^{[h-1]}}{n}\Big](1-\delta_{n,0}-\delta_{n,1})$
Tfe	$n(1-\delta_{n,1}n_0)$	$m^{[h]}\binom{m^{[h]}-1}{n-1}-\delta_{n,1}$
Tce	$n(1-\delta_{n,1})$	$m^{[h]}\binom{m^{[h]}-1}{n-1}(1-\delta_{n,1})$
Tpe	$n(1-\delta_{n,1})\Big[1-\sum_{1\le i\le m}\delta_{n,n_i}\Big]$	$m^{[h]}\binom{m^{[h]}-1}{n-1}-mm^{[h-1]}\binom{m^{[h-1]}-1}{n-1}(1-\delta_{n,1})$

where $n=\sum_{0\le j\le m}n_j$. Note that $X(\{\varepsilon\})=X(\emptyset)=0$.

($ii*$) If $s=\{x\}$, $t^{ce}(s)$ is equal to a single 'data' node $\mathcal{D}$ with a label equal to x.

This definition implies recursive expressions for $Sce(s)$ and $Tce(s)$, the number of internal nodes and the total data node path length of $t^{ce}(s)$, from which their counting root functions immediately follow. These root functions, together with their corresponding values of $\mathcal{N}_{hn}$, are recorded in Table II. These values of $\mathcal{N}_{hn}$ and Theorem 8 yield the desired values of $E_{hn}[Sce]$ and $E_{hn}[Tce]$. The outcome of these calculations appears in Table III, which is given at the end of this subsection.

DEFINITION 3: The *patrician endmarker trie* built from a finite set of keys $s \subset \mathcal{A}_m^{\circledast}$ is the $(m+1)$–ary tree with data nodes, denoted by $t^{pe}(s)$, which is recursively defined as follows:

(*i*) If $s = \emptyset$, $t^{pe}(s)$ is equal to the empty tree Λ.

(*ii*) If $s = \{x\}$, $t^{pe}(s)$ is equal to a single 'data' node $\mathcal{D}$ with a label equal to x.

(*iii*) Otherwise, let x be the longest common prefix shared by all the keys in s. Then, $t^{pe}(s)$ is the $(m+1)$–ary tree with data nodes having an 'internal' root node with a label equal to x and subtrees $t^{pe}(s_x \cap \{\varepsilon\}), t^{pe}(s_{x1}), \ldots, t^{pe}(s_{xm})$ in order.

Once again the expressions of the counting root functions of *Spe* and *Tpe*, the number of internal nodes and the total data node path length of $t^{pe}(s)$, follow immediately from the recursive expressions of these cost functions resulting from Definition 3. These root functions and their corresponding values of $\mathcal{N}_{hn}$, computed using the values from Table I of §4, are recorded above in Table II. The expectations $E_{hn}[Spe]$ and $E_{hn}[Tpe]$, computed by means of Theorem 8, appear in Table III given below.

Remark. Some observations can be made on the relationship among the cost functions of the different endmarker tries. We have that $Sce(s) - Spe(s)$ is equal to the number of nodes of $t^{ce}(s)$ that have out-degree one. If $s \subset \mathcal{A}^*$, we further have $Sfe(s) - Sce(s) = Tfe(s) - Tce(s)$, and this quantity is equal to the number of nodes of $t^{fe}(s)$ that have out–degree one and only one data node among their descendants.

We close the analysis of endmarker tries with a summary of the expectations computed so far. Taking $\tau(a,b,c,d) = \frac{\binom{a-b}{c-d}}{\binom{a}{c}}$, the expectations emerging

from our analyses can be expressed in terms of a few primitives, the *basic trie sums for the prefix model*, which are

$$\sigma[a,b] \equiv \sigma[a,b;h,n] \equiv \sum_{a \le j < h+a} m^j \tau(m^{[h]} - a, m^{[h-j]}, n, b), \tag{10}$$

$$\kappa[a,b] \equiv \kappa[a,b;h,n] \equiv m^h \sum_{a \le j < h+a} \tau(m^{[h]} - a, m^{[h-j]}, n, b), \tag{11}$$

$$\theta[a,b] \equiv \theta[a,b;h,n] \equiv \sum_{0 \le j < h} m^j \tau(m^{[h]} - a, m^{h-j}, n, b), \tag{12}$$

$$\eta[a,b] \equiv \eta[a,b;h,n] \equiv m^h \sum_{0 \le j < h} \tau(m^{[h]} - a, m^{h-j}, n, b). \tag{13}$$

Table III: Endmarker tries

X	$E_{hn}[X]$
Tl	$n\left[h - \frac{1}{m-1}\left(1 - \frac{h+1}{m^{[h]}}\right)\right]$
P	$\frac{n}{m}\left[1 - \frac{1}{m^{[h]}}\right] - \sigma[0,1]$
Sfe	$m^{[h-1]} - \sigma[0,0] - \sigma[0,1]$
Sce	$E_{hn}[Sfe] - \frac{m}{m-1}\left(\kappa[0,1] - \sigma[0,1]\right)$
Spe	$m^{[h-1]}\left(1 - \frac{n}{m^{[h]}}\right) - m\,\theta[0,0] + (m-1)\,\sigma[0,0] + E_{hn}[P]$
Tfe	$E_{hn}[Tl] + E_{hn}[P]$
Tce	$E_{hn}[Tfe] - \frac{m}{m-1}\left(\kappa[0,1] - \sigma[0,1]\right)$
Tpe	$E_{hn}[Tl] + E_{hn}[P] + \frac{m}{m-1}\left(\theta[1,1] - \eta[1,1]\right)$

5.2 *Analysis of doubly–chained prefix tries*

We shall now compute the expectations of the cost functions of the doubly–chained prefix tries $t^{fd}(s)$, $t^{cd}(s)$, and $t^{pd}(s)$. These cost functions are $Sfd(s)$ and $Tfd(s)$, $Scd(s)$ and $Tcd(s)$, $Spd(s)$ and $Tpd(s)$. To make explicit the relationship between the cost functions of endmarker tries and doubly–chained prefix tries, and following Knuth [Knu73–6.3–Ex 24], it is valuable to break up the depth $depth(d)$ of a data node d into two components: $ldepth(d)$ and $rdepth(d)$. The left–link depth $ldepth(d)$ is the number of edges ending at left sons on the path that connects the root and d; the right–link depth $rdepth(d)$ is defined analogously, but counting path edges that end at right sons. The left–link and right–link total data node path lengths of a binary tree with data nodes g are defined by

$$ltpl(g) \equiv \sum_{all\ data\ nodes\ d\ of\ g} ldepth(d),$$
$$rtpl(g) \equiv \sum_{all\ data\ nodes\ d\ of\ g} rdepth(d).$$

For the doubly–chained prefix tries $t^{fd}(s)$, $t^{cd}(s)$, and $t^{pd}(s)$, the respective left–link total data node path lengths will be respectively denoted by $Lfd(s)$, $Lcd(s)$, and $Lpd(s)$; the right–link total data node path lengths by $Rfd(s)$, $Rcd(s)$, and $Rpd(s)$. The total data node path lengths are then given by $Tfd(s) = Lfd(s) + Rfd(s)$, $Tcd(s) = Lcd(s) + Rcd(s)$, and $Tpd(s) = Lpd(s) + Rpd(s)$.

DEFINITION 4: The *full doubly–chained prefix trie* built from a finite set of $s \subset \mathcal{A}^*$, denoted by $t^{fd}(s)$, is the labeled binary tree with data nodes recursively defined as follows.

(*i*) If $s = \emptyset$, $t^{fd}(s)$ is equal to the empty tree Λ.

(*ii*) If $s = \{\varepsilon\}$, $t^{fd}(s)$ is equal to the 'data' node $\mathcal{D}$.

(*iii*) Otherwise, let a be the highest ranked character which begins a string of s, i.e. $a \equiv max(first(s))$ where $first(s) \equiv \{c \in \mathcal{A} \mid s_c \neq \emptyset\}$. Then, $t^{fd}(s)$ is the labeled binary tree having an 'internal' root node with label a, left subtree $t^{fd}(s_a)$ and right subtree $t^{fd}(s - as_a)$.

The counting root functions of the compact endmarker tries cost functions, *Sfd*, *Lfd* and *Rfd*, can be deduced from Definition 4 by the following observation. Let the triple (t, a, t') denote the labeled binary tree having a root node with label a, left subtree t, and right subtree t'. If $first(s) \equiv \{c \in \mathcal{A}_m \mid s_c \neq \emptyset\} = \{c_1, \ldots c_r\}$ with $c_1 < \ldots < c_r$, then

$$t^{fd}(s) = \Big(t^{fd}(s_{c_r}), c_r, \Big(t^{fd}(s_{c_{r-1}}), c_{r-1}, \ldots \Big(t^{fd}(s_{c_1}), c_1, t^{fd}(s \cap \{\varepsilon\})\Big) \ldots\Big)\Big).$$

This relation, together with Definition 4, yields recursive expressions for $Sfd(s)$, $Lfd(s)$, and $Rfd(s)$. The implied counting root functions and their corresponding values of $\mathcal{N}_{hn}$ are recorded in Table IV. The expectations $E_{hn}[Lfd]$, $E_{hn}[Lfd]$, and $E_{hn}[Rfd]$ computed by means of Theorem 8 appear in Table V, which is given at the end of this subsection.

A recursive definition of the compact doubly–chained prefix trie $t^{cd}(s)$ can be obtained from Definition 4 by considering finite subset $s \subset \mathcal{A}^{\circledast}$, and replacing (*ii*) with the following compaction condition:

(*ii**) If $s = \{x\}$, $t^{cd}(s)$ is equal to a single 'data' node $\mathcal{D}$ with a label equal to x.

The counting root functions of *Sce*, *Lcd* and *Rcd*, and their corresponding values of $\mathcal{N}_{hn}$ are recorded in Table IV. The expectations $E_{hn}[Sce]$, $E_{hn}[Lcd]$ and $E_{hn}[Rcd]$, deduced with the help of Theorem 8, appear in the Table V.

Remark. Further observations can be made [delaT87a] on the relationship among the cost functions of tries analyzed so far. If $s \subset \mathcal{A}^*$,

Table IV: Doubly–chained prefix tries

X	$\rho_X(n_0,\ldots,n_m)$	$\mathcal{N}_{hn}[\rho_X]$
Sfd	$\sum_{1\le i\le m}(1-\delta_{0,n_i})$	$m\left[\binom{m^{[h]}}{n}-\binom{m^h}{n}\right]$
Scd	$(1-\delta_{1,n})\sum_{1\le i\le m}(1-\delta_{0,n_i})$	$m\left[\binom{m^{[h]}}{n}-\binom{m^h}{n}\right](1-\delta_{1,n})$
$\widetilde{Spd}$	$(1-\delta_{n,0})\sum_{1\le i\le m}(1-\delta_{0,n_i}-\delta_{n,n_i})$	$m\left[\binom{m^{[h]}}{n}-\binom{m^h}{n}-\binom{m^{[h-1]}}{n}\right](1-\delta_{n,0})$
Lfd	$n-n_0$	$(m^{[h]}-1)\binom{m^{[h]}-1}{n-1}$
Lcd	$(n-n_0)(1-\delta_{1,n})$	$(m^{[h]}-1)\binom{m^{[h]}-1}{n-1}(1-\delta_{1,n})$
$\widetilde{Lpd}$	$(n-n_0)\left[1-\sum_{1\le i\le m}\delta_{n,n_i}\right]$	$(m^{[h]}-1)\left[\binom{m^{[h]}-1}{n-1}-\binom{m^{[h-1]}-1}{n-1}\right]$
$Rfd=Rcd$ $=Rp$	$\sum_{0\le j<i\le m}n_j(1-\delta_{0,n_i})$	$\left[\frac{m-1}{2}(m^{[h]}-1)+m\right]\left[\binom{m^{[h]}-1}{n-1}-\binom{m^h-1}{n-1}\right]$

where $n=\sum_{0\le j\le m}n_j$. Note that $X(\{\varepsilon\})=X(\emptyset)=0$.

$Rfd(s)=Rcd(s)$, $Sfd(s)-Scd(s)=Lfd(s)-Lcd(s)$ and this quantity is equal to the number of internal nodes of $t^{fd}(s)$ that are left sons, have an empty right subtree and a left subtree that contains only one data node. As for the cost functions of endmarker tries and doubly–chained prefix tries, we

have $Scd(s) - Sce(s) = |s| - P(s) - 1 + \delta_{|s|,0}$, where $P(s)$ is the number of keys in s which are prefixes of other keys in s (compare §4.5), and $Tce(s) = Lcd(s) + P(s)$. If $s \subset \mathcal{A}^*$, we further have $Sfd(s) - Sfe(s) = Scd(s) - Sce(s)$ and $Tfe(s) = Lfd(s) + P(s)$.

DEFINITION 5: The *patrician doubly-chained prefix trie* built from a finite set of keys $s \subset \mathcal{A}_m^{\circledast}$ is the labeled binary tree with data nodes, denoted by $t^{pd}(s)$, which is recursively defined as follows:

(*i*) If $s = \emptyset$, $t^{pd}(s)$ is the empty tree Λ.

(*ii*) If $s = \{\varepsilon\}$, $t^{pd}(s)$ is equal to the 'data' node $\mathcal{D}$.

(*iii*) Otherwise, let a be the highest ranked character that begins a key of s $\big($i.e., $a \equiv max(first(s))\big)$, and let x be the longest prefix shared by all the keys of s that begin with a. Then, $t^{pd}(s)$ is the binary tree with data nodes having an 'internal' root node with label a, left subtree $t^{pd}(s_x)$, and right subtree $t^{pd}(s - as_a)$ $\big($note that $t^{pd}(s - as_a) = t^{pd}(s - xs_x)\big)$.

The relation $Rpd(s) = Rcd(s)$ follows from Definitions 4 and 5. Although *Sfp* and *Lfp* are not prefix-cardinality dependent functions [delaT87a], we can write $Spd(s) = \widetilde{Spd}(s) + F(s)$ and $Tpd(s) = \widetilde{Tpd}(s) + |s|F(s)$, where $F(s) = 0$ unless all the keys in s share a common prefix of positive length in which case $F(s) = 1$.

The counting root functions of $\widetilde{Spd}(s)$ and $\widetilde{Tpd}(s)$, which can be deduced with the aid of Definition 5, appear in Table IV given above. The expectations of $\widetilde{Spd}$ and $\widetilde{Lpd}$ can then be computed with the help of Theorem 8. Furthermore, since $F(s) = 1$ precisely when $\emptyset \neq s \subseteq c\mathcal{A}^{[h-1]}$ for some $c \in \mathcal{A}$, we have

$$E_{hn}[F] = m\left[\frac{\binom{m^{[h]}-m^h}{n}}{\binom{m^{[h]}}{n}} - \delta_{n,0}\right].$$

We can now compute $E_{hn}[Spd] = E_{hn}[\widetilde{Spd}] + E_{hn}[M]$ and also $E_{hn}[Lpd] = E_{hn}[\widetilde{Lpd}] + nE_{hn}[M]$. The outcome of these calculations is recorded in Table V.

Remark. It is also possible to show [delaT87a] that $Spd(s) \leq 2|s| - P(s) + F(s) - 2 \leq 2|s| - 1$, when $s \neq \emptyset$. About the relationships between the costs functions of compact and patrician doubly–chained prefix tries, we know that $Rcd(s) = Rpd(s)$, and also that $Scd(s) - Spd(s) + F(s)$ equals the number of internal nodes of $t^{cd}(s)$ which are left sons and have empty right subtrees.

Table V: doubly–chained prefix tries

X	$E_{hn}[X]$
Tl	$n\Big[h - \frac{1}{m-1}\Big(1 - \frac{h+1}{m^{[h]}}\Big)\Big]$
P	$\frac{n}{m}\big[1 - \frac{1}{m^{[h]}}\big] - \sigma[0,1]$
F	$m\Big[\tau(m^{[h]}, m^h, n, 0) - \delta_{n,0}\Big]$
Sfd	$n\frac{m^h}{m^{[h]}} - 1 + \delta_{0,n} + m^{[h-1]} - \sigma[0,0]$
Scd	$E_{hn}[Sfd] - \frac{m}{m-1}\big(\kappa[0,1] - \sigma[0,1]\big)$
Spd	$n\frac{m^h}{m^{[h]}} + m^{[h-1]} - 1 + \delta_{0,n} - m\,\theta[0,0] + (m-1)\,\sigma[0,0] + E_{hn}[F]$
Lfd	$E_{hn}[Tl]$
Lcd	$E_{hn}[Lfd] - \frac{m}{m-1}\Big(\kappa[0,1] - \sigma[0,1]\Big)$
Lpd	$E_{hn}[Tl] + \frac{m}{m-1}\Big(\theta[1,1] - \eta[1,1]\Big) + n\,E_{hn}[F]$
$Rfd = Rcd = Rpd$	$\frac{m-1}{2}E_{hn}[Tl] + n\Big(1 - \frac{1}{m^{[h]}}\Big) - \frac{m}{2}\kappa[1,1] - \frac{1}{2}\sigma[1,1]$

where the basic trie sums $\sigma[a,b]$, $\kappa[a,b]$, $\theta[a,b]$ and $\eta[a,b]$ are as defined in (10), (11), (12), and (13).

6. Concluding remarks

We have computed exact average space and time complexities of the retrieval algorithms for several trie varieties that store unrestricted sets of keys. To this end we developed a unified approach, the root function method, which provides a general framework for computing expectations of random variables belonging to a certain wide class with respect to the prefix model. This method suggests itself as a potentially helpful tool when confronted with other trie related problems such as the analysis of yet another kind of trie with respect to the prefix model.

Further trie variants of interest are furnished, for instance, by the *bucket size* b (b an integer greater than 1) version of endmarker and doubly–chained prefix tries. For a finite set of keys $s \subset \mathcal{A}^*$, the *endmarker trie of bucket size* b is obtained from the compact endmarker trie by stopping the branching as soon as we reach a node of $t^{ce}(s)$ that has b or fewer data nodes among its descendants. The subtree rooted at such a node, v say, is collapsed into a data node which takes as its label the set s_x, where x is the string corresponding to the path that connects the root to v. The resulting tree is denoted by $t^{be}(s)$. It is possible to give a recursive definition of $t^{be}(s)$ which is analogous to the definition given earlier (compare §5.1) for the compact endmarker trie, wherein condition $(ii*)$ must be replaced with

(iib) If $|s| \leq b$, $t^{be}(s)$ is equal to a single 'data' node $\mathcal{D}$ with a label equal to the set s.

Let $Sbe(s)$ and $Tbe(s)$ be the number of internal nodes and the total data node path length of $t^{be}(s)$, and let $sbe(s)$ and $tbe(s)$ be their respective root functions. Then,

$$sbe(s) = \big(1 - \sum_{0 \leq j \leq b} \delta_{|s|,j}\big) sce(s),$$
$$tbe(s) = \big(1 - \sum_{0 \leq j \leq b} \delta_{|s|,j}\big) tce(s),$$

where *sce* and *tce* denote the root functions of the compact endmarker trie cost functions *Sce* and *Tce* (compare Table II of §5.1). The normalized expectations of *sbe* and *tbe* follow immediately from the corresponding expectations of *sce* and *tce* (which are given in Table II of §5.1). The desired expectations of *Sbe* and *Tbe* can then be deduced by applying the general formula of Theorem 8 once again. In a similar manner, it is possible to define the bucket versions of patrician endmarker tries and of doubly–chained prefix tries, and to compute the exact average time and space required by their respective retrieval algorithms.

Acknowledgement. I would like to thank my thesis advisor Gary Knott for proposing this problem.

REFERENCES

AHU83 A. V. Aho, J. E. Hopcroft, and J. Ullman, *The Design and Analysis of Computer Algorithms*, Addison–Wesley, Reading, Mass. 1983.

delaB59 R. de la Brandais, "File Searching Using Variable Length Keys," *Proc. Western Joint Computer Conference* 15, pp. 295–298, 1959.

delaT87a P. de la Torre, "Analysis of Tries," Report CS–TR–1890, Ph. D. Thesis, Department of Computer Science, University of Maryland, 1987.

delaT87b P. de la Torre, "Extending the Flajolet–Regnier–Sotteau Method of Trie Analysis to the Prefix Model," Systems Research Center, University of Maryland, 1987.

Dev82 L. Devroye, "A Note on the Average Depth of Tries," *Computing*, vol. 28, no. 4, pp. 367–371, 1982.

Fla83 Ph. Flajolet, "Methods in the Analysis of Algorithms: Evaluation of a Recursive Partitioning Process," *Proceedings of the 1983 International FCT–Conference*, Borgholm, Sweden, pp. 141–158 in *Lecture Notes in Computer Science 158*, ed. M. Karpinski, 1983.

FRS85 Ph. Flajolet, M. Regnier, and D. Sotteau, "Algebraic Methods for Trie Statistics ," *Annals of Discrete Mathematics*, vol. 25, pp. 145–188, 1985.

FS86 Ph. Flajolet, and R. Sedgewick, "Digital Search Trees Revisited," *SIAM J. Comput.*, vol. 15, no. 3, pp. 748–767, 1986.

Fra77 J. Françon, "On the Analysis of Algorithms for Trees," *Theor. Comp. Sci.*, vol. 4, pp. 155–169, 1977.

Fre60 E. Fredkin, "Trie Memory," *CACM*, vol 3, no. 9, 490–499, 1960.

Gon84 G. H. Gonnet, *Handbook of Algorithms and Data Structures*, Addison–Wesley, Reading, Mass. 1984.

GG78 C. C. Gotlieb, and L. R. Gotlieb, *Data Types and Structures*, Prentice Hall, Englewood Cliffs, N. J. 1978.

Gwe68 G. Gwegenberger, "Anwendung einer binaren Verweiskettenmetothe beim Aufbau von Listen," *Elektronische Rechenanlagen*, vol. 10, pp. 223–226, 1968.

HS76 E. Horowitz, and S. Sahni, *Fundamental of Data Structures*, Computer Science Press, Potomac, Md. 1976.

Kno86 G. D. Knott, "Including Prefixes in Doubly–Chained Tries," Report CAR–TR–236, Computer Science Department, University of Maryland, 1986.

Knu73b D. E. Knuth, *The Art of Computer Programming*, Volume 3: *Sorting and Searching*; Addison–Wesley, Reading, Mass. 1973.

Mel84 K. Melhorn, *Data Structures and Algorithms 1: Sorting and Searching*; Springer–Verlag, Berlin, 1984.

Mor68 D. R. Morrison, "PATRICIA – Practical Algorithm To Retrieve Information Coded In Alphanumeric," *JACM*, vol. 15, pp. 514–534, 1968.

Nie81 J. Nievergelt, "Trees as Data and File Structure," *Proceedings CAAP 81*, pp. 35–45 in *Lecture Notes in Computer Science 112*, 1981.

Pla84 D. Plateau, "A Pruned Trie to Index a Sorted File and its Evaluation," *Inf. Systems*, vol. 9, no. 2, pp. 157–165, 1984.

Pla83 D. Plateau, "Une Structure Compacte pour Indexer un Fichier Totalment Ordonné: évaluation et mise en ouvre," Thesis, Univ. Paris XI Orsay, 1983.

Reg81 M. Regnier, "On the Average Height of Trees in Digital Search and Dynamic Hashing," *Inf. Proc. Letters*, vol. 13, no. 2, pp. 64–66, 1981.

RND77 E. M. Reingold, J. Nievergeld, and N. Deo, *Combinatorial Algorithms: Theory and Use*, Prentice–Hall, 1977.

Riv74 R. L. Rivest, "Analysis of Associative Retrieval Algorithms," Report STAN–CS–74–415, Ph.D. Thesis, Computer Science Dept., Stanford University, 1974.

Sed83 R. Sedgewick, *Algorithms*, Addison–Wesley, Reading, Mass. 1983.

Sta80 T. A. Standish, *Data Structure Techniques*, Addison–Wesley, Reading, Mass. 1980.

Sus63 E. H. Sussenguth, "Use of Tree Structures for Processing Files," *CACM*, vol. 6, pp. 272–279, May 1963.

Tra78 L. I. Trabb Pardo, "Set Representation and Set Intersection," Report STAN–CS–78–681, Ph. D. Thesis, Department of Computer Science, Stanford University, 1978.