BIJECTIONS BETWEEN CERTAIN FAMILIES OF LABELLED AND UNLABELLED d-ARY TREES

ALOIS PANHOLZER AND HELMUT PRODINGER

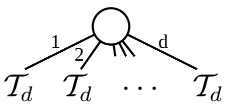
ABSTRACT. We present enumeration results for d-ary trees whose vertices are coloured by k colours in a specific way. Besides generating functions proofs of these results we also give direct bijections between these coloured trees and uncoloured d-ary trees.

1. INTRODUCTION AND RESULTS

We are dealing here with d-ary trees, which are amongst the most fundamental tree structures with applications, e.g., in combinatorics, computer science and biology, see, e.g., [1, 3, 4, 8].

A d-ary tree is either an empty tree or it consists of a root node, to which an ordered sequence of exactly d subtrees is attached that are itself d-ary trees. We denote the family of d-ary trees by $\mathcal{T}_d$ and the empty tree by ε. In particular in the computer science related literature one sometimes uses instead of the symbol ε the notion of external nodes with a certain symbol distinguishable from the proper nodes, called internal nodes. Each node v in the tree has then exactly d children attached to v and we may speak of the first child, ..., the d-th child, where we have to allow "empty children" ε. This recursive description can also be expressed via the following formal equation for $\mathcal{T}_d$, where $\dot\cup$ denotes the disjoint union of two combinatorial families:

$$\mathcal{T}_d \quad = \quad \varepsilon \quad \dot\cup \qquad \tag{1}$$



It is well known and can be shown in many ways that the number $T_n^{(d)}$ of d-ary trees of size n, where the size $|T|$ of a tree T is here always measured by its number of nodes, is given by the generalized Catalan numbers:

$$T_n^{(d)} = \frac{1}{(d-1)n+1}\binom{dn}{n}, \quad \text{for } n \geq 0. \tag{2}$$

Date: March 27, 2008.

2000 *Mathematics Subject Classification.* 05C05, 05A15.

Key words and phrases. labelled trees, d-ary trees, bijections.

This work was supported by the South African Science Foundation NRF, grant 2053748, and by the Austrian Science Foundation FWF, grant S9608-N13. The second author wants to thank the Department of Mathematical Sciences, University of Stellenbosch, for its support and hospitality during a research visit, where this work has been carried out.

For the special instance $d = 2$, i.e., binary trees where each node has a left and a right child, several studies of certain subclasses of two-coloured binary trees appeared. In two-coloured binary trees the nodes in the tree are coloured either black or white. In [2, 5, 6] remarkable enumeration formulæ and relations between subclasses of two-coloured trees, where restrictions on the colours of two connected nodes are made, and other combinatorial objects are obtained. In particular in [2, 6] it has been shown in a bijective way that the number of $\bullet\!\searrow\!\circ$-free two-coloured binary trees with a black root and of size $n \geq 1$ are equal to the number of (uncoloured) ternary trees of size n. Here a two-coloured binary tree is called $\bullet\!\searrow\!\circ$-free, when there is no edge occurring in the tree that connects a parent coloured black with a right child coloured white. The bijection presented in [6] gives a procedure, which is easy to implement and that allows to encode (and decode) a ternary tree by a $\bullet\!\searrow\!\circ$-free two-coloured binary tree with a black root of the same size.

This motivated us to have a closer look on relations between certain subclasses of coloured trees and uncoloured trees, leading to generalizations in two directions: first we deal now with arbitrary d-ary trees, for $d \geq 2$, and second we consider k-coloured trees (or k-labelled trees), i.e., there is a set of k colours (or labels), $k \geq 2$, and each node in a tree has a colour (label) from this set.

Although we could deal directly with k-coloured d-ary trees, we will present our generalizations in two steps, which should improve the readability: first we consider two-coloured d-ary trees, and only then we consider the general case of k-labelled d-ary trees.

So, first we deal with two-coloured d-ary trees, where the nodes in a tree are coloured either black or white. We introduce here the notion of $\bullet\!\searrow\!\circ$-free two-coloured d-ary trees (by generalizing the definiton for binary trees), whose meaning is now, that there is no edge occurring in the tree that connects a parent coloured black with a d-th child (i.e., the rightmost child) coloured white. The family of $\bullet\!\searrow\!\circ$-free two-coloured d-ary trees with a black root is denoted here by $\mathcal{B}_d$. An example of a tree in $\mathcal{B}_3$ is given in Figure 1.

We state now our enumeration result for trees in $\mathcal{B}_d$ of a given size, which relates black-rooted $\bullet\!\searrow\!\circ$-free two-coloured d-ary trees with ordinary $(2d-1)$-ary trees.

Theorem 1. *The number $B_n^{(d)}$ of trees in $\mathcal{B}_d$ of size $n \geq 1$ is equal to the number $T_n^{(2d-1)}$ of trees in $\mathcal{T}_{2d-1}$ of size n and is thus given by*

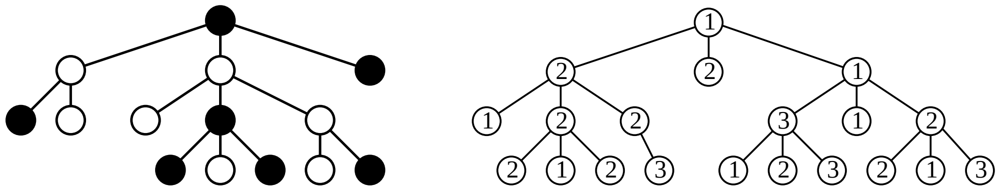
$$B_n^{(d)} = \frac{1}{(2d-2)n+1} \binom{(2d-1)n}{n}.$$

Next we consider the general case of k-coloured d-ary trees, where we have a set of k colours. For simplicity in presentation we use the set of colours $\{1, 2, \ldots, k\}$ and consider thus d-ary trees, which are labelled with labels from $\{1, 2, \ldots, k\}$.

We introduce now the notion of $\textcircled{i}\!\overset{>}{\searrow}\!\textcircled{j}$-free k-coloured d-ary trees, whose nodes are labelled with labels from the set $\{1, 2, \ldots, k\}$ in such a way that each node labelled i, $1 \leq i \leq k$, does not have a d-th child labelled $j < i$. We denote by $\mathcal{G}_{d,k}$ the family of $\textcircled{i}\!\overset{>}{\searrow}\!\textcircled{j}$-free k-coloured d-ary trees.

FIGURE 1.

An example of a -free two-coloured binary tree of size 14 with a black root and an example of a -free 3-coloured ternary tree of size 20.



For an arbitrary labelled or unlabelled d-ary tree T we define the *rightmost path* of T as the path $v_0, \ldots, v_r$, where v_0 is the root of T, v_i, for $1 \leq i \leq r$, is the d-th child of v_{i-1}, and v_r only has an empty d-th child. We can give then an alternative definition of the family $\mathcal{G}_{d,k}$ as the family of k-coloured d-ary trees T with the property, that the rightmost path of any subtree of T consists of a sequence of non-decreasing labels. An example of a tree in $\mathcal{G}_{3,3}$ is given in Figure 1.

We state now our first enumeration result for trees in $\mathcal{G}_{d,k}$ of a given size, which relates the subclass of -free k-coloured d-ary trees whose roots are labelled by k with ordinary $(k(d-1)+1)$-ary trees.

Theorem 2. *The number $G_n^{[k]}$ of those trees in $\mathcal{G}_{d,k}$ of size $n \geq 1$ whose roots are labelled by k is equal to the number $T_n^{(k(d-1)+1)}$ of trees in $\mathcal{T}_{k(d-1)+1}$ of size n and is thus given by*

$$G_n^{[k]} = \frac{1}{k(d-1)n+1} \binom{(k(d-1)+1)n}{n}.$$

Of course, Theorem 1 follows from Theorem 2, where we identify a white node with label 1 and a black node with label 2.

As a corollary we obtain a second enumeration result for trees in $\mathcal{G}_{d,k}$ of a given size, which relates the whole set of -free k-coloured d-ary trees with k-tuples of ordinary $(k(d-1)+1)$-ary trees.

Corollary 1. *The number G_n of trees in $\mathcal{G}_{d,k}$ of size $n \geq 0$ is equal to the number of k-tuples $(T_1, \ldots, T_k)$ of trees in $\mathcal{T}_{k(d-1)+1}$ with total size $|T_1| + \cdots + |T_k| = n$ and is thus given by*

$$G_n = \frac{k}{k(d-1)n+k} \binom{(k(d-1)+1)n+k-1}{n}.$$

For all our results we give a proof via generating functions, but furthermore we also present bijective proofs for our findings. E.g., when considering the special instance of k-coloured binary trees, our bijection gives a procedure, which is easy to implement and that allows to encode (and decode) a $(k+1)$-ary tree by a -free k-coloured binary tree with a root node labelled by k of the same size.

A proof of Theorem 1 is given in Section 2, whereas Theorem 2 and Corollary 1 are proven in Section 3.

2. TWO-COLOURED d-ARY TREES

2.1. **A generating functions proof of Theorem 1.** We begin by stating the well-known fact that the generating function $T_d(z) = \sum_{n\ge 1} T_n^{(d)} z^n$ of non-empty d-ary trees satisfies the functional equation

$$T_d(z) = z(1+T_d(z))^d. \tag{3}$$

This equation can be obtained, e.g., directly from the formal equation (1), see [7] for a description of this symbolic method. An application of the Lagrange inversion formula, see, e.g., [8], gives then immediately the enumeration result (2) for $T_n^{(d)}$:

$$T_n^{(d)} = [z^n]T_d(z) = \frac{1}{n}[T^{n-1}](1+T)^{dn} = \frac{1}{n}\binom{dn}{n-1} = \frac{1}{(d-1)n+1}\binom{dn}{n}. \tag{4}$$

Now we are going to enumerate the number $B_n^{(d)}$ of trees in $\mathcal{B}_d$ of size n, i.e., of black-rooted two-coloured -free d-ary trees. To do this we introduce the auxiliary family $\mathcal{W}_d$ of two-coloured -free d-ary trees with a white root. The number of trees in $\mathcal{W}_d$ of size n is denoted by $W_n^{(d)}$.

Next we introduce the generating functions (where we drop the dependence of these functions on z and d, for a better readability):

$$B := \sum_{n\ge 1} B_n^{(d)} z^n, \quad \text{and} \quad W := \sum_{n\ge 1} B_n^{(d)} z^n.$$

We use now the formal equation (1) of d-ary trees, which gives a decomposition of a tree according to the root node, but additionally we take into account that, for a root node coloured black, the d-th child, if it is non-empty, must be coloured black also. This decomposition leads then immediately to the following system of equations for the functions B and W:

$$W = z(1+B+W)^d, \tag{5a}$$

$$B = z(1+B+W)^{d-1}(1+B). \tag{5b}$$

Multiplying equation (5a) with $(1+B)$ and plugging in (5b) we obtain

$$(1+B)W = z(1+B+W)^d(1+B) = B(1+B+W),$$

which gives by subtracting BW on both sides:

$$W = B(1+B). \tag{6}$$

When we plug in (6) into equation(5b) we obtain the following functional equation for B:

$$B = z(1+B+B(1+B))^{d-1}(1+B) = z(1+B)^{d-1}(1+B)^{d-1}(1+B) = z(1+B)^{2d-1}. \tag{7}$$

Thus we obtain exactly the functional equation, which is satisfied by the generating function $T^{(2d-1)}(z)$ of the number of non-empty $(2d-1)$-ary trees $T_n^{(2d-1)}$ of size n, see (3). This, together with equation (4), shows then Theorem 1.

2.2. **A bijective proof of Theorem 1.** Consider a tree C in $\mathcal{B}_d$. We will give now a simple recursive procedure, which eventually leads to an unlabelled $(2d-1)$-ary tree of the same size.

Consider the d (possibly empty) subtrees $C_1, \dots, C_d$ of the root of C. The basic idea is to split each of the first $d-1$ subtrees $C_1, \dots, C_{d-1}$ in exactly two (possibly empty) trees, i.e., the tree C_i, $1 \le i \le d-1$, will be split into the pair of trees $C_{i,1}$ and $C_{i,2}$. This sequence of trees $C_{1,1}$, $C_{1,2}$, $C_{2,1}$, $C_{2,2}$, $\dots$, $C_{d-1,1}$, $C_{d-1,2}$ together with the remaining subtree C_d will be attached, in this order, to the (now uncoloured) root node forming the $2d-1$ subtrees.

To split any subtree C_i into two parts we use that C is -free, which gives in particular, that the rightmost path of C_i consists of a possibly empty sequence of white nodes, followed by a (possibly empty) sequence of black nodes. By cutting the possibly occurring edge , which connects a white parent node with a d-th child coloured black, we obtain then the pair of (possibly empty) trees $C_{i,1}$ and $C_{i,2}$, where all nodes on the rightmost path of $C_{i,1}$ are coloured white and all nodes on the rightmost path of $C_{i,2}$ are coloured black.

When we apply this procedure recursively to all of the subtrees obtained this leads to a $(2d-1)$-ary tree T of the same size as C. An example that illustrates this procedure is given in Figure 2.

We retain here from giving the description of the inverse procedure, since it follows from the general case as special instance $k=2$ (and the convention, that a white node corresponds to a labelling by 1 and a black node to a labelling by 2), and this procedure for the general case is presented in Subsection 3.2.

3. k-LABELLED d-ARY TREES

3.1. **A proof of Theorem 2 via generating functions.** We consider now the family $\mathcal{G}_{d,k}$ of -free k-coloured d-ary trees. Let us denote by $G_n^{[i]}$, $1 \le i \le k$, the number of those trees in $\mathcal{G}_{d,k}$ of size $n \ge 1$ whose roots are labelled by i. Furthermore we introduce the generating functions (again we drop the dependence of these functions on z and d):

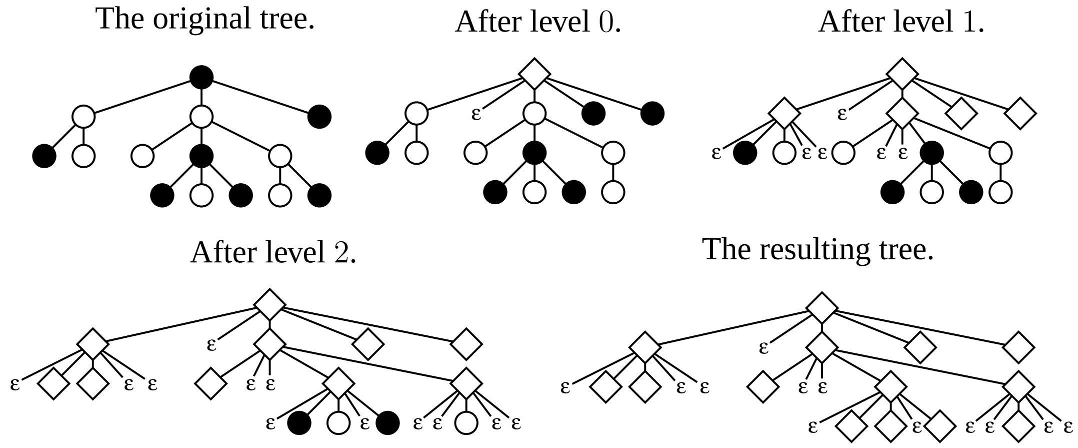
$$G_1 := \sum_{n \ge 1} G_n^{[1]} z^n, \quad \dots, \quad G_k := \sum_{n \ge 1} G_n^{[k]} z^n.$$

We use now the formal equation (1) of d-ary trees, which gives a decomposition of a tree according to the root node, but here we have to take into account that, for a root node labelled i, the d-th child, if it is non-empty, must be labelled by a label from the set $\{i, \dots, k\}$. This decomposition leads then immediately to the following system of equations for the functions G_i, $1 \le i \le k$:

$$\begin{aligned} G_1 &= z(1 + G_1 + \dots + G_k)^d, \\ G_2 &= z(1 + G_1 + \dots + G_k)^{d-1}(1 + G_2 + G_3 + \dots + G_k), \end{aligned}$$

FIGURE 2.

An illustrating example of the bijection presented between a black-rooted $\bullet\!\!\!\searrow\!\circ$-free two-coloured ternary tree of size 14 and a 5-ary tree of the same size. The recursive procedure is here performed level by level, where, for a better readability, the nodes in the resulting uncoloured 5-ary tree are drawn as diamonds. We omitted here to draw the empty subtrees ε in the original tree as well as for the leaves of the resulting tree.



$$\begin{aligned} &\vdots \\ G_{k-1} &= z(1+G_1+\cdots+G_k)^{d-1}(1+G_{k-1}+G_k), \\ G_k &= z(1+G_1+\cdots+G_k)^{d-1}(1+G_k). \end{aligned} \tag{8}$$

Next we will show by induction on i that the functions G_{k-i} can be expressed by G_k via

$$G_{k-i} = G_k(1+G_k)^i, \quad 1 \le i \le k-1. \tag{9}$$

First we show (9) for $i = 1$. To do this we multiply the $(k-1)$-th equation of (8) with $1 + G_k$, which gives

$$(1+G_k)G_{k-1} = z(1+G_1+\cdots+G_k)^{d-1}(1+G_k)(1+G_{k-1}+G_k) = G_k(1+G_{k-1}+G_k),$$

and further the desired equation by subtracting $G_{k-1}G_k$ on both sides:

$$G_{k-1} = G_k(1+G_k). \tag{10}$$

Now we assume that (9) holds for all j, with $1 \le j < i \le k-1$. We multiply the $(k-i)$-th equation of (8) with $1 + G_{k-i+1} + \cdots + G_k$, which gives

$$\begin{aligned} &(1+G_{k-i+1}+\cdots+G_k)G_{k-i} \\ &\quad = z(1+G_1+\cdots+G_k)^{d-1}(1+G_{k-i+1}+\cdots+G_k)(1+G_{k-i}+\cdots+G_k) \\ &\quad = G_{k-i+1}(1+G_{k-i}+\cdots+G_k) \end{aligned}$$

and further by subtracting $G_{k-i}G_{k-i+1}$ on both sides:

$$(1 + G_{k-i+2} + \cdots + G_k)G_{k-i} = G_{k-i+1}(1 + G_{k-i+1} + \cdots + G_k). \tag{11}$$

We use now the induction hypothesis and evaluate the following sum, for $1 \leq j < i$:

$$\begin{aligned} 1 + G_{k-j} + \cdots + G_k &= 1 + G_k + \sum_{\ell=1}^{j} G_k(1 + G_k)^\ell = 1 + G_k \sum_{\ell=0}^{j} (1 + G_k)^\ell \\ &= 1 + G_k \left(\frac{(1 + G_k)^{j+1} - 1}{G_k} \right) = (1 + G_k)^{j+1}. \end{aligned} \tag{12}$$

When we plug in (12) into both sides of equation (11) we obtain then the desired equation, which finishes the proof of (9) by induction:

$$G_{k-i} = G_k(1 + G_k)^i. \tag{13}$$

Since we have shown now equation (9) for all i, with $1 \leq i \leq k - 1$, we can evaluate the following sum analogous to (12) and obtain

$$1 + G_1 + \cdots + G_k = (1 + G_k)^k. \tag{14}$$

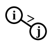
When we plug in equation (14) into the last equation of (8) we get the following functional equation for G_k:

$$G_k = z(1 + G_k)^{k(d-1)+1}. \tag{15}$$

Thus we obtain for G_k exactly the functional equation, which is satisfied by the generating function $T_{k(d-1)+1}(z)$ of the number of non-empty $(k(d-1)+1)$-ary trees $T_n^{(k(d-1)+1)}$ of size n, see (3). This, together with equation (4), shows thus Theorem 2.

3.2. **A bijective proof of Theorem 2.** Consider a tree H in $\mathcal{G}_{d,k}$ whose root is labelled by k. We will give now a simple recursive procedure, which eventually leads to an unlabelled $(k(d-1)+1)$-ary tree of the same size.

Consider the d (possibly empty) subtrees $H_1, \ldots, H_d$ of the root of H. The basic idea is again to split each of the first $d-1$ subtrees $H_1, \ldots, H_{d-1}$, but now in exactly k (possibly empty) trees, i.e., the tree H_i, $1 \leq i \leq d-1$, will be split into a k-tuple of trees $H_{i,1}$, $H_{i,2}$, $\ldots$, $H_{i,k}$. This sequence of trees $H_{1,1}$, $\ldots$, $H_{1,k}$, $H_{2,1}$, $\ldots$, $H_{2,k}$, $H_{3,1}$, $\ldots$, $H_{d-1,k}$ together with the remaining subtree H_d will be attached, in this order, to the (now uncoloured) root node forming the $k(d-1)+1$ subtrees.

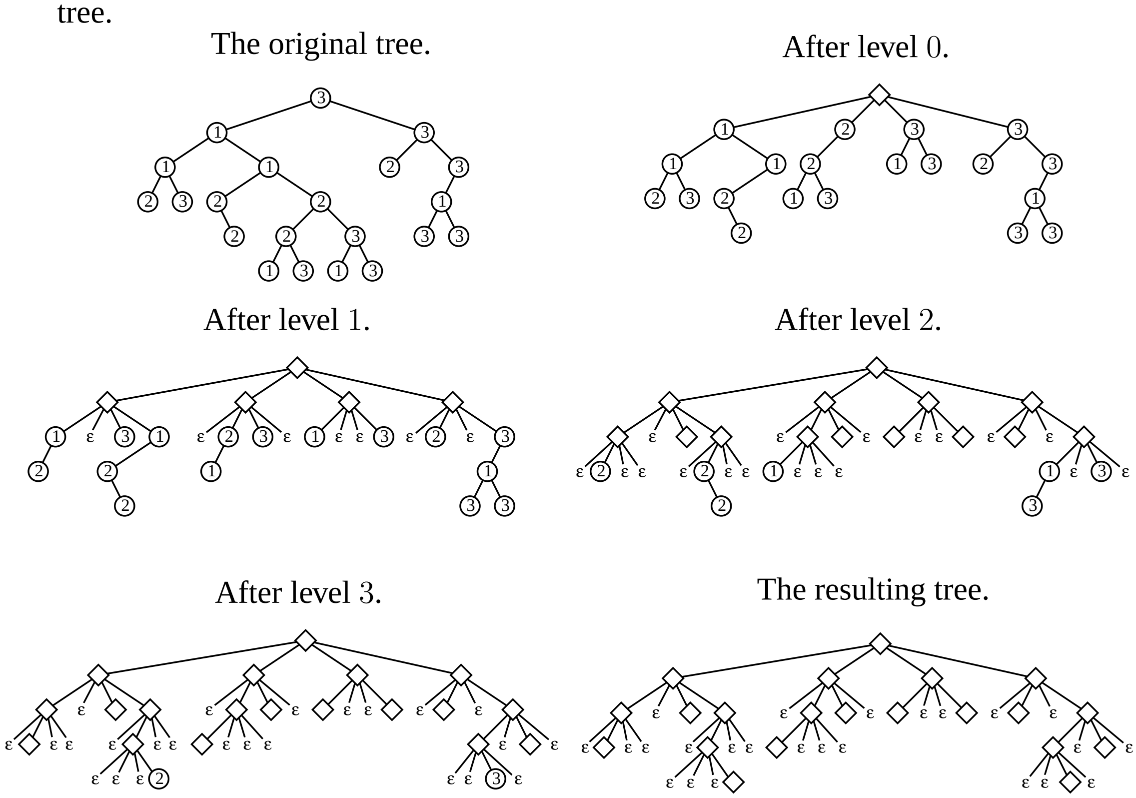
To split any subtree H_i into k parts we use that the rightmost path of H_i is 

-free and thus forming a non-decreasing sequence of labels, i.e., a possibly empty sequence of 1, followed by a (possibly empty) sequence of 2, and so on, and ending by a (possibly empty) sequence of k. By cutting each edge on the rightmost path that is connecting two nodes with an unequal label, we obtain then the sequence of (possibly empty) trees $H_{i,1}$, $\ldots$, $H_{i,k}$, where all nodes on the rightmost path of $H_{i,\ell}$ are labelled by ℓ, $1 \leq \ell \leq k$.

When we apply this procedure recursively to all of the subtrees obtained this leads to a $(k(d-1)+1)$-ary tree T of the same size as H. An example that illustrates this procedure is given in Figure 3.

FIGURE 3.

An illustrating example of the bijection presented between a ①ⓘ-free 3-coloured binary tree of size 21 and a 4-ary tree of the same size. The recursive procedure is here performed level by level, where, for a better readability, the nodes in the resulting uncoloured 4-ary tree are drawn as diamonds. We omitted here to draw the empty subtrees ε in the original tree as well as for the leaves of the resulting tree.



Next we describe the inverse procedure and consider a non-empty tree T in $\mathcal{T}_{k(d-1)+1}$, which will be converted eventually into a ①ⓘ-free k-coloured d-ary tree whose root is labelled by k.

First we will label the nodes of the tree as follows. The root of T will be labelled by k. Then, for any node v in T, we are labelling all children of v by carrying out the following procedure recursively, where we are starting with the root node of T. The $(k(d-1)+1)$-th child (if non-empty) of a node v will be labelled by the same label as the parent node v. Furthermore the $(k(i-1)+\ell)$-th child (if non-empty) of v, for $1 \le i \le d-1$ and $1 \le \ell \le k$, will be labelled by ℓ. Thus the first $k(d-1)$ children of a node v are labelled by the sequence $1, 2, \ldots, k, 1, 2, \ldots, k, \ldots$.

Second, we will carry out the following recursive procedure for the now k-coloured $(k(d-1)+1)$-ary tree T. Consider the $k(d-1)+1$ (possibly empty) subtrees $S_1, \ldots, S_{k(d-1)+1}$ of the root of T. The basic idea is now to merge always k consecutive subtrees, namely $S_{k(i-1)+1}, \ldots, S_{k(i-1)+k}$ for any i with $1 \leq i \leq d-1$, into a single tree H_i. This sequence of trees $H_1, \ldots, H_{d-1}$ together with the remaining subtree $S_{k(d-1)+1}$ will be attached, in this order, to the root node forming the d subtrees.

To merge the subtrees $S_{k(i-1)+1}, \ldots, S_{k(i-1)+k}$ we use a simple consequence of the labelling done before, namely that the nodes on the rightmost path of the $(k(d-1)+1)$-ary tree $S_{k(i-1)+\ell}$ are, for $1 \leq i \leq d-1$ and $1 \leq \ell \leq k$, all labelled by ℓ. Thus we can simply concatenate the subtrees $S_{k(i-1)+1}, \ldots, S_{k(i-1)+k}$ in that order by connecting the rightmost paths of these trees leading to a single tree H_i whose rightmost path is $\textcircled{\scriptsize 1}\overset{>}{\searrow}\textcircled{\scriptsize j}$-free.

When we apply this procedure recursively to the subtrees obtained, this leads to a $\textcircled{\scriptsize 1}\overset{>}{\searrow}\textcircled{\scriptsize j}$-free k-coloured d-ary tree H whose root is labelled by k, which is of the same size as T. An example that illustrates this inverse procedure is given in Figure 4.

It is seen easily that this is indeed the inverse procedure to the previously given one and thus we obtained a bijection between the tree families considered.

3.3. **A proof of Corollary 1.** This result follows easily from the generating functions proof of Theorem 2 carried out in Subsection 3.1. We consider here the generating function $G(z) = \sum_{n\geq 0} G_n z^n$ of the number G_n of $\textcircled{\scriptsize 1}\overset{>}{\searrow}\textcircled{\scriptsize j}$-free k-coloured d-ary trees of size n, which can be expressed as $G(z) = 1 + G_1 + \cdots + G_k$, with generating functions G_i, $1 \leq i \leq k$, as defined in Subsection 3.1 (we drop here the dependence of G_n and $G(z)$ on k and d).

Due to equation (14) we have the following relation between $G(z)$ and the corresponding generating function G_k for trees whose roots are labelled by k:

$$G(z) = (1 + G_k)^k.$$

Since $1 + G_k = 1 + T_{k(d-1)+1}$ is also the generating function of (possibly empty) $(k(d-1)+1)$-ary trees it follows that $G(z)$ corresponds to the generating function of k-tuples of $(k(d-1)+1)$-ary trees of total size n. The formula for G_n stated in Corollary 1 follows immediately by using the Lagrange inversion formula:

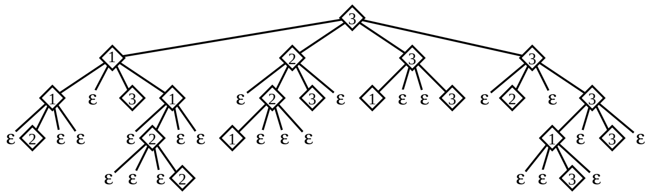
$$\begin{aligned} G_n = [z^n]G(z) &= \frac{k}{n}[G^{n-1}](1+G)^{n(k(d-1)+1)}(1+G)^{k-1} \\ &= \frac{k}{n}[G^{n-1}](1+G)^{n(k(d-1)+1)+k-1} = \frac{k}{k(d-1)n+k}\binom{(k(d-1)+1)n+k-1}{n}. \end{aligned}$$

However, a bijective proof of Corollary 1 can also be given. Consider a tree H in $\mathcal{G}_{d,k}$ of size n. We split now the tree H into k (possibly empty) trees $H_1, \ldots, H_k$ by cutting each edge on the rightmost path of H that connects nodes with unequal labels (all nodes on the rightmost path of H_i are labelled by i, $1 \leq i \leq k$). This works, since H has an $\textcircled{\scriptsize 1}\overset{>}{\searrow}\textcircled{\scriptsize j}$-free colouring.

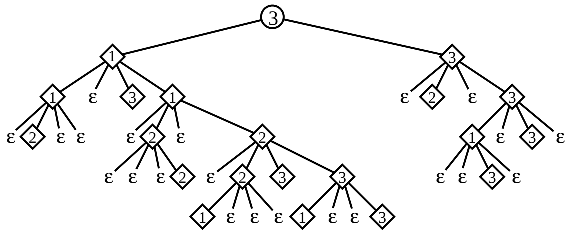
FIGURE 4.

An illustrating example of the bijection presented between a 4-ary tree of size 21 and a $\textcircled{i} \overset{>}{\searrow} \textcircled{j}$-free 3-coloured binary tree of the same size. The recursive procedure is here performed level by level, where, for a better readability, the nodes in the original 4-ary tree are drawn as diamonds. We omitted here to draw the empty subtrees ε in the resulting tree as well as for the leaves of the original tree.

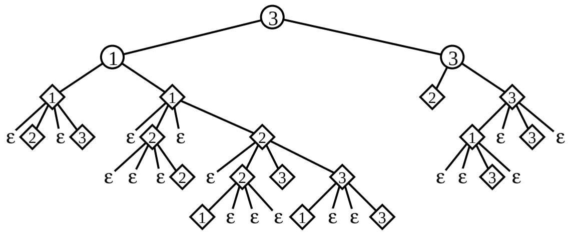
The original tree after the labelling.



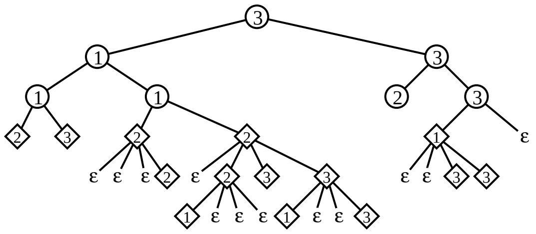
After level 0.



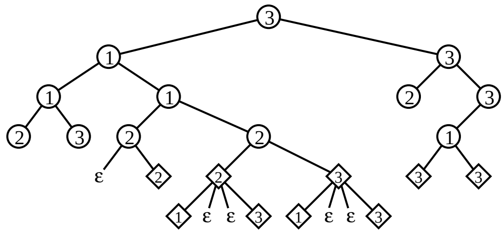
After level 1.



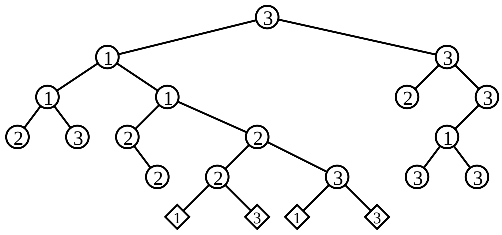
After level 2.



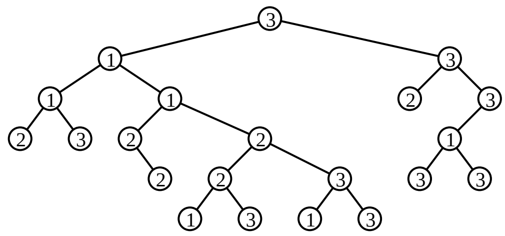
After level 3.



After level 4.



The resulting tree.



Now we can apply the recursive procedure described in the bijective proof of Theorem 2 in Subsection 3.2 to each tree H_i, $1 \leq i \leq k$, which maps H_i bijectively to a $(k(d-1)+1)$-ary tree T_i of the same size as H_i. Thus we obtain a bijection between H and a k-tuple $(T_1, \ldots, T_k)$ of $(k(d-1)+1)$-ary trees of total size n.

REFERENCES

[1] M. Blum, O. François, and S. Janson, The mean, variance and limiting distribution of two statistics sensitive to phylogenetic tree balance, *The Annals of Applied Probability* 16, 2195–2214, 2006.

[2] N. Gu, N. Li, and T. Mansour, 2-Binary trees: Bijections and related issues, *Discrete Mathematics* 308, 1209–1221, 2008.

[3] R. Kemp, *Fundamentals of the average case analysis of particular algorithms*, John Wiley & Sons, Chichester, 1984.

[4] D. Knuth, *The art of computer programming, vol. 1*, third edition, Addison-Wesley, Reading, 1997.

[5] J. Pallo, On the listing and random generation of hybrid binary trees, *International Journal of Computer Mathematics* 50, 135–145, 1994.

[6] H. Prodinger, A simple bijection between a subclass of 2-binary trees and ternary trees, *Discrete Mathematics*, to appear, 2008.

[7] R. Sedgewick and P. Flajolet, *An introduction to the Analysis of Algorithms*, Addison-Wesley, Reading, 1996.

[8] R. Stanley, *Enumerative Combinatorics, vol. 2*, Cambridge University Press, Cambridge, 1999.

ALOIS PANHOLZER, INSTITUT FÜR DISKRETE MATHEMATIK UND GEOMETRIE, TECHNISCHE UNIVERSITÄT WIEN, WIEDNER HAUPTSTR. 8-10/104, 1040 WIEN, AUSTRIA

E-mail address: `Alois.Panholzer@tuwien.ac.at`

HELMUT PRODINGER, MATHEMATICS DEPARTMENT, STELLENBOSCH UNIVERSITY, 7602 STELLENBOSCH, SOUTH AFRICA

E-mail address: `hproding@sun.ac.za`