ELSEVIER

Discrete Mathematics 256 (2002) 719–741

DISCRETE MATHEMATICS

www.elsevier.com/locate/disc

Symmetries in trees and parking functions

Louis H. Kalikow*

Department of Mathematics and Computer Science, College of the Holy Cross, Worcester, MA 01610, USA

Received 29 September 2000; received in revised form 18 May 2001; accepted 22 July 2001

Abstract

We investigate a particular symmetry in labeled trees first discovered by Gessel, which can be stated as follows: In the set of rooted labeled trees on $n+1$ vertices rooted at the smallest vertex, the number of trees with a descents and $b+1$ leaves equals the number of trees with b descents and $a+1$ leaves. We present two new ways to prove the symmetry resulting from decompositions of trees, which lead to three different bijections from trees to trees in which leaves and descents are swapped. We also interpret the symmetry in terms of parking functions: the number of parking functions on $[n]$ with a descents and b unfavorable spaces (defined in this paper) equals the number of parking functions on $[n]$ with b descents and a unfavorable spaces. We conclude with a generalization of these results to binary trees.

Résumé

Nous étudions une symétrie particulière dans les arbres étiquetés, découverte par Gessel, qu'on peut énoncer comme suit: Dans l'ensemble des arbres étiquetés pointés avec $n+1$ sommets, pointés au sommet minimum, le nombre d'arbres avec a descentes et $b+1$ feuilles égale le nombre d'arbres avec b descentes et $a+1$ feuilles. Nous présentons deux nouvelles démonstrations de la symétrie, qui resultent des décompositions des arbres; à partir des décompositions, nous obtenons trois bijections des arbres sur les arbres qui échangent les feuilles et les descentes. De plus, nous interprétons la symétrie en termes des "fonctions de stationnement" (parking functions): le nombre des fonctions de stationnement avec a descentes et b positions défavorables (définies dans cette article) égale le nombre de fonctions de stationnement avec b positions défavorable et a descentes. Nous donnons aussi une généralisation de ces resultats aux arbres binaires.

Keywords: Trees; Parking functions; Decents; Leaves; Binary trees

* Corresponding author. Department of Mathematics, The George Washington University, Washington, DC 20052, USA.

E-mail address: lkalikow@gwu.edu (L.H. Kalikow).

0012-365X/02/$ - see front matter
PII: S0012-365X(02)00344-8

1. Introduction

A *labeled tree* is a tree whose vertices are labeled with elements of some set. We shall consider trees labeled by non-negative integers. In particular, if i is an integer and S a finite set of integers for which $i \notin S$, then $\mathcal{T}_{i,S}$ will denote the set of rooted labeled trees on $|S|+1$ vertices with root labeled i and other vertices labeled by the elements of S (with each label used exactly once). In a rooted labeled tree, the *parent* of v is the first vertex after v on the unique path from v to the root; v is a *child* of the vertex w if w is the parent of v. Two vertices w and v are *siblings* if they have the same parent.

We will consider two statistics. A *leaf* of a tree is a vertex with no children. A *descent* of a labeled tree is a vertex whose label is greater than at least one of its children's labels (see [4,6]). We will be considering two sets of trees, $\mathcal{T}_{0,[n]}$ and $\mathcal{T}_{n,[n-1]}$ (where we use $[n] := \{1,2,\ldots,n\}$).

We will first focus on the fact, first discovered by Gessel [4], that the number of trees in $\mathcal{T}_{0,[n]}$ with a descents and $b+1$ leaves equals the number of trees in $\mathcal{T}_{0,[n]}$ with b descents and $a+1$ leaves. Gessel's proof used marked forests, instead of directly counting the trees. In Sections 2 and 3, we give two new proofs which use decompositions of trees. The decompositions lead to a generalization of a well-known bijection from permutations of $[n]$ to increasing trees in $\mathcal{T}_{0,[n]}$ (i.e., trees with no descents). They also lead to a generalization, discussed in Section 4, of an identity found by Knuth [10] involving *intransitive* trees, which are labeled trees in which all the vertices adjacent to i (i.e., the parent and the children of i) are either less than i or greater than i (see [16, p. 90]).

It turns out the first decomposition for trees is related to a symmetry in parking functions. A *parking function on* $[n]$ is a function $p:[n]\to[n]$ such that for all $j\in[n]$, $|\{i\in[n] \mid p(i)\leqslant j\}|\geqslant j$. Parking functions were first considered by Konheim and Weiss [11]. Their name comes from the following colorful (and equivalent) description: Consider n cars numbered 1 to n driving down a one-way street containing n spaces numbered 1 to n. Each car i has a preferred space $p(i)$ it wishes to park in. In increasing order of i, the cars try to park. When car i goes to park, it goes straight to space $p(i)$; if that space is taken, the car proceeds forward looking for the next empty space. If no space is empty, it drives away. The parking functions are the preference functions $p:[n]\to[n]$ such that all cars park using the above algorithm. For more on parking functions, see [1,2,7,12] (the last deals with *suites majeures*, which are equivalent to parking functions).

We define some terms related to parking functions. For a parking function p on $[n]$, the permutation of parked cars resulting from running the parking algorithm is called the *output permutation* of the parking function. We then define a *descent* of a parking function to be a descent of the output permutation. Next, for a parking function $p\in P_n$, let $p([n])$ denote the range of the function. We call an element of $[n]-p([n])$ an *unfavorable space*. In other words, an unfavorable space is a space in which no car prefers to park. For example, for the parking function $p(1)=2$, $p(2)=2$, $p(3)=1$, $p(4)=3$, the output permutation is 3124, and p has 1 unfavorable space (space 4) and 1 descent (we do not consider the last element of the output permutation to be a descent).

T reduction of T with respect to $[4]$

Fig. 1. A tree on 4 vertices and its reduction with respect to the set [4].

In Section 5, we will show that number of parking functions on $[n]$ with a descents and b unfavorable spaces equals the number of parking functions on $[n]$ with b descents and a unfavorable spaces. Finally, in Section 6, we discuss how these results can be generalized to binary trees.

The following very useful definitions follow Gessel [3]. Given any object with n different labels taken from an arbitrary totally ordered finite set $S=\{x_1<x_2<\cdots<x_n\}$, we can define the *content* of the object to be the set S. We define the *reduction* of the object *with respect to a set of n consecutive integers* $\{a,a+1,\ldots,a+(n-1)\}$ to be the object obtained by replacing the label x_i with the label $a+(i-1)$. We note that a tree with labels in a totally ordered set is determined by its content and its reduction with respect to a given set of consecutive integers. For example, Fig. 1 shows a tree T with content $\{2,4,7,8\}$ and its reduction with respect to the set $[4]=\{1,2,3,4\}$.

Unless otherwise specified, the reduction of a rooted labeled tree with n non-root vertices will be taken with respect to $[n]\cup\{0\}$. In addition, we can generalize parking functions to functions $p:A\to[n]$ where A is a totally ordered set of n elements. We can then take the reduction of such a function to obtain a parking function on $[n]$. The reduction of a parking function on a set A with n elements will always be taken with respect to $[n]$.

There is one useful operation on trees we will use, called *contraction*. The contraction of a tree (or any graph) by an edge e is obtained by identifying the endpoints of e and deleting e. Since we will be dealing with labeled trees, we will need to specify the label of the vertex obtained by identifying the endpoints of the contracted edge.

2. Descents and leaves in trees: first decomposition

We now define two sequences of polynomials. First, let $u_n(\alpha,\beta)$ be the polynomial in which the coefficient of $\alpha^a\beta^b$ is the number of trees in $\mathcal{T}_{0,[n]}$ with a descents and $b+1$ leaves. Next, let $t_n(\alpha,\beta)$ be the polynomial in which the coefficient of $\alpha^a\beta^b$ is the number of trees in $\mathcal{T}_{n,[n-1]}$ with a descents and b leaves. It is not hard to see that by these definitions, $\alpha\beta u_n=t_{n+1}$.

We first prove the following theorem.

Theorem 1. *For all* $n\geqslant 0$, $u_n(\alpha,\beta)$ *is symmetric in* α *and* β.

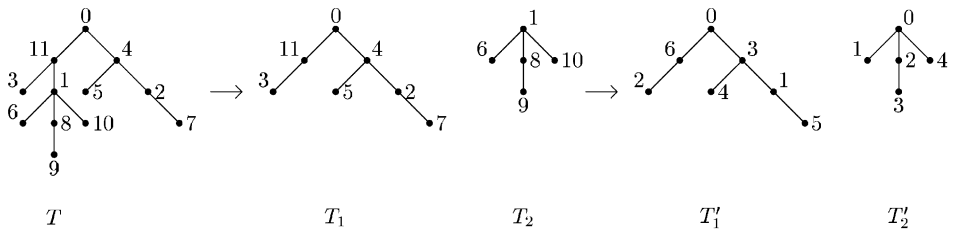

Fig. 2. First decomposition of trees.

This result was first shown by Gessel [4], who obtained a functional equation for the generating function $U(x;\alpha,\beta)=\sum_{n=1}^{\infty} u_n(\alpha,\beta)x^n/n!$,

$$1+U=(1+\alpha U)(1+\beta U)e^{x(1-\alpha-\beta-\alpha\beta U)}. \tag{1}$$

Gessel obtained this equation by considering marked forests, in which a set of vertices containing all of the descents and none of the leaves are marked, and using the exponential generating function for Eulerian polynomials. In this section and the next, we present two new proofs of Theorem 1, which use various decompositions of labeled trees to obtain recurrences for the u_n.

Our first proof that the u_n are symmetric uses a decomposition that involves forming two trees from a tree in $\mathscr{T}_{0,[n]}$ by deleting the edge e connecting 1 to its parent. The decomposition works as follows. Given a tree T, let $T-e$ be the graph obtained by removing the edge e from T. Then let T_1 be the component of $T-e$ containing the root of T (labeled 0), and let T_2 be the component of $T-e$ containing the vertex labeled 1. We root T_2 at 1. We can then take the reduction of the two trees T_1 and T_2. Fig. 2 shows a tree T, the trees T_1 and T_2, and the decomposition of T into reduced trees, T_1' and T_2'.

We can see that two reduced trees T_1', T_2', the content of T_1, and the parent of 1 in T specify T. In Fig. 2, the quadruple $(T_1',T_2',\{0,2,3,4,5,7,11\},11)$ specifies T. Note that vertex 11, the parent of vertex 1 in T, comes from a vertex in T_1' which is a descent.

To prove Theorem 1, we use the following lemma. In the statement of the lemma, the arguments of the polynomials $u_n(\alpha,\beta)$ are suppressed.

Lemma 1. *If* $n\geqslant 1$, *then*

$$u_n=u_{n-1}+\sum_{i=1}^{n-1}\binom{n-1}{i}u_{n-1-i}\left(\beta\frac{\partial}{\partial\alpha}\alpha u_i+\alpha\frac{\partial}{\partial\beta}\beta u_i+\alpha\beta\left((i+1)u_i-\frac{\partial}{\partial\beta}\alpha u_i-\frac{\partial}{\partial\beta}\beta u_i\right)\right). \tag{2}$$

Note that this recurrence is symmetric in α and β, as are the initial cases $u_0(\alpha,\beta)=u_1(\alpha,\beta)=1$. So if (2) holds, then the u_n must all be symmetric polynomials. Hence, once Lemma 1 is proven, Theorem 1 follows immediately.

Proof. Let T_1' be a tree in $\mathscr{T}_{0,[i]}$ and T_2' a tree in $\mathscr{T}_{0,[n-i-1]}$. By considering four cases of how we can obtain a tree $T \in \mathscr{T}_{0,[n]}$ which decomposes into T_1 and T_2, we can obtain a recurrence for u_n. In each case, to build T from smaller trees, we first choose an i-element subset A of $\{2,3,4,\ldots,n\}$ and form T_1 with reduction T_1' and content $A \cup \{0\}$. Then, we form T_2 with content $[n]-A$ and reduction T_2'; note that T_2 will be rooted at 1. To form a tree $T \in \mathscr{T}_{0,[n]}$, we can choose any vertex v of T_1 and make the root of T_2 a child of v. The number of descents and leaves of T depend upon the number of descents and leaves of T_1 and T_2 as well as the vertex of T_1 chosen to be the parent of the root of T_2.

In each case below, for $j=1$ or 2, a_j denotes the number of descents of T_j' (as well as of T_j) and b_j denotes the number of leaves of T_j' (as well as of T_j). Since any tree $T_1' \in T_{0,[i]}$ has i non-root vertices, we must choose i labels from $[n]-\{1\}$ to be the content of T_1; clearly, this choice can be made in $\binom{n-1}{i}$ ways. We also assume $n \geqslant 1$.

Case i: $i=0$. In this case, T_1 is a tree consisting only of a root labeled 0. When we attach the root of T_2 (labeled 1) to 0, we get a tree with a_2 descents and b_2 leaves. These trees are simply counted by u_{n-1}.

Case ii: $i>0$ *and we attach the root of* T_2 *to a descent of* T_1, *or to the root of* T_1. Here, the number of descents of the new tree will simply be a_1+a_2, and the number of leaves will be b_1+b_2. Since there are b_1 descents and one root in T_1', the contribution of this case to u_n is

$$\binom{n-1}{i} u_{n-1-i}\beta\left(\frac{\partial}{\partial\beta}\alpha u_i\right).$$

We need the extra factor of β since the coefficient of β^j in u_i counts trees with $j+1$ leaves.

Case iii: $i>0$ *and we attach the root of* T_2 *to a leaf of* T_1. In this case, the number of descents of the new tree will be a_1+a_2+1, since the parent of 1 in T is a leaf in T_1 and hence not a descent in T_1. The number of leaves of the new tree will be b_1+b_2-1. Thus, the contribution to u_n is

$$\binom{n-1}{i} u_{n-1-i}\alpha\left(\frac{\partial}{\partial\beta}\beta u_i\right).$$

Case iv: $i>0$ *and we attach the root of* T_2 *to a vertex of* T_1 *which is neither a leaf nor a descent nor the root*. The new tree in this case will have a_1+a_2+1 descents and b_1+b_2 leaves, so that the contribution to u_n is

$$\binom{n-1}{i}\alpha\beta u_{n-1-i}\left((i+1)u_i-\frac{\partial}{\partial\beta}\alpha u_i-\frac{\partial}{\partial\beta}\beta u_i\right).$$

We add up the contribution for each i from 0 to $n-1$ to get the recurrence stated in the lemma. □

Note that in $U(x;\alpha,\beta)=\sum_{n=1}^{\infty} u_n(\alpha,\beta)x^n/n!$, we do not include the constant term $u_0=1$. If we wish, we can rewrite (2) without u_0 by simply replacing it with 1. By

multiplying both sides of (2) by $x^{n-1}/(n-1)!$ and summing both sides from 1 to ∞, we can obtain the differential equation

$$\left(\frac{1}{U+1}\right)\frac{\partial}{\partial x}U = \beta\frac{\partial}{\partial x}(\alpha U) + \alpha\frac{\partial}{\partial \beta}(\beta U) + \alpha\beta\frac{\partial}{\partial x}(xU) - \alpha\beta\frac{\partial}{\partial \alpha}(\alpha U) - \alpha\beta\frac{\partial}{\partial \beta}(\beta U) + 1. \tag{3}$$

We will later use this differential equation to obtain Gessel's functional equation for U.

From the recurrence, it is easy to describe an involution Π_n on $\mathscr{T}_{0,[n]}$ which sends a tree with a descents and $b+1$ leaves to a tree with b descents and $a+1$ leaves. We define the involution recursively.

It is trivial to describe Π_0 and Π_1. We now assume Π_i has been defined for $0 \leqslant i < n$. For simplicity, if the content of a labeled tree T on $n+1$ vertices is some totally ordered set and T is rooted at its smallest vertex, we let $\Pi_n(T)$ be the tree whose reduction is $\Pi_n(T')$ (where T' is, as above, the reduction of T with respect to $[n]\cup\{0\}$) and whose content is the content of T. By an *increasing* vertex, we shall mean a non-root vertex which is neither a descent nor a leaf (that is, an increasing vertex is a vertex, neither a leaf nor the root, all of whose children are greater than it).

Consider a tree $T \in \mathscr{T}_{0,[n]}$ which decomposes into T_1 and T_2 (which are not reduced) as in Fig. 2. We denote the number of vertices of T_1 by i. We note whether the parent p of 1 in T becomes a descent, leaf, an increasing vertex, or the root in T_1. We then form $\Pi_n(T)$ by adding an edge from v in $\Pi_i(T_1)$ to 1 in $\Pi_{n-i-1}(T_2)$, where v is defined as follows. If p is the root of T_1, let v the least leaf in $\Pi_i(T_1)$. If p is the least leaf of T_1, let v be the root of $\Pi_i(T_1)$. In all other cases, suppose that the parent p is the kth greatest vertex of its type—descent, leaf, or increasing vertex—in T_1. Then let v be the vertex in $\Pi_i(T_1)$ which is, respectively, the kth greatest leaf, descent, or increasing vertex (that is, if p is a leaf, v is a descent; if p is a descent, v is a leaf; and if p is an increasing vertex, so is v).

It is not hard to show by induction that Π_n is an involution for any non-negative integer n. To obtain Theorem 1 from Π_n, we just consider how to obtain the number of descents and leaves of $\Pi_n(T)$ from T in each of the four cases in the proof of Lemma 1.

It turns out that Π_n is an extension of a well-known bijection Π_n^{I} which takes permutations to increasing trees (see, for example [15, p. 25]). We can consider a permutation $\pi = \pi(1)\pi(2)\ldots\pi(n)$ of $[n]$ to be a rooted tree with a single leaf in $\mathscr{T}_{0,[n]}$, such that the parent of $\pi(1)$ is 0 and the parent of $\pi(k)$ is $\pi(k-1)$. To map a permutation π to an increasing tree $\Pi_n^{\mathrm{I}}(\pi) \in \mathscr{T}_{0,[n]}$, we use the following procedure:

1. Set $k := 1$. Define $\pi(0) = 0$. Set the tree T to a lone root labeled 0.
2. While $k \leqslant n$, do the following:

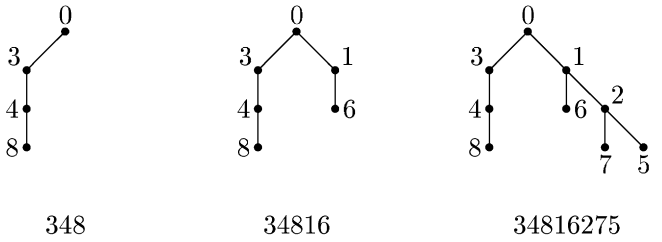

Fig. 3. The bijection between permutations and increasing trees.

Let j be the largest index such that $j<k$ and $\pi(j)<\pi(k)$.
Set $p:=\pi(j)$.
Make $\pi(k)$ a child of p in T.
Increase k by 1.

Fig. 3 shows three stages of the construction of the image of the permutation $\pi=34816275$ under Π_8^{I}.

It is not hard to prove by induction that Π_n restricted to rooted trees with one leaf is Π_n^{I}. First, note that for a given permutation π, any element appearing to the right of 1 will be a descendent of 1 in Π_n^{I}. Thus, if we write π as $\pi_1 1\pi_2$, we obtain $\Pi_n^{\mathrm{I}}(\pi)$ by first finding the increasing trees $\Pi_i^{\mathrm{I}}(\pi_1)$ and $\Pi_{n-1-i}^{\mathrm{I}}(\pi_2)$; then changing the label of the root of $\Pi_n^{\mathrm{I}}(\pi_2)$ from 0 to 1; and finally making 1 a child of the root of $\Pi_i^{\mathrm{I}}(\pi_1)$. In the example of Fig. 3, for $\pi=34816275$, we have $\pi_1=348$ and $\pi_2=6275$.

Next, if we consider the tree T with one leaf corresponding to π, note that if T is decomposed into two tress T_1 and T_2 as in Fig. 2, the parent of the vertex 1 in T becomes the sole leaf of T_1. Therefore, when we combine $\Pi_i(T_1)$ and $\Pi_{n-1-i}(T_2)$ to form $\Pi_n(T)$, the vertex 1 must become a child of the root of $\Pi_i(T_1)$. But by induction, $\Pi_i(T_1)=\Pi_i^{\mathrm{I}}(T_1)$ and $\Pi_{n-1-i}(T_2)=\Pi_{n-1-i}^{\mathrm{I}}(T_2)$. Thus, the procedure described for finding $\Pi_n^{\mathrm{I}}(\pi)$ exactly mimics the procedure for finding $\Pi_n(T)$.

3. Descents and leaves in trees: additional decompositions

The symmetry of the u_n can be proved using two other recurrences, in which switching α and β in one yields the other. Since the u_n satisfy both, this will show that the u_n are symmetric.

In the first additional decomposition, as before, we create two rooted trees from one rooted tree in $\mathscr{T}_{0,[n]}$. The only difference from the decomposition in Section 2 is that now we delete the edge connecting n to its parent. We then relabel n to be 0 and take the reduction of each resulting tree. Fig. 4 shows the decomposition of a tree using this procedure.

As we had for the first decomposition, the two reduced trees T_1' and T_2', the content of T_1, and the parent of n in T determine T. Using this second decomposition, we obtain the following lemma.

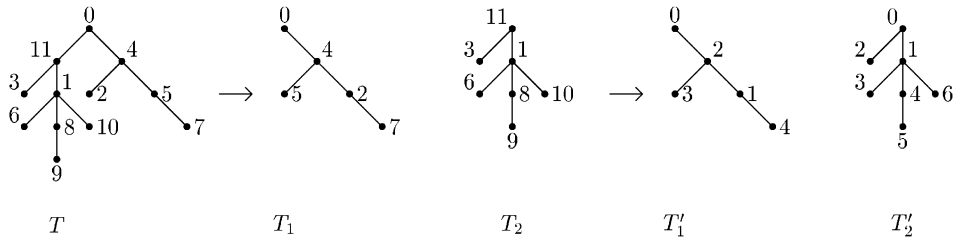

Fig. 4. Second decomposition of trees.

Lemma 2. *If* $n \geqslant 1$, *then*

$$u_n = \frac{\partial}{\partial\beta}\beta u_{n-1} + \beta\left(nu_{n-1} - \frac{\partial}{\partial\beta}\beta u_{n-1}\right) + \sum_{i=0}^{n-2}\binom{n-1}{i}\alpha u_{n-1-i}\left(\frac{\partial}{\partial\beta}\beta u_i + \beta(i+1)u_i - \beta\frac{\partial}{\partial\beta}\beta u_i\right). \tag{4}$$

Proof. We will need four cases to obtain the recurrence. As in the proof of Lemma 1, in each case we will have $T_1' \in \mathscr{T}_{0,[i]}$, where $0 \leqslant i \leqslant n-1$, and $T_2' \in \mathscr{T}_{0,[n-i-1]}$.

To form a tree with $n+1$ vertices, we first choose a subset A of i labels from $\{1,2,\ldots,n-1\}$. We then form tree T_1 with reduction T_1' and content $A \cup \{0\}$. To form T_2, we first take the tree with reduction T_2' and content $([n-1]-A)\cup\{0\}$, and then relabel the root of that tree with n. Finally, we connect T_1 to T_2 by making n the child of some vertex in T_1. We assume $n \geqslant 1$.

Case i: $0 \leqslant i \leqslant n-2$ *and vertex* n *is made a child of a leaf of* T_1. In this case, a leaf of T_1 is lost, and a descent is gained since the root of T_2, which is labeled n, must have at least one child. For each i, the contribution to u_n is

$$\binom{n-1}{i}\alpha u_{n-1-i}\frac{\partial}{\partial\beta}\beta u_i.$$

Case ii: $0 \leqslant i \leqslant n-2$ *and vertex* n *is made a child of a non-leaf of* T_1. In this case, a descent is gained since the root of T_2 must have at least one child. For each i, the contribution to u_n is

$$\binom{n-1}{i}\alpha\beta u_{n-1-i}\left((i+1)u_i - \frac{\partial}{\partial\beta}\beta u_i\right).$$

Case iii: $i = n-1$ *and vertex* n *is made a child of a leaf of* T_1. In this case, a leaf of T_1 is lost, and no descent is gained since the root of T_2 has no children. The contribution to u_n is

$$\frac{\partial}{\partial\beta}\beta u_{n-1}.$$

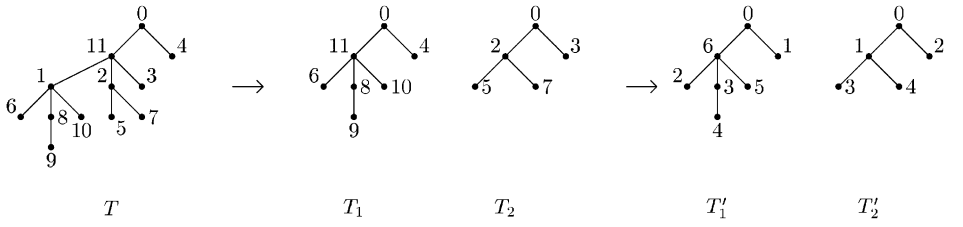

Fig. 5. Third decomposition of trees.

Case iv: $i = n-1$ *and vertex* n *is made a child of a non-leaf of* T_1. No leaves or descents are gained or lost. The contribution to u_n is

$$\beta\left(nu_{n-1} - \frac{\partial}{\partial\beta}\,\beta u_{n-1}\right).$$

Adding up all the cases on i from 0 to $n-1$ yields (4). □

From (4), we can derive the differential equation

$$\left(\frac{1}{\alpha U+1}\right)\frac{\partial}{\partial x}\,U = \frac{\partial}{\partial\beta}\,(\beta U) + \beta\,\frac{\partial}{\partial x}\,(xU) - \beta\,\frac{\partial}{\partial\beta}\,(\beta U) + 1. \tag{5}$$

Since (4) is not symmetric, it does not prove Theorem 1. However, we can derive another recurrence from a different decomposition of trees which turns out to be (4) with α and β swapped.

In this third decomposition, for $T \in \mathscr{T}_{0,[n]}$, we remove all subtrees of T rooted at siblings of 1 (if there are any), and we make the roots of these subtrees the children of a root labeled 0. This tree will be called T_2. From what remains of T, we then contract the edge connecting 1 to its parent, leaving the label of the parent on the vertex formed by identifying 1 and its parent. The remaining tree is T_1. Finally, we take the reductions of T_1 and T_2. Fig. 5 demonstrates this procedure.

From this decomposition, we obtain the following.

Lemma 3. *If* $n \geqslant 1$,

$$u_n = \frac{\partial}{\partial\alpha}\,\alpha u_{n-1} + \alpha\left(nu_{n-1} - \frac{\partial}{\partial\alpha}\,\alpha u_{n-1}\right) + \sum_{i=0}^{n-2}\binom{n-1}{i}\beta u_{n-1-i}\left(\frac{\partial}{\partial\alpha}\,\alpha u_i + \alpha(i+1)u_i - \alpha\,\frac{\partial}{\partial\alpha}\,\alpha u_i\right). \tag{6}$$

Proof. As in the derivation of the previous recurrences, we first choose $T_1' \in \mathscr{T}_{0,[i]}$ and $T_2' \in \mathscr{T}_{0,[n-i-1]}$, and then choose contents for each to form T_1 and T_2. To form a tree $T \in \mathscr{T}_{0,[n]}$ which decomposes into T_1 and T_2, we reverse the procedure shown in the

first part of Fig. 5. We first choose a vertex v of T_1. Then we remove the subtrees rooted at children of v. Next, we make 1 a child of v and make the former children of v into children of 1. Finally, we get rid of the root of T_2 and make the children of the root of T_2 into children of v.

There are four cases we consider. As in Lemma 2, for $j=1$ or 2, a_j denotes the number of descents of T_j and b_j denotes the number of leaves of T_j.

Before we look at the cases, we observe that 1 is a leaf in T if v is a leaf in T_1.

Case i: $0 \leqslant i \leqslant n-2$ *and* v *is a descent or the root of* T_1. The number of descents of T is $a_1 + a_2$, and the number of leaves of T is $b_1 + b_2$. The contribution to u_n is

$$\binom{n-1}{i} \beta u_{n-1-i} \frac{\partial}{\partial \alpha} \alpha u_i.$$

Case ii: $0 \leqslant i \leqslant n-2$ *and* v *is neither a descent nor the root of* T_1. In this case, v becomes a descent of T, so that T has $a_1 + a_2 + 1$ descents and $b_1 + b_2$ leaves. The contribution to u_n is

$$\binom{n-1}{i} \alpha\beta u_{n-1-i} \left((i+1)u_i - \frac{\partial}{\partial \alpha} \alpha u_i \right).$$

Case iii: $i = n-1$ *and* v *is a descent or the root of* T_1. Note that in this case all we do is make the children of v into children of 1 and make 1 the lone child of v. T has a_1 descents and b_1 leaves, so the contribution to u_n is

$$\frac{\partial}{\partial \alpha} \alpha u_{n-1}.$$

Case iv: $i = n-1$ *and* v *is neither a descent nor the root of* T_1. T has $a_1 + 1$ descents and b_1 leaves. The contribution to u_n is

$$\alpha \left(n u_{n-1} - \frac{\partial}{\partial \alpha} \alpha u_{n-1} \right).$$

Adding up the cases as i goes from 0 to $n-1$, and using the fact that $u_0 = 1$, we get the recurrence in (6). □

Note that (6) is simply (4) with α and β swapped. Thus, (4) and (6) provide another proof of Theorem 1.

As with the decomposition described in Section 2, we can use the decompositions in this section to describe bijections which transform trees with a descents and $b+1$ leaves to trees with b descents and $a+1$ leaves. The two bijections we can define are inverses of each other, but neither of them are involutions. We first describe one of these bijections, $\Psi_n : \mathcal{T}_{0,[n]} \to \mathcal{T}_{0,[n]}$, where n is a nonnegative integer. As before, for $n=0$ or 1, the mappings are trivial. We define the other mappings by induction.

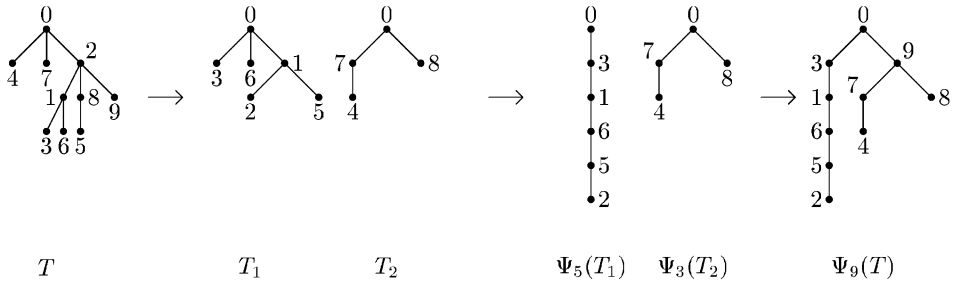

Fig. 6. Example of the bijection Ψ_9.

Given a tree $T \in \mathcal{T}_{0,[n]}$, we decompose it into two trees using the decomposition of Lemma 3 (as in Fig. 5). Then, we reduce the label of each non-root vertex in both trees by 1. Let $T_1 \in \mathcal{T}_{0,[i]}$ be the tree which contains the vertex formerly the parent of 1; let T_2 be the other tree. At this stage, we note whether the parent p of 1 in T becomes a descent, a non-descent, or the root in T_1; assume p is the kth greatest of its kind in T_1. We then take $\Psi_i(T_1)$ and $\Psi_{n-1-i}(T_2)$, and we relabel the root of the latter tree with n. Finally, if p was the root of T_1, let v be the least leaf of $\Psi_i(T_1)$; if p was the kth greatest descent or non-descent (not the root) in T_1, let v be, respectively, either the kth greatest leaf or non-leaf in $\Psi_i(T_1)$. We form $\Psi_n(T)$ by making the root of $\Psi_{n-1-i}(T_2)$, now labeled n, into a child of v in $\Psi_i(T_1)$.

Fig. 6 shows an example of how the bijection works on a tree $T \in \mathcal{T}_{0,[9]}$. Note that in the figure, the vertex 1 in T is attached to the vertex which becomes the fifth-greatest non-descent in T_1. Thus, in $\Psi_9(T)$, the vertex 9 is attached to the fifth-greatest non-leaf of $\Psi_i(T_1)$.

The second bijection, Ψ_n^{-1}, can be obtained by running the procedure for Ψ_n backwards. To find $\Psi_n^{-1}(T)$, we first decompose T into T_1 and T_2 using the decomposition from Lemma 2 (as in Fig. 4). Then we add 1 to the label of each non-root vertex of both T_1 and T_2. We note whether the parent p of n in T becomes a leaf or non-leaf in T_1. If p is the least leaf in T_1, let v be the root of $\Psi_i^{-1}(T_1)$. Otherwise, p is either the kth greatest leaf or kth greatest non-leaf in T_1 (for some k); let v be, respectively, either the kth greatest descent or non-descent of $\Psi_i^{-1}(T_1)$. We form $\Psi_n^{-1}(T)$ by removing the subtrees of the children of v in $\Psi_i^{-1}(T_1)$; making 1 a child of v and making the former children of v into children of 1; and finally, making the children of the root in $\Psi_{n-1-i}^{-1}(T_2)$ into the children of v.

We now derive (1), Gessel's functional equation. If in (6) we multiply both sides by $x^{n-1}/(n-1)!$ and sum both sides on n from 2 to ∞, we obtain the differential equation

$$\left(\frac{1}{\beta U+1}\right)\frac{\partial}{\partial x}U = \frac{\partial}{\partial \alpha}\alpha U + \alpha\frac{\partial}{\partial x}xU - \alpha\frac{\partial}{\partial \alpha}\alpha U + 1. \tag{7}$$

We can use (5) and (7) to eliminate the derivative with respect to α and β from (3), obtaining

$$\left(\frac{1}{U+1}\right)\frac{\partial}{\partial x}U = \left(\frac{\alpha}{\alpha U+1}\right)\frac{\partial}{\partial x}U + \left(\frac{\beta}{\beta U+1}\right)\frac{\partial}{\partial x}U - \alpha\beta\frac{\partial}{\partial x}xU - \alpha - \beta + 1.$$

If we integrate both sides of the last equation, using the condition $U=0$ when $x=0$, we obtain (1).

4. A generalization related to intransitive trees

Let $v_n(\alpha,\beta)$ count trees in $\mathscr{T}_{n,[n-1]}$ in which every vertex is either a descent or a leaf. In other words, v_n is the sum of the terms of total degree n in t_n; so in particular, v_n is homogeneous. Using some known bijections [9, p. 333; 13,14], it is not hard to show that the coefficient of $\alpha^i\beta^{n-i}$ in v_n also counts intransitive trees on $[n]$ (rooted at some vertex) in which i vertices are larger than their parent and children and $n-i$ vertices are smaller than their parent and children. Knuth [10] found that if $V=\sum_{n=2}^{\infty} v_n(\alpha,\beta)/n!$, we have

$$\beta + \left(\alpha\frac{\partial}{\partial\alpha} + \beta\frac{\partial}{\partial\beta}\right)V = \beta\exp\left(\alpha + \alpha\frac{\partial}{\partial\alpha}V\right). \tag{8}$$

This equation can be derived from (5), which was a consequence of Lemma 2. To do this, recall $t_n(\alpha,\beta)=\alpha\beta u_{n-1}(\alpha,\beta)$, and let $T(x;\alpha,\beta)=\sum_{n=2}^{\infty} t_n x^n/n!$. Since t_n counts trees in $\mathscr{T}_{n,[n-1]}$ by descents and leaves, we have

$$x^n t_n\left(\frac{\alpha}{z},\frac{\beta}{x}\right)\Big|_{x=0} = v_n(\alpha,\beta), \quad T\left(x;\frac{\alpha}{x},\frac{\beta}{x}\right)\Big|_{x=0} = V. \tag{9}$$

Moreover, by the definition of the t_n, we have

$$\alpha\beta\int_0^x U(y;\alpha,\beta)\,\mathrm{d}y = T(x;\alpha,\beta). \tag{10}$$

Integrating both sides of (5) and multiplying by β, we obtain

$$\beta + \frac{\partial}{\partial x}T = \beta\exp\left(\frac{\partial}{\partial\beta}T + \left(x\frac{\partial}{\partial x}T - \beta\frac{\partial}{\partial\beta}T\right) + x\alpha\right). \tag{11}$$

This equation generalizes (8), and we can easily obtain (8) from (11). We first multiply both sides of (11) by x. We then replace α with α/x and β with β/x. Finally, we set $x=0$. Since v_n is homogeneous of degree n, $\alpha(\partial/\partial\alpha)v_n + \beta(\partial/\partial\beta)v_n = nv_n$, so that $nV - \beta(\partial/\partial\beta)V = \alpha(\partial/\partial\alpha)V$.

We can also explain (11) combinatorially using a decomposition of trees. The coefficient of $x^n/n!$ in $(\partial/\partial x)T$ counts rooted trees in $\mathscr{T}_{\infty,[n]}$. On the right-hand side, the

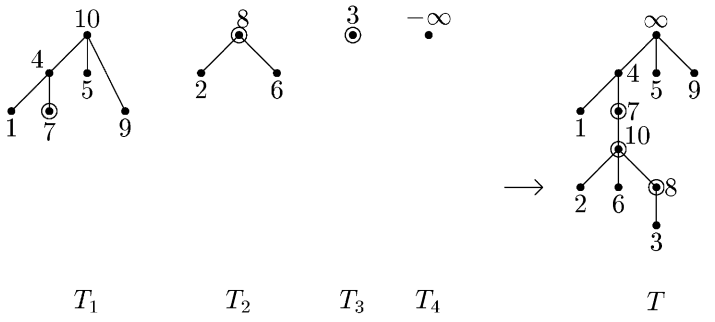

Fig. 7. Interpretation of (11).

term $(\partial/\partial\beta)T$ counts trees in $\mathscr{T}_{n,[n-1]}$ which are additionally rooted at a leaf; the term $x(\partial/\partial x)T-\beta(\partial/\partial\beta)T$ counts trees in $\mathscr{T}_{n,[n-1]}$ additionally rooted at a non-leaf (perhaps the root); and αx simply counts a lone vertex, which we can consider to be rooted tree additionally rooted at its original root. To form an object counted by the left side, we take a set of trees $T_1, T_2, T_3, \ldots, T_{j-1}$, whose reductions are counted by the exponent on the right and the set of whose contents is a partition of $[n]$. We add to these trees a tree T_j consisting of a lone root labeled $-\infty$; this last tree corresponds to the extra factor of β on the right side of (11). Note that the greatest vertex in each tree is the (original) root; we assume that if $1\leqslant i\leqslant j-1$, the root of T_{j+1} is greater than the root of T_j. Next, for $1\leqslant i\leqslant j-1$, we relabel the root of T_{i+1} with the root of T_i, and we relabel the root of T_1 with ∞. Finally, for each i from 1 to $j-1$, we attach the root of T_{i+1} to the additional root of T_i. We now have an object counted by the left side of (11). Fig. 7 gives an example of this procedure. In the figure, the additional roots of the T_i are marked with a circle, and the vertices which the additional roots give rise to in T are also marked with a circle. Note that because of the relabeling procedure, some of the circled vertices of T are labeled with a different set of elements from the circled vertices of the T_i.

To run this bijection the other way, given a tree T, identify the largest non-root vertex v_1 in the tree (the vertex 10 in T in Fig. 7). Eliminate the edge connecting v_1 to its parent, relabel the vertex which was the root of T by v_1, and relabel the vertex originally labeled v_1 by ∞. Take component of the remaining graph containing the former root of T, now labeled v_1, and give it an additional root at the vertex which was the parent of v_1 in T; this component will be T_1. Let C_1 be the other component of what remained of T. We now perform the same procedure on C_1 as we did on T originally to get T_2 and another component C_2. We continue recursively until, for some j, C_j is a lone vertex. We relabel this vertex $-\infty$, and we are done.

5. A symmetry in parking functions

Recurrence (4) has a simple interpretation in terms of parking functions.

Theorem 2. *The coefficient of $\alpha^a\beta^b$ in $u_n(\alpha,\beta)$ is the number of parking functions on $[n]$ with a descents and b unfavorable spaces.*

Proof. We exhibit a bijection from trees in $\mathscr{T}_{0,[n]}$ to the parking functions on $[n]$ which sends a tree with a descents and $b+1$ leaves to a parking function with a descents and b unfavorable spaces. We note that this bijection was described in a different form by Françon [2]. We present the bijection and then show that it preserves the requisite statistics.

The map from parking functions to trees works as follows. Let $\tau(p)$ be the output permutation of a parking function p on $[n]$. The image of p is simply the tree in $\mathscr{T}_{0,[n]}$ such that the children of the root 0 are the elements of $p^{-1}(1)$ (i.e., the cars which prefer the first space) and the children of vertex $v\in[n]$ are the elements of $p^{-1}(\tau(p)^{-1}(v)+1)$. In other words, to determining the children of vertex v, find the space in which car v parks; then the children of v are the numbers of the cars which prefer the next space. This of course implies that car $\tau(p)(n)$ which parks in the last available space is a leaf in the tree.

It is not hard to see that this map is well-defined. The procedure produces a connected graph, since new vertices are added so that they are adjacent to vertices already in the same component as the root. Since only n edges are drawn among $n+1$ vertices, the connected graph must be a tree.

For a tree T in $\mathscr{T}_{0,[n]}$, the inverse function works as follows.

1. Set $i:=1$ and $S=\{0\}$.
2. While $i\leqslant n$, do the following:
   Set $v(i):=$ minimum element of S.
   Set $p^{-1}(i):=\{\text{children of } v(i) \text{ in } T\}$.
   Set $S:=(S\backslash\{v(i)\})\cup\{\text{children of } v(i) \text{ in } T\}$.
   Increase i by 1.

For the tree T in Fig. 2, we have the following series of operations to produce a parking function using the algorithm for the inverse function.

i	$v(i)$	$p^{-1}(i)$	S
1	0	$\{4,11\}$	$\{4,11\}$
2	4	$\{2,5\}$	$\{2,5,11\}$
3	2	$\{7\}$	$\{5,7,11\}$
4	5	$\emptyset$	$\{7,11\}$
5	7	$\emptyset$	$\{11\}$
6	11	$\{1,3\}$	$\{1,3\}$
7	1	$\{6,8,10\}$	$\{3,6,8,10\}$
8	3	$\emptyset$	$\{6,8,10\}$
9	6	$\emptyset$	$\{8,10\}$
10	8	$\{9\}$	$\{9,10\}$
11	9	$\emptyset$	$\{10\}$

To show this is a well-defined map, we need only show that this produces a function for which $|\bigcup_{j=1}^{i} p^{-1}(j)| \geqslant i$ for each $i \in [n]$. After $p^{-1}(1), p^{-1}(2), \ldots, p^{-1}(m)$ have been assigned, in order to choose $v(m+1)$, the algorithm requires that $|\bigcup_{j=1}^{i} p^{-1}(j)| \geqslant i$. But this must be true, for if not, then the vertices in $\bigcup_{j=1}^{i} p^{-1}(j)$ are adjacent to no vertices in T outside of that set, contradicting the fact we started with a tree.

If we let $v(n+1)$ be the sole element left in S when the algorithm terminates, we claim that the permutation $v(2)v(3)\cdots v(n+1)$ is in fact the permutation of parked cars arising from the parking function which is associated to the tree. This results from the fact that, as cars park according to a parking function, once the first $i-1$ spaces are filled, then the car in the ith space is the smallest-numbered car preferring one of the first i space which has not parked in one of the first $i-1$ spaces.

Since $v(2)v(3)\cdots v(n+1)$ is the permutation of parked cars of the parking function p, it is clear that if we find the tree corresponding to the parking function p, we will get back our original tree. Similarly, it is easy to see that starting with a parking function, applying the second function after the first yields the original parking function. Hence, the maps are mutually inverse.

We now show that the first bijection maps a parking function p with a descents to a tree T with a descents. As above, let $v(2)v(3)\cdots v(n+1)$ be the permutation obtained in running the algorithm for the second bijection to get p from T. Suppose there is a descent in the output permutation $\tau(p) = v(2)v(3)\cdots v(n+1)$, say $v(i) > v(i+1)$. But this can happen if and only if $v(i+1)$ is a child of $v(i)$ in T, since according to the algorithm for the second bijection, $v(i)$ was the smallest element of S at step i. So $v(i)$ is a descent of T, and we conclude the bijection preserves descents.

It is immediate that the first bijection maps a parking function with b unfavorable spaces to a tree with $b+1$ leaves. This is because no car prefers space i if and only if vertex $\tau(p)(i-1)$ is a leaf, and $\tau(p)(n)$ is always a leaf.

This establishes the theorem. □

We note that we could also prove the theorem by obtaining a recurrence for polynomials which count parking functions on $[n]$ by descents and unfavorable spaces. To derive the recurrence, we form a parking function p on n from a parking function $p_1' \in P_i$ and a parking function $p_2' \in P_{n-1-i}$ $(0 \leqslant i \leqslant n-1)$ as follows. Let $A = \{a_1, a_2, \ldots, a_i\}$ be an i-element subset of $[n-1]$. Then let p_1 be the parking function with content A and reduction p_1', and let p_2 be the parking function with content $[n-1] - A = \{b_1, b_2, \ldots, b_{n-1-i}\}$ and reduction p_2'. We define a parking function $p \in P_n$ by letting $p(n)$ be any element of $[i+1]$ and setting

$$p(a_j) = p_1(a_j), \quad j \in [i],$$

$$p(b_j) = p_2(b_j) + i + 1, \quad j \in [n-1-i].$$

Fig. 8 demonstrates how this works. In the figure, if car i likes space j (i.e., $p(i) = j$) in a parking function, i is written in space j of the diagram. Additionally, for a parking function p, the notation $p = (j_1, j_2, \ldots, j_n)$ means $p(i) = j_i$. In the middle step, we see that $A = \{1, 3, 5\}$. In the last step, car 8 is assigned space 2 (i.e., $p(8) = 2$), and the two sequences of spaces are joined by putting a new empty space between them.

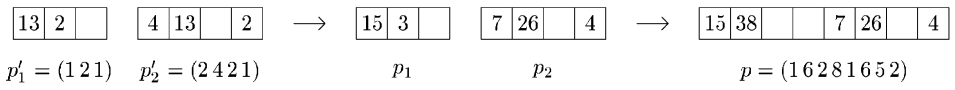

Fig. 8. Composition of parking functions.

To recover the original two parking functions, it is simplest to think of the parking function in the form of numbers written in boxes (as in Fig. 8). Then, for a parking function on n cars, let $i+1$ be the space car n *parks in* (often not equal to $p(n)$). If we simply erase car n from the diagram and remove space $i+1$, we will end up with two parking functions (one possibly the "empty" function). This follows because no car except car n will attempt to park in the space car n parks in; see [7, Section 5] for more on this property of parking functions. If we take the reduction of the parking functions, we end up with the original two.

Using this technique, we can show that the polynomials $w_n(\alpha,\beta)$ counting parking functions on $[n]$ by descents and leaves satisfy the recurrence in (4) also satisfied by the u_n. Since $w_0 = u_0$ and $w_1 = u_1$, we can then conclude that $w_n = u_n$ for all n. The method of proof is very similar to that for Lemma 2. The statistics depend on what space we assign car n in p.

6. Generalization to binary trees

A *binary tree* is a rooted tree in which each vertex has either zero, one, or two children, and the children of each vertex are ordered. Each non-root vertex of a binary tree is either the *left child* or the *right child* of its parent. A vertex without a left (respectively, right) child is called a *left* (*right*) *leaf*. The subtree rooted at the left (right) child of a vertex v is called the *left* (*right*) *subtree* of v. We will focus on binary trees with n vertices which are labeled by elements of $[n]$; we will also consider binary trees labeled with n positive integers, whose reduction we take with respect to $[n]$.

In Fig. 9 is an example of a binary tree, whose root is labeled 5. Its left children are 1, 3, and 4; its right children are 2 and 6. Its left leaves are 3, 4, and 6; its right leaves are 2,3,4 and 6.

A vertex of a binary tree is a *left* (*right*) *ascent* if it is a left (right) child and it is greater than its parent. We similarly define *left* (*right*) *descents* of binary trees. Note that for every binary tree on n vertices, the sum of the numbers of left descents, right descents, left ascents, and right ascents is $n-1$. The left ascents of the binary tree in Fig. 9 are 3 and 4; the only right descent is 2.

Let $f_n(\alpha,\beta,\gamma,\delta)$ be the polynomial in which the coefficient of $\alpha^a\beta^b\gamma^c\delta^d$ is the number of binary trees having a left ascents, b left descents, c right ascents, and d right descents. Note that f_n is homogeneous, since $a+b+c+d$ must equal $n-1$. For convenience, we let $f_0 = 1$.

Some symmetries of f_n are obvious. In a binary tree with n vertices, we can replace each label i with $n+1-i$, swapping descents and ascents. We could also flip the

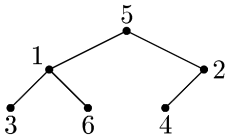

Fig. 9. Example of a binary tree.

binary tree about its root, so that right objects are swapped with left objects. In terms of f_n, we have

$$f_n(\alpha,\beta,\gamma,\delta) = f_n(\gamma,\delta,\alpha,\beta) \text{ (the symmetry between left and right)},$$
$$f_n(\alpha,\beta,\gamma,\delta) = f_n(\beta,\alpha,\delta,\gamma) \text{ (the symmetry between ascents and descents)}. \quad (12)$$

Our focus is the following theorem, first shown by Gessel [5], which implies that the number of binary trees with a left ascents, b left descents, c right ascents, and d right descents equals the number of binary trees with a left ascents, c left descents, b right ascents, and d right descents. (That is, we can swap left descents and right ascents but leave the number of left ascents and right descents the same.)

Theorem 3. *For all* $n \geqslant 0$, $f_n(\alpha,\beta,\gamma,\delta) = f_n(\alpha,\gamma,\beta,\delta)$.

We note that by (12), once Theorem 3 is established, we also have $f_n(\alpha,\beta,\gamma,\delta) = f_n(\delta,\beta,\gamma,\alpha)$.

Gessel has shown this by establishing that for

$$F(x;\alpha,\beta,\gamma,\delta) := \sum_{n=1}^{\infty} f_n(\alpha,\beta,\gamma,\delta)\frac{x^n}{n!}, \quad (13)$$

we have

$$\frac{(\alpha F+1)(\delta F+1)}{(\beta F+1)(\gamma F+1)} = \exp[((\alpha\delta-\beta\gamma)F+\alpha+\delta-\beta-\gamma)x].$$

We show it below using several decompositions of binary trees, closely related to each other, which are similar to those employed in previous sections of this chapter. We can also obtain Gessel's functional equation from them. We will conclude this section by noting how to use the results involving binary trees to obtain our results for rooted trees.

The first decomposition of binary trees works as follows. We start with a binary tree B on $[n]$. We remove the left subtree of 1 and call it B_2; it may be empty. Next, working with what remains of B, we contract the edge connecting 1 to its right subtree and label the vertex obtained by identifying 1 and its right child c with c. If 1 has no right child, we simply delete 1. This tree is B_1. To complete the decomposition, we can take the reductions B_1' and B_2' of B_1 and B_2. Fig. 10 depicts how we can decompose the binary tree in Fig. 9 into two reduced binary trees.

We observe that if 1 is not the root of a binary tree B, then B is determined by the two reduced trees B_1' and B_2', the content of B_1, the parent of 1, and whether 1 is a

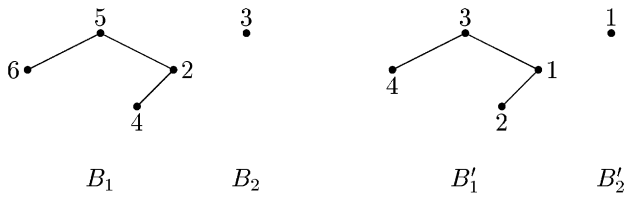

Fig. 10. Decomposition of binary tree in Fig. 9.

left or right child in B. If 1 is the root, then B is determined by B_1' and B_2' and the content of B_1.

We now show the following recurrence.

Lemma 4. *If* $n \geqslant 2$, *then*

$$\begin{aligned} f_n = & \left[\sum_{i=1}^{n-2} \binom{n-1}{i} f_{n-1-i} \left(\alpha\beta\left(f_i + \gamma\frac{\partial}{\partial\gamma} f_i + \delta\frac{\partial}{\partial\delta} f_i\right) + \beta\gamma\left(\alpha\frac{\partial}{\partial\alpha} f_i\right)\right.\right. \\ & + \alpha\gamma\left(\beta\frac{\partial}{\partial\beta} f_i\right) + \alpha\delta\left(f_i + \alpha\frac{\partial}{\partial\alpha} f_i + \beta\frac{\partial}{\partial\beta} f_i\right) \\ & \left.\left. + \alpha\delta\left(\gamma\frac{\partial}{\partial\gamma} f_i\right) + \alpha\gamma\left(\delta\frac{\partial}{\partial\delta} f_i\right) + \alpha\gamma f_i\right)\right] \\ & + \alpha f_{n-1} \\ & + \left[\beta\left(f_{n-1} + \gamma\frac{\partial}{\partial\gamma} f_{n-1} + \delta\frac{\partial}{\partial\delta} f_{n-1}\right) + \beta\gamma\left(\frac{\partial}{\partial\alpha} f_{n-1}\right)\right. \\ & + \gamma\left(\beta\frac{\partial}{\partial\beta} f_{n-1}\right) + \delta\left(f_{n-1} + \alpha\frac{\partial}{\partial\alpha} f_{n-1} + \beta\frac{\partial}{\partial\beta} f_{n-1}\right) \\ & \left. + \delta\left(\gamma\frac{\partial}{\partial\gamma} f_{n-1}\right) + \gamma\left(\delta\frac{\partial}{\partial\delta} f_{n-1}\right) + \gamma f_{n-1}\right]. \end{aligned} \tag{14}$$

Proof. Throughout the proof, for $j = 1$ or 2, B_j' will have a_j left ascents, b_j left descents, c_j right ascents, and d_j right descents. Also, B_1' will have i vertices and B_2' will have $n-1-i$ vertices, so that for instance $i = a_1 + b_1 + c_1 + d_1 + 1$.

We will get a recurrence for the f_n, as done previously for the u_n in Sections 2 and 3, by running the decomposition in Fig. 10 backwards to build a binary tree B on $[n]$. We first choose binary trees B_1' and B_2', and then we choose contents for them so that the union of their contents is $\{2,3,\ldots,n\}$. We subsequently choose a subtree C which is either all of B_1 or the left or right subtree of a vertex v of B_1. C could be empty. Once C is chosen, we remove it from B_1 and replace it with the single vertex 1. Thus, if C is all of B_1, 1 becomes the root of the tree we are making; and if C is the left

Table 1
Statistics for B built from B_1 and B_2 if $1 \leqslant i \leqslant n-2$, first decomposition

(a) Cases for which C is not empty

Root of C in B_1	Multiplicity in B_1	Left ascents of B	Left descents of B	Right ascents of B	Right descents of B
Left ascent	a_1	$a_1 + a_2$	$b_1 + b_2 + 1$	$c_1 + c_2 + 1$	$d_1 + d_2$
Left descent	b_1	$a_1 + a_2 + 1$	$b_1 + b_2$	$c_1 + c_2 + 1$	$d_1 + d_2$
Right ascent	c_1	$a_1 + a_2 + 1$	$b_1 + b_2$	$c_1 + c_2$	$d_1 + d_2 + 1$
Right descent	d_1	$a_1 + a_2 + 1$	$b_1 + b_2$	$c_1 + c_2 + 1$	$d_1 + d_2$
Root	1	$a_1 + a_2 + 1$	$b_1 + b_2$	$c_1 + c_2 + 1$	$d_1 + d_2$

(b) Cases for which C is empty

C is an empty	Multiplicity in B_1	Left ascents of B	Left descents of B	Right ascents of B	Right descents of B
Left subtree	$c_1 + d_1 + 1$	$a_1 + a_2 + 1$	$b_1 + b_2 + 1$	$c_1 + c_2$	$d_1 + d_2$
Right subtree	$a_1 + b_1 + 1$	$a_1 + a_2 + 1$	$b_1 + b_2$	$c_1 + c_2$	$d_1 + d_2 + 1$

(respecitvely, right) subtree of v in B_1, 1 becomes the left (respecitvely, right) child of v. Finally, in any case, we make C the right subtree of 1 and B_2 the left subtree of 1.

Using the above procedure, we now count the statistics for B using the statistics for B_1' and B_2' (or identically B_1 and B_2). We consider three cases: $1 \leqslant i \leqslant n-2$, $i = n-1$, and $i = 0$. The statistics for B depend on what kind of vertex the root of C is in B_1, or if C is empty, on whether C is a left or right emtpy subtree of B_1.

Case i: $1 \leqslant i \leqslant n-2$. In Table 1 are the ways we can build a binary tree B from C and B_1 and how the statistics for B depend on those for C and B_1. Table 1(a) consists of cases for which C is not empty. The first column of Table 1(a) lists the possible characterizations of the root of C when it is part of B_1. Table 1(b) consists of cases for which C is empty, that is, when v is a left or a right leaf. There are two possibilities, namely, that C is the empty left subtree of v or the empty right subtree of v. In both parts of Table 1, the next four columns in the table count the four statistics for the binary tree B built from B_1 and B_2 using the procedure given above. "Multiplicity" refers to how many ways we can choose a subtree C of B_1 corresponding to the given case. For each case in Table 1(a), the multiplicity is the number of vertices in B_1 which are of the specified kind for the root of C; in Table 1(b), the mulitplicity is the number of left or right leaves of C, depending on whether C is a left or right empty subtree. Note that C can be the left or right subtree of any vertex of B_1, or it can be all of B_1. Thus, there are $2i + 1$ choices for C.

We recall that for each pair B_1', B_2' of reduced binary trees, there are $\binom{n-1}{i}$ ways to choose the content of B_1 and B_2. Thus, for example, from the table we see that the contribution to f_n for each i in the case where C is the empty left

subtree of v is

$$\binom{n-1}{i} f_{n-1-i}\left(\alpha\beta\left(f_i+\gamma\frac{\partial}{\partial\gamma}f_i+\gamma\frac{\partial}{\partial\delta}f_i\right)\right).$$

The contribution to f_n for each i when the root of C is a left ascent is

$$\binom{n-1}{i} f_{n-1-i}\left(\beta\gamma\left(\alpha\frac{\partial}{\partial\alpha}f_i\right)\right),$$

and so on. By adding the contributions for each of the seven cases in Table 1 and sum on i from 1 to $n-2$, we get the summation on right-hand side of (14).

Case ii: $i=0$. In this case, B_1' is empty, and there is only one possibility: 1 becomes the root of B with B_2 the left subtree of 1. We obtain a binary tree with a_2+1 left ascents, b_2 left descents, c_2 right ascents, and d_2 right descents. The contribution to f_n is αf_{n-1}.

Case iii: $i=n-1$. In this case, B_2 is empty. We have the same seven possibilities for C as in Table 1. The table of results is also the same as Table 1 except we subtract one from each value in the column counting left ascents. This results from the lack of a left child for 1 in B. The contribution to f_n is given in the last three lines of (14).

If we add the resulting terms from all the cases together, we obtain the recurrence in (14). □

Since $f_0=1, f_1=1$, and the above recurrence is symmetric in β and γ, Lemma 4 establishes Theorem 3.

We can multiply both sides of (14) by $x^{n-1}/(n-1)!$ and sum on n from 2 to ∞ to obtain a differential equation for F. With a little work, we can obtain

$$\begin{aligned}\left(\frac{\alpha}{\alpha F+1}\right)\frac{\partial}{\partial x}F &= \alpha\beta F+\alpha\beta\gamma\frac{\partial}{\partial\gamma}F+\alpha\beta\delta\frac{\partial}{\partial\delta}F\\ &\quad+\alpha\beta\gamma\frac{\partial}{\partial a}F+\alpha\beta\gamma\frac{\partial}{\partial\beta}F+\alpha\delta F+\alpha^2\delta\frac{\partial}{\partial\alpha}F\\ &\quad+\alpha\beta\delta\frac{\partial}{\partial\beta}F+\alpha\gamma\delta\frac{\partial}{\partial\gamma}F+\alpha\gamma\delta\frac{\partial}{\partial\delta}F+\alpha\gamma F+\alpha. \qquad (15)\end{aligned}$$

From (14) and (12), we obtain that $f_n(\alpha,\beta,\gamma,\delta)=f_n(\delta,\beta,\gamma,\alpha)$. We in fact can prove this directly by another recurrence for the f_n which is symmetric in α and δ. This recurrence is obtained by modifying the decomposition of binary trees. This second decomposition is the same as the first, except the vertex we focus on is n, not 1. In this case, starting with a binary tree B on $[n]$, we first let B_2 be the left subtree of the vertex n. To get B_1 from B, we contract the edge connecting n to its right child, leaving the label of the right child on the vertex formed by identifying n and its right child.

The analysis of the second decomposition is very similar to that used in Lemma 4. When building B, C is the subtree of B_1 which becomes the right subtree of n in

Table 2
Statistics for B built from B_1 and B_2 if $1 \leqslant i \leqslant n-2$, second decomposition

(a) Cases for which C is not empty

Root of C in B_1	Multiplicity in B_1	Left ascents of B	Left descents of B	Right ascents of B	Right descents of B
Left ascent	a_1	a_1+a_2	b_1+b_2+1	c_1+c_2	d_1+d_2+1
Left descent	b_1	a_1+a_2+1	b_1+b_2	c_1+c_2	d_1+d_2+1
Right ascent	c_1	a_1+a_2	b_1+b_2+1	c_1+c_2	d_1+d_2+1
Right descent	d_1	a_1+a_2	b_1+b_2+1	c_1+c_2+1	d_1+d_2
Root	1	a_1+a_2	b_1+b_2+1	c_1+c_2	d_1+d_2

(b) Cases for which C is empty

C is an empty	Multiplicity in B_1	Left ascents of B	Left descents of B	Right ascents of B	Right descents of B
Left subtree	c_1+d_1+1	a_1+a_2+1	b_1+b_2+1	c_1+c_2	d_1+d_2
Right subtree	a_1+b_1+1	a_1+a_2	b_1+b_2+1	c_1+c_2+1	d_1+d_2

B. For the case where the number of vertices of B_1 is between 1 and $n-2$, Table 2 summarizes how the statistics for B depending on what the root of C is in B_1 or, if C is empty, whether C is a left or right empty subtree.

The case where B_1 is empty contributes βf_{n-1} to the recurrence for f_n, and if B_1 has $n-1$ veritices, we again have the seven cases in Table 2, but we subtract a left descent from the resulting tree B for each case. The differential equation we can obtain from the resulting recurrence, which is symmetric in α and δ, is

$$\left(\frac{\beta}{\beta F+1}\right)\frac{\partial}{\partial x}F = \alpha\beta F + \alpha\beta\gamma\frac{\partial}{\partial\gamma}F + \alpha\beta\delta\frac{\partial}{\partial\delta}F + \alpha\beta\delta\frac{\partial}{\partial\alpha}F + \alpha\beta\delta\frac{\partial}{\partial\beta}F + \beta\gamma F + \alpha\beta\gamma\frac{\partial}{\partial\alpha}F + \beta^2\gamma\frac{\partial}{\partial\beta}F + \beta\gamma\delta\frac{\partial}{\partial\gamma}F + \beta\gamma\delta\frac{\partial}{\partial\delta}F + \beta\delta F + \beta. \tag{16}$$

As we did for U in Section 3, we can obtain the functional equation (13) for F that was first derived by Gessel. To do so, we consider two more decompositions of binary trees, then derive differential equations for F, and finally combine the results from all four decompositions. The further two methods of decomposition which we need vary in only one way from the previous two: In these cases, starting with a binary tree B, B_2 is the *right* subtree of 1 (or n), and B_1 is obtained by contracting the edge connecting 1 (or n) to its left child (and keeping the label of the child), if one exists, or by deleting 1 (or n) if it is a left leaf. By again using an analysis like that in Lemma 4,

we can derive the differential equations

$$\left(\frac{\delta}{\delta F+1}\right)\frac{\partial}{\partial x}F=\beta\delta F+\beta\gamma\delta\frac{\partial}{\partial\gamma}F+\alpha\beta\delta\frac{\partial}{\partial\alpha}F$$
$$+\beta\gamma\delta\frac{\partial}{\partial\delta}F+\beta\gamma\delta\frac{\partial}{\partial\beta}F+\alpha\delta F+\delta^2\alpha\frac{\partial}{\partial\delta}F$$
$$+\alpha\beta\delta\frac{\partial}{\partial\beta}F+\alpha\gamma\delta\frac{\partial}{\partial\gamma}F+\alpha\gamma\delta\frac{\partial}{\partial\alpha}F+\gamma\delta F+\delta, \tag{17}$$

$$\left(\frac{\gamma}{\gamma F+1}\right)\frac{\partial}{\partial x}F=\alpha\gamma F+\alpha\beta\gamma\frac{\partial}{\partial\beta}F+\alpha\gamma\delta\frac{\partial}{\partial\delta}F$$
$$+\alpha\gamma\delta\frac{\partial}{\partial\alpha}F+\alpha\gamma\delta\frac{\partial}{\partial\gamma}F+\beta\gamma F+\alpha\beta\gamma\frac{\partial}{\partial\alpha}F$$
$$+\beta\gamma^2\frac{\partial}{\partial\gamma}F+\beta\gamma\delta\frac{\partial}{\partial\beta}F+\beta\gamma\delta\frac{\partial}{\partial\delta}F+\gamma\delta F+\gamma. \tag{18}$$

Note that in addition, (17) can be obtain by swapping α and δ in (15), and (18) can be obtained by swapping β and γ in (16).

In order to get a functional equation for F, observe that

$$x\frac{\partial}{\partial x}F=\alpha\frac{\partial}{\partial\alpha}F+\beta\frac{\partial}{\partial\beta}F+\gamma\frac{\partial}{\partial\gamma}F+\delta\frac{\partial}{\partial\delta}F+F, \tag{19}$$

since the sum of the left ascents, left descents, right ascents, and right descents of a binary tree with n vertices is $n-1$. We now add corresponding sides of (15) and (17) and, from that equation, we subtract corresponding sides of (16) and (18). We then apply (19) to obtain

$$\frac{\alpha}{\alpha F+1}\frac{\partial}{\partial x}F+\frac{\delta}{\delta F+1}\frac{\partial}{\partial x}F-\frac{\beta}{\beta F+1}\frac{\partial}{\partial x}F-\frac{\gamma}{\gamma F+1}\frac{\partial}{\partial x}F$$
$$=(\alpha\delta-\beta\gamma)\frac{\partial}{\partial x}(xF)+\alpha+\delta-\beta-\gamma.$$

Integrating both sides, and using the condition that $F=0$ when $x=0$, we get the desired functional equation

$$\frac{(\alpha F+1)(\delta F+1)}{(\beta F+1)(\gamma F+1)}=\exp[((\alpha\delta-\beta\gamma)F+\alpha+\delta-\beta-\gamma)x].$$

We remark that we can actually obtain our results involving the symmetry between descents and leaves in rooted trees, in (3), (5) and (7), from (15), (16), and (18), respectively. To make the connection, we can use a well-known bijection (see, for example, [9, p. 333] or [13]) which takes binary trees with no right descents to trees in $\mathscr{T}_{0,[n]}$. In this bijection, a binary tree with a left descents and b right ascents is mapped to a tree with a descents and $b+1$ leaves. Therefore, to transform a differential

equation for binary trees to one for rooted trees, we first remove any $(\partial/\partial\alpha)F$ term by applying (19), because in our correspondence left ascents do not correspond to a statistic we count for rooted trees. Then we set $\alpha = 1$ and $\delta = 0$. Finally, if we want the resulting equations to look the same as the ones we saw in Sections 2 and 3, we replace β with α (since α counts descents in the equations for rooted trees) and then γ with β (since β counts leaves in the equations for rooted trees).

Acknowledgements

This work is based on work from the author's Ph.D. thesis, written under the supervision of Ira Gessel. The author thanks Ira Gessel for suggesting study of this symmetry and for his help and encouragement. The author also thanks the referees for some helpful suggestions.

References

[1] D. Foata, J. Riordan, Mappings of acyclic and parking functions, Aequationes Math. 10 (1974) 10–22.
[2] J. Françon, Acyclic and parking functions, J. Combin. Theory Ser. A 18 (1975) 27–35.
[3] I.M. Gessel, A q-analog of the exponential formula, Discrete Math. 40 (1982) 69–80.
[4] I.M. Gessel, Counting forests by descents and leaves, The Foata Festschrift, El. J. Combin. 3(2) (1996) #R8, 5pp.
[5] I.M. Gessel, 1996, unpublished manuscript.
[6] I.M. Gessel, K.-Y. Wang, A bijective approach to the permutational power of a priority queue, 1994, unpublished.
[7] J.D. Gilbey, L.H. Kalikow, Parking functions, valet functions and priority queues, Discrete Math. 197–198(1–3) (1999) 351–373.
[8] L.H. Kalikow, Enumeration of parking functions, allowable permutation pairs, and labeled trees, Ph.D. Thesis, Brandeis University, 1999.
[9] D.E. Knuth, The Art of Computer Programming, Vol. 1: Fundamental Algorithms, Addison-Wesley, Reading, MA, 1973.
[10] D.E. Knuth, Solution to problem 10572, 1997, unpublished. The original problem was proposed by Richard P. Stanley, Problem 10572 in the American Mathematical Monthly 104 (1997) 168.
[11] A.G. Konheim, B. Weiss, An occupancy discipline and applications, SIAM J. Applied Math. 14 (1966) 1266–1274.
[12] G. Kreweras, Une famille de pôlynomes ayant plusieurs propriétés énumeratives, Periodica Mathematica Hungarica 11 (1980) 309–320.
[13] A.G. Kuznetzov, I.M. Pak, A.E. Postnikov, Increasing trees and alternating permutations, Russian Math. Surveys 49(6) (1994) 79–110.
[14] A. Postnikov, Private communication (to I. Gessel, 1995).
[15] R.P. Stanley, Enumerative Combinatorics, Vol. 1, Cambridge University Press, Cambridge, 1997. (First published by Wadsworth & Brooks/Cole, Monterey, CA, 1986.)
[16] R.P. Stanley, Enumerative Combinatorics, Vol. 2, Cambridge University Press, Cambridge, 1999.