On the number of prefix and border tables

Julien Clément[1] and Laura Giambruno[1]

GREYC, CNRS-UMR 6072, Université de Caen, 14032 Caen, France
julien.clement@unicaen.fr, laura.giambruno@unicaen.fr

Abstract. For some text algorithms, the real measure for the complexity analysis is not the string itself but its structure stored in its prefix table (or border table, as border and prefix tables can be proved to be equivalent). We give a new upper bound on the number of prefix tables for strings of length n (on any alphabet) which is of order $(1+\varphi)^n$ (with $\varphi = \frac{1+\sqrt{5}}{2}$ the golden mean) and present also a lower bound.

1 Introduction

The prefix table of a string w reports for each position i the length of the longest substring of w that begins at i and matches a prefix of w. This table stores the same information as the border table of the string, which memorises for each position the maximal length of prefixes of the string w ending at that position. Indeed two strings have the same border table if and only if they have the same prefix table.

Both tables are useful in several algorithms on strings. They are used to design efficient string-matching algorithms and are essential for this type of applications (see for example [8] or [3]). It has been noted that for some text algorithms (like the Knuth-Morris-Pratt pattern matching algorithm), the string itself is not considered but rather its structure meaning that two strings with the same prefix or border table are treated in the same manner. For instance, strings `abbbbb`, `baaaaa` and `abcdef` are the same in this aspect.

The study of these tables has become topical. In fact several recent articles in literature (cf. [7, 4, 2, 5]) focus on the problem of validating prefix and border tables, that is the problem of checking if an integer array is either the prefix or the border table of at least one string. In a previous paper [10] Moore et al. represented distinct border tables by canonic strings and gave results on generation and enumeration of these string for bounded and unbounded alphabets. Some of these results were reformulated in [5] using automata-theoretic methods. Note that different words on a binary alphabet have distinct prefix/border tables. This gives us a trivial lower bound in 2^{n-1} (since exchanging the two letters of the alphabet does not change tables). This is no longer true as soon as the alphabet has cardinality strictly greater than 2: for instance, words `abb` and `abc` admit the same prefix table $[3, 0, 0]$.

In this paper we are interested in giving better estimations on the number of prefix/border tables p_n of words of a given length n, that those known in literature.

For this purpose, we define the combinatorial class of *p-lists*, where a p-list $L = [\ell_1, \ldots, \ell_k]$ is a finite sequence of non negative integers.

n	$p_{n,1}$	$p_{n,2}$	$p_{n,3}$	$p_{n,4}$	$p_{n,5}$	p_n
1	1					1
2	1	1				2
3	1	3				4
4	1	7	1			9
5	1	15	4			20
6	1	31	15			47
7	1	63	46			110
8	1	127	134	1		263
9	1	255	370	4		630
10	1	511	997	16		1525
11	1	1023	2625	52		3701
12	1	2047	6824	162		9034
13	1	4095	17544	500		22140
14	1	8191	44801	1467		54460
15	1	16383	113775	4180		134339
16	1	32767	287928	11742	1	332439
17	1	65535	726729	32466	4	824735
18	1	131071	1831335	88884	16	2051307
19	1	262144	4610078	241023	52	5113298

Table 1: First values: p_n is the total number of prefix tables for strings of size n, $p_{n,k}$ is the number of prefix tables for strings of size n with an alphabet of size k which cannot be obtained using a smaller alphabet.

We constructively define an injection ψ from the set of prefix tables to the set of p-lists which are easier to count. In particular we furnish an algorithm associating to a prefix table a p-list. We define *prefix lists* as p-lists that are images of prefix tables under ψ. We moreover describe an "inverse" algorithm that associates to a prefix list $L = \psi(P)$ a word whose prefix table is P. This result confirm the idea that prefix lists represent a more concise representation for prefix tables.

We then deduce a *new upper bound and a new lower bound on the number p_n of prefix tables* (see Table 1 for first numerical values) for strings of length n or, equivalently, on the number of border tables of length n.

Let $\varphi = \frac{1}{2}(1+\sqrt{5}) \approx 1.618$, the golden mean, we have:

Proposition 1 (Upper bound). *The number of valid prefix tables p_n can be asymptotically upper bounded by the quantity $\frac{1}{2}\left(1+\frac{\sqrt{5}}{5}\right)\ (1+\varphi)^n + o(1)$.*

Proposition 2 (Lower bound). *For any $\varepsilon > 0$ there exists a family of prefix tables $(\mathcal{L}_n)_{n\geq 0}$ such that* $\mathrm{Card}(\mathcal{L}_n) = \Omega((1+\varphi-\varepsilon)^n)$.

2 Preliminaries

2.1 Notations and definitions

Let A be an ordered alphabet. A word w of length $|w| = n$ is a finite sequence $w[0]w[1]\dots w[n-1] = w[0\,..\,n-1]$ of letters of A. The language of all words is A^*, and A^+ is the set of nonempty words. The prefix (resp. suffix) of length ℓ, $0 \leq \ell \leq n$, of w is[1] the word $u = w[0\,..\,\ell-1]$ (resp. $u = w[n-\ell\,..\,n-1]$). A border u of w is a word that is both a prefix and a suffix of w and distinct from w itself. We define $\mathrm{bord}(w)$ as the set of all proper borders of w.

[1] With the convention that $w[0\,..\,-1] = w[n\,..\,n-1] = \varepsilon$ is the empty word whenever $\ell = 0$.

Definition 1 (Prefix table). *The prefix table* Pref_w *of a word* $w \in A^+$ *of length* n*, is the table of size* n *defined, for* $0 \leq i < n$*, by*

$$\mathrm{Pref}_w[i] = \mathrm{lcp}(w, w[i\,..\,n-1]),$$

where lcp *denotes the maximal length of common prefixes of the two words.*

Another well-known structure used to represent the correlation structure of a string is the border table of a word.

Definition 2 (Border table). *The border table* Border_w *of a word* $w \in A^+$ *of length* n*, is the table of size* n *defined, for* $0 \leq i < n$*, by*

$$\mathrm{Border}_w[i] = \max\{|u| \mid u \text{ is a border of } w[0\,..\,i]\},$$

Example 1. Let w be the word `abaababa`. We have the following representations for the prefix and border tables of w.

i	0	1	2	3	4	5	6	7
$w[i]$	a	b	a	a	b	a	b	a
$\mathrm{Pref}_w[i]$	8	0	1	3	0	3	0	1
$\mathrm{Border}_w[i]$	0	0	1	1	2	3	2	3

These structures (border and prefix tables) are in fact equivalent; actually the following proposition states a fact discussed in [3] and recently deepened in [1], where linear time conversion algorithms are given. In the following we furnish a proof of this equivalence: elements of the proof will be used in the next section.

Proposition 3. *Two strings have the same border table if and only if they have the same prefix table.*

Proof (sketch). Let w be a word in A^+ of length $|w| = n > 0$. We can relate the border table Border_w to the prefix table Pref_w.
For a position j in w of length n, let

$$I(j) = \{i \mid 0 < i \leq j \text{ and } i + \mathrm{Pref}_w[i] - 1 \geq j\}.$$

The elements in $I(j)$ represent the positions $i \leq j$ for which the longest common prefixes between w and $w[i\,..\,n-1]$ overlap position j in w. Then we have

$$\mathrm{Border}_w[j] = 0 \text{ if } I(j) = \emptyset, \text{ and } \mathrm{Border}_w[j] = j - \min I(j) + 1 \text{ otherwise.} \quad (1)$$

Conversely, given the border table Border_w of w, we define the prefix table Pref_w in the following way. First we set $\mathrm{Pref}_w[0] = \|w\|$. Then let $j > 0$ be a position in w and let $I'(j) = \{i \mid j \leq i < \|w\| \text{ and } w[j\,..\,i] \in \mathrm{bord}(w[0\,..\,i])\}$. We have

$$\mathrm{Pref}_w[j] = 0 \text{ if } I'(j) = \emptyset, \text{ and } \mathrm{Pref}_w[j] = \max(I'(j)) - j + 1 \text{ otherwise.} \quad (2)$$

With (1) and (2), one proves that two words have the same border table if and only if they have the same prefix table. □

Recent literature focuses on the problem of validating prefix and border tables and, in case of a valid table, providing the word associated to it that is the smallest in the lexicographic order (cf. [7, 2]).

2.2 Previous work

Previous work in [10] focused on counting distinct strings of length n with respect to their prefix/border tables: an upper bound is given in the form

$$b_n = \sum_{k=1}^{k^*} \left\{ {n-2^{k-1}+k \atop k} \right\}, \tag{3}$$

where $\left\{ {m \atop j} \right\}$ denotes the Stirling number of second kind (the number of partitions of m in j non empty parts), and $k^* = \lceil \log_2(n+1) \rceil$. The quantity k^* is the minimal number of distinct letters to obtain all possible prefix tables of size n.

Numerically it is clear that b_n is far from being a tight approximation of the number p_n of prefix tables of size n. One can indeed prove that $b_n \gg p_n$. The following lemma can help us to formalize this fact.

Lemma 1. *For α_n such that $\alpha_n \to \infty$ and $\alpha_n = O(n^c)$ for some $c \in]0,1[$, one has*

$$\left\{ {n \atop \alpha_n} \right\} \sim \frac{(\alpha_n)^n}{\alpha_n!} \sim \sqrt{2\pi}(\alpha_n)^{n-\alpha_n+1/2} e^{\alpha_n}.$$

The proof of this lemma relies on the fact that applications from $\{1,\dots,m\}$ to $\{1,\dots,\alpha_n\}$ are, if α_n is small enough, almost always surjective.

This lemma suffices to prove that b_n in Equation (3) is at least of order is $\left\{ {cn \atop d \log n} \right\}$ (considering for instance the Stirling number for $k = k^* - 1$ in (3)), for some positive constants c and d. So that $\log b_n$ is at least of order $n \log\log n$. Hence we have:

Corollary 1. *We have $\frac{1}{n} \log b_n = \Omega(\log\log n)$.*

In Section 4 we try to improve the bound in (3), yielding the result of Proposition 1.

3 Prefix Lists

The information in a valid prefix table is somewhat redundant since we do not need to use all values in the table to build a corresponding word. We introduce prefix lists which are more concise and sufficient to be able to reconstruct such a word. We first define the combinatorial class of p-lists as it follows:

Definition 3. *We define a p-list $L = [\ell_1, \dots, \ell_k]$ as a finite sequence of positive integers together with a* size *defined for a list as $\|L\| = \sum_{i=1}^{k} \|\ell_i\|$, where the size $\|i\|$ is i if $i > 0$ and 1 if $i = 0$.*

Let $\mathcal{P}$ denote the set of prefix tables and $\mathcal{L}$ the set of p-lists. In this section we define an injection $\psi : \mathcal{P} \longrightarrow \mathcal{L}$ in a constructive manner. We define prefix lists as:

Definition 4. *Let L be a p-list. We say that L is a* prefix list *if $L = \psi(P)$ for a prefix table $P \in \mathcal{P}$.*

3.1 Algorithms

From prefix tables to prefix lists. We define constructively an injection ψ from $\mathcal{P}$ to $\mathcal{L}$ by defining an algorithm in a "right-to-left manner".

Procedure `PrefixToList`$(P = P[0\,..\,n-1])$

```
L ← []
i ← n − 1
while i > 0 do
    I ← {j | 0 < j ≤ i and j + P[j] − 1 ≥ i}
    if I = ∅ then
        (ℓ, i) ← (0, i − 1)
    else
        (ℓ, i) ← (i − min(I) + 1, min(I) − 1)
    L ← [ℓ] · L
return L
```

Intuitively, it scans the prefix table from right to left, starts with the last position $i = n - 1$ and gets from the prefix table the length ℓ of the leftmost longest common (proper) prefix which overlaps the current position i, or sets $\ell = 0$ if there is no such prefix. This length is inserted at the beginning of the list and the position i is updated to the position immediately before the prefix (if it exists) or just one position before (if it is not the case). The algorithm stops when the first position $i = 0$ is attained.

For each position i in P, the elements in I represent, as in the proof of Proposition 3, the positions less or equal to i, such that the longest common prefix with w starting at these positions overlap position i.

Definition 5. *For a given prefix table P in $\mathcal{P}$, we define $\psi(P)$ as the prefix list obtained by executing the algorithm PrefixToList on P.*

Example 2. Let w be the word `abaababa`. We have the following representation for the prefix table of w.

i	0	1	2	3	4	5	6	7
$w[i]$	a	b	a	a	b	a	b	a
$\mathrm{Pref}_w[i]$	8	0	1	3	0	3	0	1

For this table we get the associated list $L = [0, 1, 2, 3]$. In fact, executing the algorithm `PrefixToList` to Pref_w, we start with $i = 7$ and we get that the set of starting indexes of prefixes overlapping i is $I = \{5, 7\}$. Thus $\ell = i - \min(I) + 1 = 3$, the length of the overlapping prefix until i, is appended to $L = [\,]$. Now i is initialised to 4 the position before $\min(I) = 5$. Next we have $I = \{3\}$ and so $\ell = 2$ is prefixed to $L = [3]$ and $i := 2$. Again $I = \{2\}$, $\ell = 1$ and $i := 1$. Now $I = \emptyset$, thus $\ell = 0$, $i := 0$ and the algorithm stops.

Remark 1. At first view, it would be more intuitive to define prefix lists with an algorithm visiting the prefix table from left to right. For instance let P be a prefix table and $L = [\,]$ an empty list. A greedy algorithm for this construction starts with the second position $i = 1$ and appends at the beginning of L, the length $\ell = P[i]$ of the longest common prefix starting there. Then i is updated to $i + \ell$, if $\ell > 0$ and to $i + 1$ if $\ell = 0$. Again the algorithm appends $\ell = P[i]$ and so on until position $n - 1$ is attained. However this construction of "prefix list" from left to right fails to define an injection from

prefix tables to prefix lists (which is our goal for finding an upper bound). In fact let $P = [8, 0, 1, 3, 0, 3, 0, 1]$ be a valid prefix table, as in Example 2, and $P' = [8, 0, 1, 3, 0, 1, 0, 1]$ be a valid prefix table associate to w', then one has

i	0	1	2	3	4	5	6	7
$w'[i]$	a	b	a	a	b	a	c	a
$\mathrm{Pref}_{w'}[i]$	8	0	1	3	0	1	0	1

Since to P and P' is associated the same list $L = [0, 1, 3, 0, 1]$ then the correspondence between prefix tables and these lists cannot be injective.

From border tables to prefix lists. In order to prove the injection we will define the function ψ in term of border tables: we define another function ψ' from the set of border tables to the set of prefix lists defined by the following algorithm:

Procedure `BorderToList`$(B = B[0\,..\,n-1])$

```
L ← []
i ← n − 1
while i > 0 do
    ℓ ← B[i]
    if B[i] = 0 then
        i ← i − 1
    else
        i ← i − B[i]
    L ← [ℓ] · L
Return L
```

From the algorithm follows the definition of ψ' in an inductive way:

Definition 6. *For a border table B of length n, let $\ell = B[n-1]$. We define $\psi'(B)$ as:*

$$\psi'(B) = \begin{cases} \psi'(B[0\,..\,n-1-\ell]) \cdot [\ell], & \text{if } \ell > 0; \\ \psi'(B[0\,..\,n-2]) \cdot [\ell], & \text{if } \ell = 0. \end{cases}$$

The functions ψ and ψ' applied on equivalent border and prefix tables give rise to the same prefix lists:

Proposition 4. *Let B be a border table of a word w and P be the prefix table of w. Then we have that $\psi(P) = \psi'(B)$.*

Proof. For a given position i in w, let $I = \{j \mid 0 \leq j \leq i \text{ et } j + P[j] - 1 \geq i\}$ as defined in the proof of Proposition 3 and in the algorithm `PrefixToList` for the computation of $\psi(P)$. By the conversion rules from prefix table to border table (see proof of Proposition 3), we have that

$$B[i] = \begin{cases} 0, & \text{if } I = \emptyset; \\ i - \min(I) + 1, & \text{if } I \neq \emptyset. \end{cases}$$

Thus if $I = \emptyset$ then the value $\ell = 0 = B[i]$ is inserted at the beginning of L for both algorithms `PrefixToList` and `BorderToList`. If $I \neq \emptyset$ then $\ell = i - \min(I) + 1 = B[i]$ is inserted at the beginning of the list L for both algorithms. Then i is decremented in the same way for both the algorithms.

□

Thus, for a given border table B, there exist $0 \leq i_1 \leq \cdots \leq i_r = n-1$, such that $\psi'(B) = L = (B[i_1], \ldots, B[i_r])$ and $i_j = i_{j+1} - B[i_{j+1}]$.

Example 3. Let w be the word `abaababa`. The following table shows its Border table Border_w for all values of i.

i	0	1	2	3	4	5	6	7
$w[i]$	a	b	a	a	b	a	b	a
$\mathrm{Border}_w[i]$	0	0	1	1	2	3	2	3

The associated prefix list is $L = [0, 1, 2, 3] = [B[1], B[2], B[5], B[7]]$.

From p-lists to words. We now describe an "inverse" algorithm that associates to a prefix list $L = \psi(P)$ a word w whose prefix table is P. Let $L = [\ell_1, \ldots, \ell_m]$ and $n = \|L\|$, for the length $\|\cdot\|$ defined for p-lists, then the string $w[0\,..\,n]$ is computed in the following way.

Procedure `ListToWord`$(L = [\ell_1, \ldots, \ell_m])$

```
w[0] ← new letter
pos ← 1
for i ← 1 to m do
    if ℓ_i > 0 then
        for j ← 0 to ℓ_i − 1 do
            w[pos + j] ← w[j]
        pos ← pos + ℓ_i
    else
        w[pos] ← new letter
        pos ← pos + 1
n ← pos
return w[0..n]
```

The `new letter` function returns a new letter not used so far. Informally the algorithm proceeds from left to right on the p-list input $[\ell_1, \ldots, \ell_m]$. It starts with a word reduced to one letter. Then iteratively for $i \in [1\,..\,m]$, if $\ell_i > 0$ the algorithm copies ℓ_i symbols, from the previous constructed word u, at the end of u, otherwise the algorithm introduces a new symbol in w. Note that overlapping is allowed since we are building the word from left to right.

Example 4. Let $w =$ `abaababa` as in Example 2, whose associated prefix list is $L = \psi(\mathrm{Pref}_w) = [0, 1, 2, 3]$. Choosing arbitrarily the first letter to be a, one can build $w =$ `a` $\cdot$ `b` $\cdot$ `a` $\cdot$ `ab` $\cdot$ `aba`. A value 0 in the prefix list implies we can choose a new letter (here `b` at the second position).

One key property is that the word w obtained by this algorithm performed on a prefix list $\psi(P)$ for a prefix table P is such that $\mathrm{Pref}_w = P$. This means that prefix lists and prefix tables are somehow equivalent and represent the same information.

Proposition 5. *Given the prefix list $L = \psi(P)$ associated with a prefix table P the word w build by the algorithm* `ListToWord` *is such that* $\mathrm{Pref}_w = P$.

Proof. We prove the proposition on border tables: let $L = \psi(P) = \psi'(B)$. We prove the result by induction on $\|L\|$. If $\|L\| = 0$, that is $L = []$, then $w = a$ and $\text{Border}_w = [0] = B$.

Let $\|L\| > 0$ and $L = L' \cdot [\ell]$. We denote by w and w' the words built by the algorithm `ListToWord` on input L and L' respectively. By construction $w = w' \cdot v$, where v consists necessarily of the first ℓ symbols (considered eventual overlapping) of w'. By the inductive definition of prefix lists there exists a decomposition $B = B' \cdot B''$ such that $L = \psi(B) = \psi(B') \cdot [\ell]$, $B''[\ell - 1] = \ell$ and the length of B'' is equal to ℓ. Thus $L' = \psi(B')$ and, by the inductive hypothesis, $\text{Border}_{w'} = B'$.

In general (see [5], [10]), given a border table $B = H \cdot T$, every word with border table H is prolonging to a word with border table B. In our case $B = B' \cdot B''$ and since $B[n-1] = \ell$, the word u prolonging w' consists necessarily of the first ℓ symbols (considering possible self overlap) of w' and is equal necessarily to v. Thus $\text{Border}_w = B$. □

3.2 Injectivity

Proposition 6. *The application ψ is injective.*

Proof. Let us consider two prefix tables $P \neq P'$ and suppose that $\psi(P) = \psi(P') = L$. By Proposition 5 the algorithm performed on L gives a word w such that $\text{Pref}_w = P = P'$. Hence we must have $\psi(P) \neq \psi(P')$.

Let us remark that the application ψ is not surjective. To a list $[0, 2, 2]$, we can associate a word $w = \mathtt{a} \cdot \mathtt{b} \cdot \mathtt{ab} \cdot \mathtt{ab} = \mathtt{ababab}$ with prefix table $\text{Pref}_w = [6, 0, 4, 0, 2, 0]$, but we have $\psi(\text{Pref}_w) = [0, 4]$.

4 Upper bound

p-lists. We can see the set of p-lists as a combinatorial class $\mathcal{L}$ of lists of positive integers

$$\mathcal{L} = \textsc{Seq}(\{0, 1, 2, 3, \dots\}), \tag{4}$$

together with a *special size measure* which can be defined for a p-list $L = [\ell_1, \dots, \ell_k]$ as $\|L\| = \sum_{i=1}^{k} \|\ell_i\|$, where the size $\|i\|$ is i if $i > 0$ and 1 if $i = 0$. It just means that $\|L\| = \sum_{i=1}^{k} \ell_i + \text{Card}\{i \mid \ell_i = 0\}$. The Seq operator applied to a combinatorial class $\mathcal{A}$ corresponds to all finite sequences of elements from $\mathcal{A}$, i.e., $\textsc{Seq}(\mathcal{A}) = \cup_{i=0}^{\infty} \mathcal{A}^i$ (reminiscent of the Kleene star operation for regular languages). By convention $\mathcal{A}^0 = \{\varepsilon\}$.

Combinatorial specifications and generating functions. In order to study a sequence $(a_n)_{n \in N}$, it is now usual [6] to consider its generating function $A(z)$, that is the formal power series defined by $A(z) = \sum_{n \geq 0} a_n z^n = \sum_{\alpha \in \mathcal{A}} z^{\|\alpha\|}$.

In our case, given the combinatorial specification of $\mathcal{L}$, it is easy [6] to compute the generating function $L(z) = \sum_{n \geq 0} \ell_n z^n$ where ℓ_n denotes the numbers of p-lists of size n. This is true when specification are unambiguous

(in the same way as unambiguity is considered in regular expressions or formal grammars).

Indeed, the general idea is the following: here we first consider a set of atoms $\mathbb{N}$. We need a size $\|\cdot\|$ compatible with the cartesian product and disjoint union, i.e., here for $i \in \mathbb{N}$ the size of atom i is $\|i\| = i$ if $i > 0$ and $\|0\| = 1$. Let us define an empty element ε (the only one with size 0). Then we have the following dictionary for translating directly from combinatorial constructions to generating functions.

Empty element:	$\varepsilon \mapsto 1$,	Cartesian product:	$\mathcal{A} \times \mathcal{B} \mapsto A(z) \times B(z)$,
Symbols:	$\alpha \in \mathbb{N} \mapsto z^{\|\alpha\|}$,	Disjoint Union:	$\mathcal{A} \cup \mathcal{B} \mapsto A(z) + B(z)$,
Sequence product:	$\textsc{Seq}(\mathcal{A}) \mapsto \frac{1}{1-A(z)}$		

Let $\varphi = \frac{1}{2}(1+\sqrt{5}) \approx 1.618$. With this dictionary and the combinatorial description (4), we get the following result for $\ell_n = [z^n]L(z)$ the number of p-lists of size n.

Proposition 7. *The number of p-lists of size n is given by*

$$\ell_n = \tfrac{1}{2}\left(1+\tfrac{\sqrt{5}}{5}\right)\ \varphi^n + \tfrac{1}{2}\left(1-\tfrac{\sqrt{5}}{5}\right)\ \varphi^{-n} = \tfrac{1}{2}\left(1+\tfrac{\sqrt{5}}{5}\right)\ \varphi^n + o(1).$$

Proof. Let $\mathcal{I} = \{0\} \cup \{1, 2, 3, \dots\}$ then by definition $\mathcal{L} = \textsc{Seq}(\mathcal{I})$. The generating function associated with $\mathcal{I}$ is $I(z) = 2z + z^2 + z^3 + \ldots = z + z\sum_{n\geq 0} z^n = z + \frac{z}{1-z}$. With this dictionary and the combinatorial description we get

$$L(z) = \frac{1}{1-\left(z+\frac{z}{1-z}\right)} = \frac{1-z}{1-3z+z^2}.$$

Since this is a rational function, using decomposition in simple elements we get

$$L(z) = \tfrac{1}{2}\left(1-\tfrac{\sqrt{5}}{5}\right)\tfrac{1}{1-z/\phi} + \tfrac{1}{2}\left(1+\tfrac{\sqrt{5}}{5}\right)\tfrac{1}{1-z/\phi'},$$

where $\phi = \frac{3+\sqrt{5}}{2}$ and $\phi' = \frac{3-\sqrt{5}}{2}$ are the two solutions of $1 - 3z + z^2$. By geometric series formula, we have that

$$\ell_n = [z^n]L(z) = \frac{1}{2}\left(1-\frac{\sqrt{5}}{5}\right)\phi^{-n} + \frac{1}{2}\left(1+\frac{\sqrt{5}}{5}\right)\phi'^{-n}.$$

Let $\varphi = \frac{1}{2}(1+\sqrt{5})$ then we have that $\phi = (1+\varphi)$ and $\overline{\phi} = 2 - \varphi = \frac{1}{(1+\varphi)}$. We thus obtain

$$\ell_n = [z^n]L(z) = \tfrac{1}{2}\left(1-\tfrac{\sqrt{5}}{5}\right)(1+\varphi)^{-n} + \tfrac{1}{2}\left(1+\tfrac{\sqrt{5}}{5}\right)(1+\varphi)^{n},$$

and the desired result follows. □

Result. The main result on the upper bound (see Proposition 1) is a reformulation of the following corollary, which is a consequence of Proposition 6.

Corollary 2. *The number p_n of prefix tables of size n is upper bounded by the number ℓ_{n-1} of p-lists of size $n-1$.*

5 Lower bound

For the lower bound, we exhibit some sets of valid prefix lists such that we are able to count them. We wish these sets to be as large as possible. In this paper, as a first step, our goal is to evaluate the exponential order growth given in Proposition 9 rather than to give a precise estimate.

The idea for proving Proposition 9 is to exhibit a language which maps bijectively to a set of prefix lists, hence maps bijectively to a set of prefix tables. Let us consider, for a fixed k, $\mathcal{L}_k = ab^k \left(ab^{<k}(\varepsilon + cb^*)\right)^*$.

Proposition 8. *Two distinct words in $\mathcal{L}_k$ have distinct prefix tables.*

Proof. We will prove the statement by proving that the words in $\mathcal{L}_k$ are in bijection with prefix lists. Then, since a prefix table is associated to a uniquely determined prefix list, the desired result immediately follows.

First we prove that prefix lists associated with words in $\mathcal{L}_k$ are concatenations of non negative integers $\ell < k$. Indeed by construction, for any word $u \in \mathcal{L}_k$ we have that the longest border of a prefix of u is of length strictly less than $k+1$. Let us note by $L(u)$ the prefix list associated with u: $L(u) = \psi(\mathrm{Pref}_u)$. Since the elements in $L(u)$ are border of prefixes of u we get the result.

Let us prove the main statement by contradiction. Let us consider u, v in $\mathcal{L}_k$ such that $u \neq v$ and $L(u) = L(v) = L = [\ell_1, \ldots, \ell_r]$ (for some $r > 0$). The prefix list L induces the same factorization in u and v. Let i be the smallest position in the words such that $u[i] \neq v[i]$ and let ℓ_j such that $\sum_{k=1}^{j-1} \|\ell_k\| < i < \sum_{k=1}^{j} \|\ell_k\|$, where $\|\cdot\|$ is defined as in Definition 3. Let $i_1 = \sum_{k=1}^{j-1} \|\ell_k\|$ and $i_2 = \sum_{k=1}^{j} \|\ell_k\|$. If $\ell_j \neq 0$ then $u[i_1+1\,..\,i_2] = ab^{\ell_j - 1} = v[i_1+1\,..\,i_2]$ since $1 \leq \ell_j < k$, that is a contradiction since $u[i] \neq v[i]$. If $\ell_j = 0$ then the longest border of $u[0\,..\,i]$ is the empty word. Then $u[i]$ can be equal either to b or to c. If $u[i] = b$ then, by definition of $\mathcal{L}_k$, $u[i]$ must be preceded by cb^t for some $t \geq 0$. The element $v[i]$, by definition, must be preceded by ab^s for some $s \geq 0$, that is a contradiction since $u[0\,..\,i-1] = v[0\,..\,i-1]$. □

Proposition 9 (Lower bound). *For any $\varepsilon > 0$ there exists a family of prefix tables $(\mathcal{L}_n)_{n\geq 0}$ such that* $\mathrm{Card}(\mathcal{L}_n) = \Omega((1+\varphi-\varepsilon)^n)$.

Proof (Sketch of the proof).

For a given k, by using analytic combinatorics for regular expressions and since the regular expression $ab^k \left(ab^{<k}(\varepsilon + cb^*)\right)^*$ is unambiguous, one can compute easily the generating function $L_k(z)$ for $\mathcal{L}_k$

$$L_k(z) = z^{k+1} \frac{1}{1 - \left(z \frac{1-z^k}{1-z}(1 + \frac{1}{1-z})\right)} = \frac{z^{k+1}(z-1)^2}{1 - 3z + z^2 + z^{k+1}}.$$

By general principles [6] we have that the number of words of length n in $\mathcal{L}_k$ is $\ell_{n,k} := [z^n]L_k(z) \sim C_k \frac{1}{\rho_k^n}$, where C_k is a constant and ρ_k is the smallest real (simple) root of $1 - 3z + z^2 + z^{k+1}$. Considering $\rho_k = \frac{1}{1+\varphi-\varepsilon_k}$ as a perturbation of the solution of $1 - 3z + z^2 = 0$, we can then evaluate ε_k by using the

bootstrapping method (as in [9]). We get that $\varepsilon_k = \frac{1}{2}(1+3\frac{\sqrt{5}}{5})\frac{1}{(1+\varphi)^k}(1+o(1))$. Hence, for any $\varepsilon > 0$, one can fix k such that $\ell_{n,k} = \Omega((\varphi + 1 - \varepsilon)^n)$ yielding the result of Proposition 9. □

This result gives only rough information on the asymptotics of $\ell_{n,k}$. A more thorough study is in order to get better estimates. However this hints at the following conjecture.

Conjecture 1. There exists a constant $c > 0$ such that the number p_n of prefix tables of size n is asymptotically equivalent to $c(1 + \varphi)^n$.

Conclusion

In this paper we have provided some bounds for the number of prefix (or border) tables. The problem of finding an asymptotic equivalent for the number of prefix tables is however still open, and would require a very fine understanding of the autocorrelation structure of words. For this purpose it would be interesting to find characterizations on prefix lists in order to get better bounds. It would be also interesting to study other families of words in bijection with prefix tables to get better lower bounds.

Acknowledgements

We would like to thank Maxime Crochemore, Cyril Nicaud and Giuseppina Rindone for helpful discussions.

References

1. W. Bland, G. Kucherov, and W. Smyth. Prefix Table Construction and Conversion. In S. Verlag, editor, *IWOCA*, Lecture Notes on Computer Science, pages 1–12, Rouen, France, May 2013.
2. J. Clement, M. Crochemore, and G. Rindone. Reverse engineering prefix tables. In S. Albers and J.-Y. Marion, editors, *26th International Symposium on Theoretical Aspects of Computer Science (STACS 2009)*, volume 3 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 289–300, Dagstuhl, Germany, 2009. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.
3. M. Crochemore, C. Hancart, and T. Lecroq. *Algorithms on strings*. Cambridge University Press, Cambridge, UK, 2007.
4. J.-P. Duval, T. Lecroq, and A. Lefebvre. Border array on bounded alphabet. *Journal of Automata, Languages and Combinatorics*, 10(1):51–60, 2005.
5. J.-P. Duval, T. Lecroq, and A. Lefebvre. Efficient validation and construction of border arrays and validation of string matching automata. *RAIRO-Theoretical Informatics and Applications*, 43(2):281–297, 2009.
6. P. Flajolet and R. Sedgewick. *Analytic Combinatorics*. Cambridge University Press, Cambridge, UK, 2009.
7. F. Franek, S. Gao, W. Lu, P. J. Ryan, W. F. Smyth, Y. Sun, and L. Yang. Verifying a border array in linear time. *Journal on Combinatorial Mathematics and Combinatorial Computing*, 42:223–236, 2002.
8. D. Gusfield. *Algorithms on strings, trees and sequences: computer science and computational biology*. Cambridge University Press, Cambridge, UK, 1997.
9. D. Knuth. The average time for carry propagation. *Indagationes Mathematicae*, 40:238–242, 1978.
10. D. Moore, W. F. Smyth, and D. Miller. Counting distinct strings. *Algorithmica*, 23(1):1–13, 1999.