Journal of Integer Sequences, Vol. 10 (2007),
Article 07.7.5

On the Number of Labeled k-arch Graphs

Saverio Caminiti and Emanuele G. Fusco
Department of Computer Science
University of Rome "La Sapienza"
Via Salaria 113
00198 Rome
Italy
caminiti@di.uniroma1.it
fusco@di.uniroma1.it

Abstract

In this paper we deal with k-arch graphs, a superclass of trees and k-trees. We give a recursive function counting the number of labeled k-arch graphs. Our result relies on a generalization of the well-known Prüfer code for labeled trees. In order to guarantee the generalized code to be a bijection, we characterize the valid code strings.

A previous attempt at counting the number of labeled k-arch graphs was made by Lamathe. We point out an error in his work, and prove it by giving a counterexample.

1 Introduction

The problem of counting labeled trees has been widely studied, and there exists a variety of proofs for the well-known Cayley formula [1], which states that the number of labeled trees of n nodes is n^{n-2} (for a survey, see [4]). Among these proofs, the one given by Prüfer [5] is based on a one-to-one correspondence between labeled trees and strings of length $n-2$ over the alphabet $\{1, \ldots, n\}$. This bijection is known as the Prüfer code.

In 1970 Rényi and Rényi [6] generalized the Prüfer bijective proof of Cayley's formula to count labeled k-trees, i.e., one of the most natural generalizations of trees.

The class of k-trees, introduced by Harary and Palmer [2], can be defined in the following recursive way:

1. A complete graph on k nodes is a k-tree.

2. If $T'_k = (V, E)$ is a k-tree, $K \subseteq V$ is a k-clique and $v \notin V$, then $T_k = (V \cup \{v\}, E \cup \{(v, x) \,|\, x \in K\})$ is also a k-tree.

A labeled k-tree is a k-tree whose nodes are assigned distinct labels. They showed the number of labeled k-trees of n nodes to be $\binom{n}{k}(k(n-k)+1)^{n-k-2}$. This result gave birth to sequences A036361, A036362, and A036506 in Sloane's On-line Encyclopedia of Integer Sequences [7].

The class of k-trees can be further generalized by relaxing the constraint in item 2 asking for the node set K to be a clique. Graphs belonging to this class, introduced by Todd [8], are known as the k-arch graphs.

A k-arch graph can be defined in the following recursive way:

1. A complete graph on k nodes is a k-arch graph.
2. If $A'_k = (V, E)$ is a k-arch graph, $K \subseteq V$ of cardinality k and $v \notin V$, then $A_k = (V \cup \{v\}, E \cup \{(v, x) \mid x \in K\})$ is also a k-arch graph.

The class of k-arch graphs can be equivalently defined as the smallest class such that:

1. A complete graph on k nodes is a k-arch graph;
2. If A'_k, obtained by removing a node of degree k from A_k, is a k-arch graph, then A_k is a k-arch graph.

A labeled k-arch graph is a k-arch graph whose nodes are assigned distinct labels. In this paper we deal with labeled k-arch graphs; we use integers in $[1, n]$ as node labels, where n always refers to $|V|$. When no confusion arises we identify a node with its label. An example of a labeled 3-arch graph on 10 nodes is given in Figure 1.

Note that when $k = 1$ both k-trees and k-arch graphs are Cayley trees.

An attempt to generalize the Prüfer bijective proof of Cayley's formula to count labeled k-arch graphs has been made by Lamathe [3]. He established a correspondence relating k-arch graphs and strings over the alphabet whose symbols are all k-subsets of the vertex set of a given k-arch graph. He claimed this correspondence to be a bijection and derived the number of labeled k-arch graphs of n nodes to be $\binom{n}{k}^{n-k-1}$. Unfortunately this result is wrong, as the majority of the strings do not represent any k-arch graph, meaning that the shown correspondence is not a bijection (see Section 3 for an example of an invalid string). Indeed, Lamathe's formula produces a serious overestimation of the number of labeled k-arch graphs. In the following table we show the overestimation ratio for certain values of n and k:

$n \backslash k$	2	3	4
10	1.6311	3.9045	5.4925
15	4.8581	85.8627	806.9044
20	18.8593	3699.9280	434531.3726

The ratio dramatically increases when n and k increase; this implies that almost all strings do not correspond to k-arch graphs. As an example, consider that when $n = 200$ and $k = 185$, the ratio is 1.6681×10^{104}.

The error made by Lamathe is to consider every possible string as the encoding of some k-arch graph. Instead the subset of strings resulting from encoding k-arch graphs needs to be correctly characterized, in the same way that Rényi and Rényi did for k-trees.

In this paper we exhibit the flaw in the Lamathe's formula by showing a simple counterexample. We provide a characterization of valid strings, and then we exploit properties of those strings in order to define a recursive function that computes the number of labeled k-arch graphs of n nodes, for any given n and k.

2 Encoding k-arch Graphs

Let $\mathcal{A}_k^n$ be the set of all k-arch graphs of n nodes, let $\binom{[1,n]}{k}$ be the set of all k-subset of the integer interval $[1,n]$, and let $\mathcal{B}_k^n$ be the set of all possible strings of length $n-k-1$ over the alphabet $\binom{[1,n]}{k}$. We use the notation $adj(v)$ to identify the set of all nodes adjacent to a given node v, and the term *k-leaf* to mean a node u such that $|adj(u)| = k$; any other node v has $|adj(v)| > k$ and is called *internal*.

Let us define the following function:

$$\rho(A_k^n) = \begin{cases} \varepsilon, & \text{if } A_k^n \text{ is a single } k+1 \text{ clique;} \\ adj(min\{v \in A_k^n : |adj(v)| = k\}) :: \rho(A_k^n \setminus \{v\}), & \text{otherwise.} \end{cases}$$

The function ρ is the injective function between $\mathcal{A}_k^n$ and $\mathcal{B}_k^n$ used by Lamathe, i.e., the generalization made by Rényi and Rényi of the Prüfer bijection applied to k-arch graphs. The recursion described by ρ embodies a pruning of the k-arch graph A_k^n that starts from the smallest k-leaf v; as v is removed from A_k^n, its adjacent set constitutes the first symbol of the string. This symbol is concatenated (by string concatenation operator ::) to the string obtained by recursively applying the function to the pruned graph. The recursion terminates when the pruning gives a clique on $k+1$ nodes, as ρ applied to a clique gives the empty string ε.

Note that, by definition of k-arch graphs, every subgraph produced during the pruning process is a k-arch graph.

It is worth remarking that we are assuming $n > k$, as well as the Prüfer code assumes the tree to have at least 2 nodes. When $n = k$, the only admissible k-arch graph is a single clique with $|\mathcal{A}_k^k| = 1$. When $n < k$, obviously $|\mathcal{A}_k^n| = 0$.

Let us show an example of the encoding process realized by the function ρ. Starting from the 3-arch graph of Figure 1, we prune it by recursively removing the smallest k-leaf. At each step the set of nodes adjacent to the removed k-leaf is added to the string.

The smallest k-leaf of the initial graph is $v_1 = 2$ and its adjacent nodes are $B_1 = \{1,6,9\}$. Then node 2 is removed from the graph, and the smallest k-leaf in the resulting graph is $v_2 = 3$, implying $B_2 = \{1,5,8\}$. Iterating this procedure we obtain $v_3 = 6$, $v_4 = 4$, $v_5 = 7$, $v_6 = 9$ and $B_3 = \{4,8,10\}$, $B_4 = \{1,5,9\}$, $B_5 = \{5,8,10\}$, $B_6 = \{1,5,8\}$ respectively. The remaining graph is a single clique of 4 nodes $\{1,5,8,10\}$. Therefore the resulting string is $(B_1, B_2, B_3, B_4, B_5, B_6) = (\{1,6,9\}, \{1,5,8\}, \{4,8,10\}, \{1,5,9\}, \{5,8,10\}, \{1,5,8\})$.

For a given k-arch graph A_k^n, we say a node $v \in V(A_k^n)$ *appears* in $\rho(A_k^n)$ if $\exists B_i \in \rho(A_k^n)$ such that $v \in B_i$.

Lemma 1. *v is an internal node in A_k^n if and only if it appears in $\rho(A_k^n)$.*

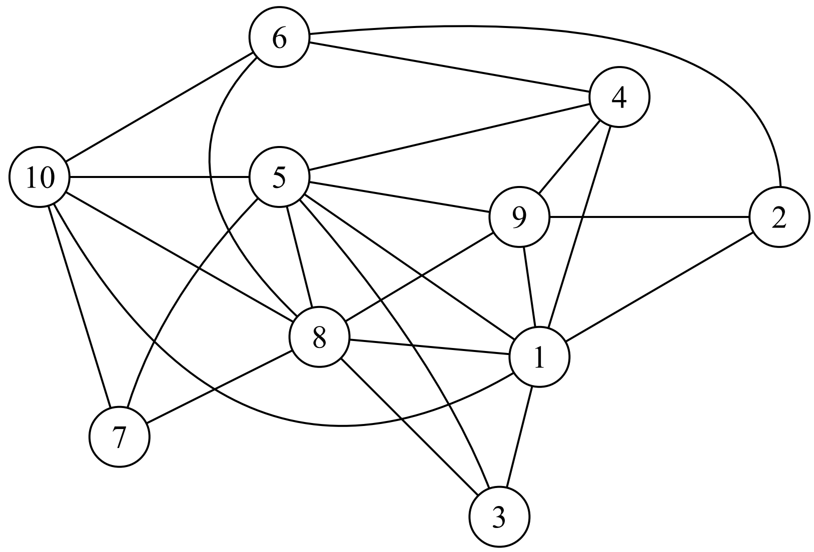

Figure 1: A labeled 3-arch graph on 10 nodes.

Proof. Consider an internal node v in A_k^n: its initial degree is strictly greater than k. The pruning process embodied by ρ ends with a $(k+1)$-clique, where each node has degree k, so either v has been eliminated in some step or it belongs to the remaining clique; in both cases its degree must decrease to k. Since the degree of an internal node v can decrease only if in some step i a k-leaf adjacent to v is removed, v must belong to B_i.

Let us now show that if an element appears in $\rho(A_k^n)$, then it is an internal node. Consider a k-leaf v, and suppose by contradiction that there exists some value i such that $v \in B_i$. This means that after removing a k-leaf on step i, in the resulting graph node v has degree $k-1$. This contradicts the fact that each subgraph produced during the encoding process is k-arch graph. □

Proposition 2. *Function ρ is injective.*

Proof. We have to show that, given two k-arch graphs $A_k^{n\prime}$ and $A_k^{n\prime\prime}$, if $\rho(A_k^{n\prime}) = \rho(A_k^{n\prime\prime}) = (B_1, \ldots, B_{n-k-1})$ then $A_k^{n\prime} = A_k^{n\prime\prime}$.

Let us proceed by induction on $n-k$. If $n-k=1$, $\rho(A_k^{n\prime}) = \rho(A_k^{n\prime\prime}) = \varepsilon$, then $A_k^{n\prime} = A_k^{n\prime\prime}$ as the only k-arch graph on $k+1$ nodes is a $(k+1)$-clique.

For inductive hypothesis, assume the hypothesis holds when $n-k < h$. We have to prove that it holds when $n-k = h$.

In order to have $\rho(A_k^{n\prime}) = \rho(A_k^{n\prime\prime})$, for Lemma 1, the sets of internal nodes and the sets of k-leaves in $A_k^{n\prime}$ and $A_k^{n\prime\prime}$ must coincide. It follows that the minimum k-leaf v_1 in $A_k^{n\prime}$ coincides with the minimum k-leaf in $A_k^{n\prime\prime}$ and both are adjacent to the same node set B_1. Moreover, the graphs obtained by pruning v_1 from $A_k^{n\prime}$ and $A_k^{n\prime\prime}$, in order to produce the same substring $(B_2, \ldots, B_{n-k-1})$, have to be the same graph by inductive hypothesis. This implies $A_k^{n\prime} = A_k^{n\prime\prime}$, as removing the same node and the same edge set from them results in the same graph. □

3 Decoding k-arch Graphs

In this section we show how to revert function ρ in order to rebuild an encoded k-arch graph.

Starting from a string $(B_1, \ldots, B_l)$ that is the encoding of an unknown k-arch graph A_k^n, at first we need to recover values n and k: $k = |B_1| = |B_2| = \cdots = |B_l|$ and, since $l = n - k - 1$, we can derive $n = l + k + 1$. The node set of A_k^n is $[1, n]$ so, to complete the decoding process, we need to reconstruct its edge set.

In view of Lemma 1, it is easy to derive the set of all k-leaves of A_k^n as $[1, n] \setminus \bigcup B_i$. We can compute v_1 (the first k-leaf removed during the encoding process) as the minimum of this set. We also know $adj(v_1) = B_1$.

Now, v_2 is the smallest k-leaf of $A_k^n \setminus \{v_1\}$ and we know both the node set of this k-arch graph (i.e., $[1, n] \setminus \{v_1\}$) and its code string $(B_2, \ldots, B_l)$. Then $v_2 = \min\{v \in [1, n] \setminus \{v_1\} \setminus \bigcup_{i=2}^{l} B_i\}$.

Generalizing this idea it is possible to derive a formula analogous to the one given by Prüfer for trees:

$$v_i = \min \left\{ v \in [1, n] \setminus \{v_h\}_{h<i} \setminus \bigcup_{j \geq i} B_j \right\} \qquad \forall i \in [1, l]$$

Knowing the k-leaf removed at each step of the encoding process it is easy to rebuild the edge set of A_k^n. Indeed, all the $k + 1$ nodes not in $\{v_1, \ldots, v_l\}$ form a clique and each v_i should be connected with all nodes in B_i. We will refer to this decoding process as ρ^{-1}. It is easy to see that the codomain of ρ^{-1} is $\mathcal{A}_k^n$.

Obviously not all strings in $\mathcal{B}_k^n$ are eligible for this decoding procedure. Indeed, it requires the set from which each v_i is chosen to be not empty. To better explain this fact we now show a simple string that does not correspond to the encoding of any k-arch graph; this is in fact the easiest counterexample that proves Lamathe's formula to be not correct.

Consider the string $(\{1, 2\}, \{3, 4\}, \{5, 6\})$: in this case $k = 2$ and $n = 3+2+1 = 6$. Since the set $[1, 6] \setminus (\{1, 2\} \cup \{3, 4\} \cup \{5, 6\})$ is empty, there is no value for v_1, so there can not exist any k-arch graph whose encoding is $(\{1, 2\}, \{3, 4\}, \{5, 6\})$.

It is quite easy to see, from definition of ρ^{-1}, that $\rho^{-1}(\rho(A_k^n)) = A_k^n$ for each k-arch graph A_k^n. We now characterize all those strings in $\mathcal{B}_k^n$ resulting by the encoding of some k-arch graph. Let us call the set of these strings $\mathcal{C}_k^n \subseteq \mathcal{B}_k^n$. Notice that $\mathcal{C}_k^n$ is the image of $\mathcal{A}_k^n$ under function ρ, i.e., $\mathcal{C}_k^n = \rho(\mathcal{A}_k^n)$.

Theorem 3. *Given* $(B_1, \ldots, B_l) \in \mathcal{B}_k^n$ *if* $\exists \{v_1, \ldots, v_l\} \in \binom{[1,n]}{l}$ *such that* $v_i \notin \bigcup_{j=i}^{l} B_j$ *then* $(B_1, \ldots, B_l) \in \mathcal{C}_k^n$.

Proof. The existence of $\{v_1, \ldots, v_l\} \in \binom{[1,n]}{l}$ ensures that the decoding process can be applied, but this is not enough to ensure $(B_1, \ldots, B_l) \in \mathcal{C}_k^n$. Indeed there is a reasonable doubt that the k-arch graph $A_k^n = \rho^{-1}(B_1, \ldots, B_l)$ obtained by decoding an arbitrary string in $\mathcal{B}_k^n$, can produce a different string $(B'_1, \ldots, B'_l) = \rho(A_k^n)$ when encoded. We will show this is not the case.

Without loss of generality assume that $v_1, \ldots, v_l$ coincides with the sequence of nodes chosen by ρ^{-1} at each step during the decoding process. Now, by induction on l, we prove that $\rho(\rho^{-1}(B_1, \ldots, B_l)) = (B_1, \ldots, B_l)$.

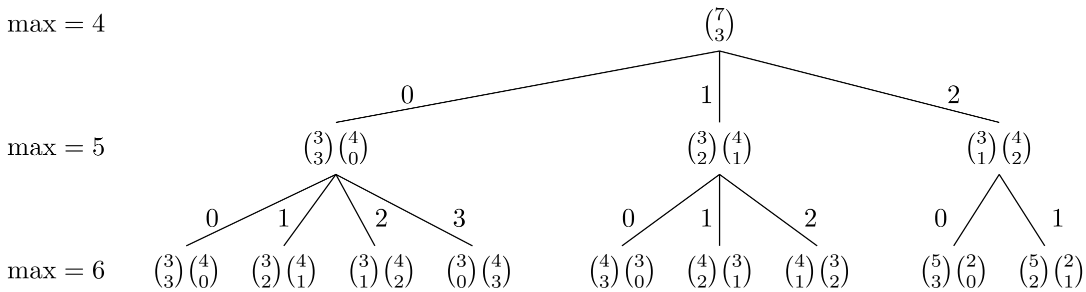

Figure 2: Recursion tree for counting 3-arch graphs on 7 nodes.

When $l = 0$, the string can only be ε, the resulting graph is a $(k+1)$-clique and its encoding is again ε. We assume, by inductive hypothesis, the thesis holds for any string of length $l < h$ and we prove it holds for strings of length $l = h$. First note that if the string $(B_1, \ldots, B_l)$ is decoded as the k-arch graph A_k^n, then the substring $B_2, \ldots, B_l$ is decodable and results in the graph $A_k^{n-1} = A_k^n \setminus \{v_1\}$. By inductive hypothesis $\rho(A_k^{n-1}) = (B_2, \ldots, B_l)$ (here the node set does not contain v_1). The degree of v_1 in A_k^n is $|B_1| = k$, so it is a k-leaf. Any other node with label smaller than v_1 appears in $(B_1, \ldots, B_l)$, as otherwise ρ^{-1} would have done a different choice for v_1. This implies that v_1 is the minimum k-leaf in A_k^n. Then $\rho(A_k^n) = adj(v_1) :: \rho(A_k^{n-1}) = (B_1, \ldots, B_l)$. $\square$

Since in proof of Theorem 3 we proved that $\rho(\rho^{-1}(B_1, \ldots, B_l)) = (B_1, \ldots, B_l)$ for each string in $\mathcal{C}_k^n$, we can state that $\rho^{-1} : \mathcal{C}_k^n \to \mathcal{A}_k^n$ is exactly the inverse function of $\rho : \mathcal{A}_k^n \to \mathcal{C}_k^n$.

4 Counting k-arch Graphs

We are interested in finding the number of k-arch graphs on n nodes, i.e., $|\mathcal{A}_k^n|$. Since $|\mathcal{A}_k^n| = |\mathcal{C}_k^n|$, in order to count labeled k-arch graphs we will count valid code strings. The condition for a string $(B_1, \ldots, B_l)$ to be a valid encoding of a k-arch graph (stated in Theorem 3) can be easily reformulated as:

$$\forall i : 1 \le i \le l, \quad |\bigcup_{h=i}^{l} B_h| \le n - i \tag{1}$$

Exploiting condition of Equation 1, it possible to define a recursive function to count the number of labeled k-arch graphs on n nodes. Before providing this general formula let us show an example of our approach applied to $|\mathcal{C}_3^7|$.

The basic idea is to simulate the generation of a valid code string (B_1, B_2, B_3), from right to left, and count how many choices we have at each step. The choice for B_3 gives $\binom{7}{3}$ alternatives, as Equation 1 requires that no more than 4 different numbers appear in substring (B_3); this substring always contains 3 distinct numbers, then the requirement is always satisfied.

Now consider B_2. Equation 1 requires at most 5 distinct numbers to appear in substring (B_2, B_3), thus imposing some limits on choices for B_2. In fact valid choices are those selecting 3, 2 or 1 numbers appearing in B_3 and respectively 0, 1 or 2 unused numbers, giving $\binom{3}{3}\binom{4}{0}$,

$\binom{3}{2}\binom{4}{1}$ and $\binom{3}{1}\binom{4}{2}$ distinct alternatives. Similar arguments hold for B_1 and constraints depend on how many distinct numbers appear in (B_2, B_3). More explicitly, since Equation 1 imposes to have at most 6 distinct numbers, if u distinct numbers appear in (B_2, B_3), then B_1 can introduce up to $\min(3, 6-u)$ unused numbers.

Figure 2 gives the complete recursion tree for the described process. The root represents choices for B_3; children of the root represent choices for B_2 and leaves choices for B_1. For each level, on the left the bound given by Equation 1 is reported; labels on edges represent how many new numbers are introduced. $|\mathcal{C}_3^7| = 34405$ is given by the sum of the products of labels given by each leaf-to-root path in the tree:

$$\binom{7}{3}\Big(\begin{array}{l} \binom{3}{3}\binom{4}{0}\left(\binom{3}{3}\binom{4}{0}+\binom{3}{2}\binom{4}{1}+\binom{3}{1}\binom{4}{2}+\binom{3}{0}\binom{4}{3}\right)+ \\ \binom{3}{2}\binom{4}{1}\left(\binom{4}{3}\binom{3}{0}+\binom{4}{2}\binom{3}{1}+\binom{4}{1}\binom{3}{2}\right)+ \\ \binom{3}{1}\binom{4}{2}\left(\binom{5}{3}\binom{2}{0}+\binom{5}{2}\binom{2}{1}\right) \end{array} \Big)$$

Now we introduce the main result of this paper.

Theorem 4. *The number of labeled k-arch graph on $n > k+1$ nodes is $|\mathcal{A}_k^n| = f_k^n(n-k-1, 0, k)$ where f_k^n is the recursive function defined as:*

$$f_k^n(i,u,j) = \begin{cases} \binom{n-u}{j}\binom{u}{k-j}, & \text{if } i = 1; \\ \binom{n-u}{j}\binom{u}{k-j}\displaystyle\sum_{c=0}^{\min(k,n-(i-1)-(u+j))} f_k^n(i-1, u+j, c), & \text{otherwise.} \end{cases}$$

When $n = k$ or $n = k+1$ $|\mathcal{A}_k^n| = 1$; when $n < k$ $|\mathcal{A}_k^n| = 0$.

Proof. Given the string $(B_1, \ldots, B_l) \in \mathcal{C}_k^n$, we call *characteristic* of this string the vector $\bar{c} = (c_1, \ldots, c_{l-1})$ such that $c_i = |B_i \setminus \bigcup_{j>i} B_j|$, i.e., the number of elements in B_i that do not appear in the substring $(B_{i+1}, \ldots, B_l)$.

Consider the recursion tree generated by applying the function f_k^n to $(n-k-1, 0, k)$. This tree is a generalization of the one presented in Fig. 2 for the special case $n = 7$ and $k = 3$: node labels correspond to the binomials product and edge labels correspond to the value of the variable c discriminating recursive applications of function f_k^n.

Notice that, considering the edge labels in any leaf to root path of this tree, we obtain a vector $(c_1, \ldots, c_{n-k-2})$ which represents the sequence of newly inserted numbers (from right to left), and so it coincides with the characteristic of some string in $\mathcal{C}_k^n$. It is also true that if $\bar{c}$ is the characteristic of a string in $\mathcal{C}_k^n$, then a leaf to root path whose edge labels vector is $\bar{c}$ must exist.

$|\mathcal{C}_k^n|$ can be obtained by summing cardinalities of disjoint sets of strings sharing the same characteristic. The size of any such set is given by the product of node labels following the corresponding leaf to root path in the recursion tree. By summing those products, we thus obtain $|\mathcal{C}_k^n|$, i.e., the value computed by $f_k^n(n-k-1, 0, k)$. □

4.1 Experimental Results

We implemented the recursive function to enumerate the labeled k-arch graphs on n nodes using the open source algebraic system *PARI/GP* (http://pari.math.u-bordeaux.fr/). The code performing the counting is given in Figure 3.

```
f(n,k,i,u,j)={
    local(s=0);
    if (i==1,
        binomial(n-u,j)*binomial(u,k-j),
        for (c=0, min(k,n-(i-1)-(u+j)),
            s+=f(n,k,i-1,u+j,c)
        );
        binomial(n-u,j)*binomial(u,k-j)*s
    )
}
```

Figure 3: Code implementing the recursive function f_k^n.

The size of the recursion tree is exponential in the order of $(k+1)^{n-k-2}$ so the value can only be computed if the difference between n and k is small. As done by Lamathe we report values of $|\mathcal{A}_k^n|$ for $n \in [1, 10]$ and $k \in [1, 7]$ in the following table:

$k\backslash n$	1	2	3	4	5	6	7	8	9	10
1	1	1	3	16	125	1296	16807	262144	4782969	100000000
2	0	1	1	6	100	3285	177471	14188888	1569185136	229087571625
3	0	0	1	1	10	380	34405	5919536	1709074584	764754595200
4	0	0	0	1	1	15	1085	216230	92550276	74358276300
5	0	0	0	0	1	1	21	2576	982926	898027452
6	0	0	0	0	0	1	1	28	5376	3568950
7	0	0	0	0	0	0	1	1	36	10200

The first row of this table gives exactly the well-known Cayley's formula, as 1-arch graphs are trees. Apart from this row (reported as Sequence A000272) no other row of the table is listed in the on-line Encyclopedia of Integer Sequences [7]. Moreover Sequences A098721, A098722, A098723, and A098724 should be updated to reflect our correction of Lamathe's results (rows 2, 3, 4 of above table respectively).

5 Conclusion and Open Problems

In this paper we have presented a recursive function that computes the number of labeled k-arch graphs of n nodes, for any given n and k. In order to obtain this function, we have used a code that maps labeled k-arch graphs to strings and we have derived the counting function by characterizing valid code strings. Moreover, we have proved the counting function for k-arch graphs provided by Lamathe to be incorrect by showing a counterexample.

It remains an open problem to find, provided that it exists, a closed-form solution for the recursive function computing $|\mathcal{A}_k^n|$, when $k > 1$, perhaps exploiting generating functions. When $k = 1$, from Cayley's formula, we have $|\mathcal{A}_1^n| = n^{n-2}$. Furthermore, it would be interesting to investigate the case of rooted and unlabeled k-arch graphs.

Acknowledgments

We want to thank Prof. Rossella Petreschi for her support and her useful comments.

References

[1] A. Cayley, A theorem on trees, *Quarterly J. Math.* **23** (1889), 376–378.

[2] F. Harary and E. M. Palmer, On acyclic simplicial complexes, *Mathematika* **15** (1968), 115–122.

[3] C. Lamathe, The number of labelled k-arch graphs, *J. Integer Sequences* **7** (2004), Article 04.3.1.

[4] J. W. Moon, Various proofs of Cayley's formula for counting trees, In F. Harary, ed., *A Seminar on Graph Theory*, Holt, Rinehart and Winston, New York, 1967, Chapter 11, pp. 70–78.

[5] H. Prüfer, Neuer Beweis eines Satzes über Permutationen, *Archiv der Mathematik und Physik* **27** (1918), 142–144.

[6] C. Rényi and A. Rényi, The Prüfer code for k-trees, P. Erdős et al., editors, *Combinatoral Theory and its Applications (Proc. Colloq. Balatonfured, 1969)*, North-Holland (1970), 945–971.

[7] N. J. A. Sloane, *The On-Line Encyclopedia of Integer Sequences*, 2007, http://www.research.att.com/~njas/sequences.

[8] P. Todd, A k-tree generalization that characterizes consistency of dimensioned engineering drawings, *SIAM J. Disc. Math.* **2** (1989), 255–261.

2000 *Mathematics Subject Classification*: Primary 05A15, 05C30; Secondary 05A10.
Keywords: k-arch graphs, trees, k-trees, coding, Prüfer code, Cayley's formula.

(Concerned with sequences A000272, A098721, A098722, A098723 and A098724.)

Received November 3 2006; revised version received July 16 2007. Published in *Journal of Integer Sequences*, July 17 2007.
