Permutation Group Algebras and Parking Functions

Julian David Gilbey

Queen Mary & Westfield College
University of London

Submitted for the degree of
Doctor of Philosophy

2002

Part I

לזכר אבא מרי

נח בן יעקב ז״ל

In memory of my father
who was the first to teach me that
$(n + \frac{1}{2})^2 = n(n + 1) + \frac{1}{4}$

Part II

Dedicated to Ann Cook

Departmental secretary from before my arrival
until her retirement in December 1996

Acknowledgements

It would be impossible in this short space to individually thank everyone who has helped me to bring this thesis into existence; a few special 'thank you's must suffice.

On the academic side, Prof. Peter Cameron, my supervisor, has been wonderful. From explaining F_{20} to me when we first met, through to these final stages, his continual good cheer, encouragement and belief in me (not to mention his encyclopædic knowledge!) have kept me going throughout a very tough period of my life. Also, many of the ideas in Part I of this thesis had their origins in our weekly meetings.

Our departmental combinatorics study group, including in its number Prof. Rosemary Bailey and Dr. Leonard Soicher, gave me an opportunity to present various results and saved me from grave error at least once.

I met my co-author, Louis Kalikow, at his wedding shower. His unexpected email in response to some mathematics I wrote for him on a serviette (napkin) led to a fruitful and enjoyable collaboration, the results of which you can see in Part II of this thesis. Ben Tarlow had introduced me to the problem, and Louis's wife Aurora Mendelsohn was responsible for the colourful naming of the objects. Thanks, all!

The postgrads in rooms 203 and 201 have been fun and stimulating to be around—cheers, guys!

The secretarial staff—especially Ann Cook, whose caring for the postgrads even extended to hunting me down in Heathrow Airport to tell me that I had received a grant—have constantly helped me with all of those niggling little (and big) practical things.

A big collective thanks to all those who wrote and keep Linux, Debian and TEX and friends running; without them, this thesis would have had to be typed on a typewriter and the symbols written in by hand.

Finally, on the nonacademic side of life, thanks to my mum for supporting me through a long and tough stretch of graduate life. Thanks to Lis for being, and to Melissa for rescuing me. A special thank you goes to my 'surrogate mothers' Sue, Tamra and Leo for supporting and feeding me, along with their husbands Clive, Ian and Jonathan; their highly intelligent kids also provided much needed relief and other challenges. Also, thanks goes to my various housemates for keeping me in good spirits and livening the house with music at all hours.

There are many others who have kept me going in myriad ways; to all of them I am grateful.

London, December 2001

Abstract

Part I

Permutation Group Algebras

We consider the permutation group algebra defined by Cameron and show that if the permutation group has no finite orbits, then no homogeneous element of degree one is a zero-divisor of the algebra. We proceed to make a conjecture which would show that the algebra is an integral domain if, in addition, the group is oligomorphic. We go on to show that this conjecture is true in certain special cases, including those of the form H Wr S and H Wr A, and show that in the oligormorphic case, the algebras corresponding to these special groups are polynomial algebras. In the H Wr A case, the algebra is related to the shuffle algebra of free Lie algebra theory. We finish by considering some integer sequences which arise from certain of these groups.

Part II

Parking Functions, Valet Functions and Priority Queues

Parking functions on $[n] = \{1, \ldots, n\}$ are those functions $p : [n] \to [n]$ satisfying the condition $|\{\, i : p(i) \leqslant r \,\}| \geqslant r$ for each r, and are $(n+1)^{n-1}$ in number. These are equinumerate with allowable input-output pairs of permutations of $[n]$ in a priority queue. We present a new bijection between parking functions and allowable pairs which has many interesting invariance properties. We extend our bijection to allowable pairs of multisets and introduce valet functions as the corresponding extension of parking functions. Using our bijection, we interpret the inversion enumerator for trees in the case of allowable pairs. We end with a comparison of our bijection with other known bijections involving these combinatorial structures, including a new bijection between parking functions and labelled trees.

Table of Contents

Part I

Permutation Group Algebras

Table of Contents

Part II

Parking Functions, Valet Functions and Priority Queues

Part I

Permutation Group Algebras

In which we make some progress on a conjecture of Cameron, and discover some interesting connections with free Lie algebras

1 Introduction

Let G be a permutation group on an (infinite) set Ω. Cameron [2] defined a commutative, associative, graded algebra $A(G)$ which encodes information about the action of G on finite subsets of Ω. It is known that this algebra has zero divisors if G has any finite orbits. The question of what happens when G has no finite orbits is the subject of several conjectures due to Cameron [2], and we will be exploring two of them. The first is:

Conjecture 1.1. If G has no finite orbits, then ε is a prime element in $A(G)$.

Here ε is a certain element in the degree one component of the algebra, defined in section 2. The following weaker conjecture would follow from this, as we explain below.

Conjecture 1.2. If G has no finite orbits, then $A(G)$ is an integral domain.

The first conjecture would give us insight into the following question. If the number of orbits of G on unordered k-element subsets of Ω is n_k, then for which groups does $n_k = n_{k+1} < \infty$ hold? We will not study this question directly here; more information can be found in [2] and [3, sect. 3.5].

We first show that no homogeneous element of degree one in the algebra is a zero-divisor. Unfortunately, it is not obvious how to extend this argument to higher degrees. We then go on to give a conjecture which would, if proven, yield a proof of the weaker conjecture 1.2, and show that it holds in two interesting classes of permutation groups. It also turns out in these two cases that the algebra $A(G)$ is a polynomial algebra, and we determine an explicit set of polynomial generators. It will follow that the stronger conjecture also holds in these cases. Although these results do not help to answer the question raised in the previous paragraph (as in these cases, $n_k < n_{k+1}$ for all k), they do provide further evidence to support the conjectures.

Finally, using the inverse Euler transform, Cameron [5] determined the number of polynomial generators of each degree which would be needed for certain of these algebras if they were actually polynomial algebras. Some of these sequences appear in The On-Line Encyclopedia of Integer Sequences [12] in the context of free Lie algebras. Our work gives an explanation for the sequences observed and the connection with free Lie algebras.

2 The graded algebra of a permutation group

We now give the definition of the algebra under consideration. Let G be a permutation group acting on Ω. Let K be a field of characteristic 0 (either $\mathbb{Q}$ or $\mathbb{C}$ will do). Define $V_n(G)$ to be the K-vector space of all functions from n-subsets of Ω to K which are invariant under the natural action of G on n-subsets of Ω. Define the graded algebra

$$A(G) = \bigoplus_{n=0}^{\infty} V_n(G)$$

with multiplication defined by the rule that for any $f \in V_m(G)$ and $g \in V_n(G)$, the product $fg \in V_{m+n}(G)$ is such that for any $(m+n)$-subset $X \subseteq \Omega$,

$$(fg)(X) = \sum_{\substack{Y \subseteq X \\ |Y|=m}} f(Y)g(X \setminus Y).$$

It is easy to check that, with this multiplication, $A(G)$ is a commutative, associative, graded algebra.

If G has any finite orbits, then this algebra contains zero-divisors. For let $X \subseteq \Omega$ be a finite orbit, $|X| = n$, and let $f \in V_n(G)$ be the characteristic function of this set (so $f(X) = 1$ and $f(Y) = 0$ for $Y \neq X$); then clearly $f^2 = 0$.

Considering Conjecture 1.2, it is clear that there are no zero-divisors in $V_0(G)$, as multiplying by an element of $V_0(G)$ is equivalent to multiplying by an element of K.

We also note that if there is a zero-divisor in $A(G)$, so we have $fg = 0$ with $0 \neq f, g \in A(G)$, then we can consider the non-zero homogeneous components of f and g with lowest degree; say these are f_m of degree m and g_n of degree n respectively. Then the term of degree $m+n$ in fg will be precisely $f_m g_n$, and as $fg = 0$, we must have $f_m g_n = 0$. So we may restrict our attention to considering homogeneous elements, and showing that for any positive integers m and n, we cannot find non-zero $f \in V_m(G)$ and $g \in V_n(G)$ with $fg = 0$.

Furthermore, we will show in the next section that $V_1(G)$ contains no zero-divisors as long as G has no finite orbits, so in particular, the element $\varepsilon \in V_1(G)$ defined by $\varepsilon(x) = 1$ for all $x \in \Omega$ is a non-zero-divisor. So if f is a homogeneous zero-divisor of degree m, with $fg = 0$, and g is homogeneous of degree $n > m$, we also have $(\varepsilon^{n-m} f)g = 0$, so $\varepsilon^{n-m} f \neq 0$ is a zero-divisor of degree n. Thus, if we wish, we can restrict our attention to showing that, for each positive integer n, we cannot find non-zero $f, g \in V_n(G)$ with $fg = 0$.

Turning now to the stronger Conjecture 1.1, we see that the second conjecture

follows from this (as in [2]). For if $fg = 0$, with f and g homogeneous and non-zero, and $\deg f + \deg g$ is minimal subject to this, then $\varepsilon \mid fg$, so we can assume $\varepsilon \mid f$ by primality. Thus $f = \varepsilon f'$, and $\deg f' = \deg f - 1$. Thus $\varepsilon f' g = 0$, which implies $f'g = 0$ by the above, contrary to the minimality of $\deg f + \deg g$.

3 The degree one case

We intend to prove the following theorem.

Theorem 3.1. *If G has no finite orbits, then $V_1(G)$ contains no zero-divisors.*

In order to prove this theorem, we will make use of a technical proposition, which is based on a theorem of Kantor [8]. We first quote a version of Kantor's theorem, as we will have use for it later.

Proposition 3.2. *Let $0 \leqslant e < f \leqslant d - e$. Let X be a set with $|X| = d$. We define (E, F) for subsets $E, F \subset X$ with $|E| = e$ and $|F| = f$ by*

$$(E, F) = \begin{cases} 1 & \text{if } E \subset F \\ 0 & \text{otherwise,} \end{cases}$$

and the matrix $M = ((E, F))$, where the rows of M are indexed by the e-subsets of X and the columns by the f-subsets.

Then $\operatorname{rank} M = \binom{d}{e}$.

The extension of this result is as follows.

Proposition 3.3. *Let $0 \leqslant e < f \leqslant d - 2e$. Let X be a set with $|X| = d$, and let $E_0 \subset X$ with $|E_0| = e$ be a distinguished subset of X. Let w be a weight function on the $(f - e)$-subsets of X with values in the field K, satisfying the condition that $w(X') = 1$ whenever X' is an $(f - e)$-subset of X such that $X' \nsubseteq E_0$. We define (E, F) for subsets $E, F \subset X$ with $|E| = e$ and $|F| = f$ by*

$$(E, F) = \begin{cases} w(F \setminus E) & \text{if } E \subset F \\ 0 & \text{otherwise,} \end{cases}$$

and the matrix $M = ((E, F))$, where the rows of M are indexed by the e-subsets of X and the columns by the f-subsets.

Then $\operatorname{rank} M = \binom{d}{e}$.

Proof of theorem 3.1. Let $g \in V_1(G)$ with $g \neq 0$, and assume $h \in V_n(G)$ with $n \geqslant 1$ and $gh = 0$ (the $n = 0$ case has been dealt with in section 2). We must show that $h = 0$, so that for any $Y \subset \Omega$ with $|Y| = n$, we have $h(Y) = 0$. We assume that a set Y has been fixed for the remainder of this proof.

Since $g \neq 0$, there exists some (infinite) orbit $\Delta \subseteq \Omega$ on which g is non-zero; multiplying by a scalar if necessary, we may assume that $g(\delta) = 1$ for all $\delta \in \Delta$. Pick $X \subset \Omega$ with $|X| = 3n + 1$, $Y \subset X$ and $X \setminus Y \subset \Delta$.

Now for any $(n+1)$-subset $F \subset X$, we have $(hg)(F) = 0$ as $gh = hg = 0$, so that

$$(hg)(F) = \sum_{\substack{E \subset F \\ |E|=n}} h(E)g(F \setminus E) = 0.$$

This can be thought of as a system of linear equations in the unknowns $h(E)$ for $E \subset X$, $|E| = n$, with the matrix $M = (m_{EF})$ given by $m_{EF} = g(F \setminus E)$ if $E \subset F$, and $m_{EF} = 0$ otherwise.

This is precisely the situation of the proposition if we let $e = n$, $f = n+1$ (so that $f - e = 1$), $d = 3n + 1$, $E_0 = Y$ and $w(\alpha) = g(\alpha)$; note that $w(\alpha) = 1$ whenever $\alpha \notin E_0$. (We write $g(\alpha)$ instead of the more correct $g(\{\alpha\})$; no confusion should arise because of this.) Thus $\operatorname{rank} M = \binom{d}{e}$ and the system of equations has a unique solution, which must be $h(E) = 0$ for all $E \subset X$ with $|E| = n$, as this is a possible solution. In particular, this means that $h(Y) = 0$, and since Y was chosen arbitrarily, it follows that $h = 0$.

Hence g is not a zero-divisor. □

Proof of proposition 3.3. Let $R(E)$ be the row of M corresponding to E. M has $\binom{d}{e}$ rows, so we must show that the rows are linearly independent. We thus assume that there is a linear dependence among the rows of M, so

$$R(E^*) = \sum_{E \neq E^*} a(E)R(E) \tag{1}$$

for some e-set E^* and some $a(E) \in K$. We first note that $R(E^*)$ itself is non-zero: this follows as we can pick some $F \supset E^*$ with $F \setminus E^* \not\subseteq E_0$; for this F, we have $(E^*, F) = 1$.

Let Γ be the subgroup of $\mathrm{Sym}(X)$ which stabilises E_0 pointwise and E^* setwise. If $\sigma \in \Gamma$, then

$$(E^\sigma, F^\sigma) = \begin{cases} w((F \setminus E)^\sigma) = w(F \setminus E) & \text{if } E \subset F \\ 0 & \text{otherwise;} \end{cases}$$

either way, $(E^\sigma, F^\sigma) = (E, F)$. (For the result $w((F \setminus E)^\sigma) = w(F \setminus E)$, note that both sides are equal to 1 unless $F \setminus E \subseteq E_0$, in which case σ fixes this set pointwise.) Thus (1) implies that, for all F,

$$\begin{aligned}(E^*, F) = (E^*, F^\sigma) &= \sum_{E \neq E^*} a(E^\sigma)(E^\sigma, F^\sigma) \\ &= \sum_{E \neq E^*} a(E^\sigma)(E, F).\end{aligned}$$

Thus

$$R(E^*) = \sum_{E \neq E^*} a(E^\sigma) R(E).$$

It follows that

$$\begin{aligned} |\Gamma|\, R(E^*) &= \sum_{\sigma \in \Gamma} \sum_{E \neq E^*} a(E^\sigma) R(E) \\ &= \sum_{E \neq E^*} R(E) \sum_{\sigma \in \Gamma} a(E^\sigma). \end{aligned} \tag{2}$$

We now consider the orbits of Γ on the e-subsets of X, excluding E^*. The e-sets E_1 and E_2 will lie in the same orbit if and only if $E_1 \cap E_0 = E_2 \cap E_0$ and $|E_1 \cap E^*| = |E_2 \cap E^*|$. Thus every orbit is described by a subset $E' \subseteq E_0$ and an integer $0 \leqslant i \leqslant e-1$. (We cannot have $i = e$, as we are excluding E^* from consideration.) Clearly not all possible pairs (E', i) will actually correspond to an orbit (it is not hard to see that necessary and sufficient conditions for this are $|E' \cap E^*| \leqslant i \leqslant \min\{e-1, e-|E' \setminus E^*|\}$), so that whenever we consider or sum over such pairs below, we implicitly restrict attention to those which correspond to an orbit. In such cases, we write $\mathcal{E}(E', i)$ for the orbit. Also, for each such pair, pick some $E(E', i) \in \mathcal{E}(E', i)$. Then (2) implies

$$\begin{aligned} |\Gamma|\, R(E^*) &= \sum_{(E',i)} \sum_{E \in \mathcal{E}(E',i)} R(E) \sum_{\sigma \in \Gamma} a(E^\sigma) \\ &= \sum_{(E',i)} \sum_{E \in \mathcal{E}(E',i)} R(E) \sum_{\sigma \in \Gamma} a(E(E',i)^\sigma) \\ &= \sum_{(E',i)} \sum_{\sigma \in \Gamma} a(E(E',i)^\sigma) \sum_{E \in \mathcal{E}(E',i)} R(E) \end{aligned}$$

so that

$$R(E^*) = \sum_{(E',i)} b(E',i) \sum_{E \in \mathcal{E}(E',i)} R(E) \tag{3}$$

with $b(E', i) \in K$, and clearly not all of the $b(E', i)$ can be zero as $R(E^*)$ is not zero.

We define a total order on the pairs (E', i) as follows. Extend the partial order given by $\subseteq$ on the subsets of E_0 to a total order $\leqslant$, and then define $(E', i) \leqslant (E'', j)$ if $E' < E''$ or $E' = E''$ and $i \leqslant j$. We now proceed to derive a contradiction by showing that (3) leads to a system of linear equations for the $b(E', i)$ which is triangular under this total order, with non-zero diagonal entries, and deduce that all of the $b(E', i)$ must be zero.

Let $(\bar{E}, n)$ be a pair corresponding to an orbit. Since $2e + f \leqslant d$, there exists

an f-set $F(\bar{E}, n)$ satisfying $F(\bar{E}, n) \cap E_0 = \bar{E}$ and $|F(\bar{E}, n) \cap E^*| = n$. (Simply take $E(\bar{E}, n)$ and adjoin $f - e$ points lying in $X \setminus (E_0 \cup E^*)$.) As $n \leqslant e - 1$, it follows that $F(\bar{E}, n) \not\supseteq E^*$, so $(E^*, F(\bar{E}, n)) = 0$. Hence by (3), we have

$$0 = \sum_{(E',i)} b(E', i) \sum_{E \in \mathcal{E}(E',i)} (E, F(\bar{E}, n)) \tag{4}$$

for all such pairs $(\bar{E}, n)$.

We note that $F(\bar{E}, n) \cap E_0 = \bar{E}$, and further that $E \in \mathcal{E}(E', i)$ implies that $E \cap E_0 = E'$; thus for the term $(E, F(\bar{E}, n))$ in equation (4) to be non-zero, where $E \in \mathcal{E}(E', i)$, we require $E' \subseteq \bar{E}$, hence also $E' \leqslant \bar{E}$. Furthermore, if $(E, F(\bar{E}, n)) \neq 0$, we must have $i \leqslant n$ as $E \subset F(\bar{E}, n)$. Thus if $(\bar{E}, n) < (E', i)$, we have

$$\sum_{E \in \mathcal{E}(E',i)} (E, F(\bar{E}, n)) = 0. \tag{5}$$

Also, there is an e-set $E \subset F(\bar{E}, n)$ satisfying $E \cap E^* = F(\bar{E}, n) \cap E^*$ and $E \cap E_0 = F(\bar{E}, n) \cap E_0 = \bar{E}$; just take the union of $\bar{E}$ with $F(\bar{E}, n) \cap E^*$ and sufficiently many remaining points of $F(\bar{E}, n)$. For each such E, we have $F(\bar{E}, n) \setminus E \not\subseteq E_0$, so $(E, F(\bar{E}, n)) = 1$. Since K has characteristic zero, we deduce that

$$\sum_{E \in \mathcal{E}(\bar{E},n)} (E, F(\bar{E}, n)) \neq 0, \tag{6}$$

as the sum is over all sets of precisely this form.

It then follows from (4) and (5) that for each pair $(\bar{E}, n)$:

$$0 = \sum_{(E',i) \leqslant (\bar{E},n)} b(E', i) \sum_{E \in \mathcal{E}(E',i)} (E, F(\bar{E}, n)).$$

Now this is a system of linear equations in the unknowns $b(E', i)$ which is lower triangular. Also, by (6), the diagonal entries are non-zero. It follows that the unique solution to this system is that all of the $b(E', i)$ are zero, which provides the required contradiction to equation (3) above. □

4 Oligomorphic-type cases: our conjecture

4.1 Ramsey orderings on orbits of n-sets

Cameron proved the following Ramsey-type result in [3, Prop. 1.10].

Lemma 4.1. *Suppose that the n-sets of an infinite set X are coloured with r colours, all of which are used. Then there is an ordering $c_1, \ldots, c_r$ of the colours and infinite subsets $X_1, \ldots, X_r$, such that X_i contains an n-set of colour c_i but no set of colour c_j for $j > i$.*

We use this as the inspiration for the following definition. If G is a permutation group on Ω, we say that the orbits of G on n-sets of Ω can be *Ramsey ordered* if, given any finite $N > n$, there is an ordering of the orbits c_α, $\alpha \in \mathcal{A}$, where $\mathcal{A}$ is a well-ordered set, and a corresponding sequence of (possibly infinite) subsets $X_\alpha \subseteq \Omega$ with $|X_\alpha| \geqslant N$, and such that X_α contains an n-set in the orbit c_α but no n-set in an orbit c_β for $\beta > \alpha$. (We can take $\mathcal{A}$ to be a set of ordinals with the $\in$-ordering if we wish; this is the reason for using Greek letters.) This pair of sequences forms a *Ramsey ordering*. While the particular Ramsey ordering may depend on N, we do not usually mention N unless we have to. The reader may think throughout of N having a very large finite value. It turns out that this makes certain constructions below simpler than if we required the X_α to be infinite sets.

Not every permutation group has such an ordering. For example, in the regular action of $\mathbb{Z}$ on $\mathbb{Z}$, there is no set with more than two elements, all of whose 2-subsets are in the same orbit, so there cannot be a Ramsey ordering on 2-subsets. However, Cameron's result implies that if G is *oligomorphic* (that is, there are only finitely many orbits on n-sets for each n), then the orbits of G on n-sets can be Ramsey ordered for each n.

It turns out that Ramsey orderings on n-sets naturally yield Ramsey orderings on m-sets whenever $m < n$.

Proposition 4.2. *Let G be a permutation group acting on an infinite set Ω. Let $m < n$ be positive integers, and assume that the n-set orbits of G can be Ramsey ordered, say c_α and X_α with $\alpha \in \mathcal{A}$ are a Ramsey-ordering with $N \geqslant m + n$. Then this ordering induces a Ramsey ordering on the m-set orbits as follows. There is a subset $\mathcal{B} \subseteq \mathcal{A}$ and a labelling of the m-set orbits as d_β, $\beta \in \mathcal{B}$, such that for each $\beta \in \mathcal{B}$, an m-set in the orbit d_β appears in X_β, and that for each $\alpha \in \mathcal{A}$, X_α contains no m-sets in the orbit d_β for $\beta > \alpha$.*

We call the ordering of orbits d_β, $\beta \in \mathcal{B}$ together with the corresponding sets X_β given by this proposition the *induced Ramsey ordering*. Note that we use the same parameter N in both orderings.

The proof uses the following application of Kantor's theorem (Proposition 3.2 above), shown to me by Peter Cameron.

Lemma 4.3. *Let $m < n$ be positive integers, and let X be a finite set with $|X| \geqslant m+n$. Let the m-sets of X be coloured with colours from the set $\mathbb{N}$. Given an n-subset of X, we define its* colour-type *to be the multiset of colours of its $\binom{n}{m}$ m-subsets. Then the number of distinct m-set colours used in X is less than or equal to the number of distinct colour-types among the n-subsets of X.*

Proof. We note that only a finite number of colours appear among the m-subsets of X, as they are finite in number. Without loss of generality, we may assume that the colours used are precisely 1, 2, ..., s.

As in Kantor's theorem (Proposition 3.2), we let M be the incidence matrix of the m-subsets versus n-subsets of X. By that theorem, as $m < n$ and $|X| \geqslant m+n$, this matrix has rank $\binom{|X|}{m}$, which equals the number of rows in the matrix. Thus, by the rank-nullity theorem, M represents an injective linear transformation.

Now for each $i = 1, \ldots, s$, let v_i be the row vector, with entries indexed by the m-subsets of X, whose j-th entry is 1 if the j-th m-subset has colour i, and 0 if it does not. Then v_iM is a row vector, indexed by the n-subsets of X, whose k-th entry is the number of m-subsets of the k-th n-subset which have colour i.

Consider now the matrix M' whose rows are $v_1M, \ldots, v_sM$. Note that the k-th column of this matrix gives the colour-type of the k-th n-subset of X. Its rank is given by

$$\operatorname{rank} M' = \dim \langle v_1M, \ldots, v_sM \rangle = \dim \langle v_1, \ldots, v_s \rangle = s,$$

as M represents an injective linear transformation, and the s vectors $v_1, \ldots, v_s$ are clearly linearly independent. Now since the row rank and column rank of a matrix are equal, we have $s = \operatorname{rank} M' \leqslant$ number of distinct columns in M', which is the number of n-set colour-types in X. Thus the number of m-set colours appearing in X is less than or equal to the number of n-set colour-types in X, as we wanted. □

Proof of Proposition 4.2. Let c_α be any n-set orbit, and let X be a representative of this orbit. We observe that the multiset of m-set orbits represented by the $\binom{m}{n}$ m-subsets of X is independent of the choice of X in this orbit. (For let $\bar{X}$ be another representative of the orbit c_α, with $\bar{X} = g(X)$, where $g \in G$. Then the set of m-subsets of X is mapped to the set of m-subsets of $\bar{X}$ by g, and so the multisets of m-set orbits represented by these two sets are identical.) In particular, we may say that an n-set orbit contains an m-set orbit, meaning that any representative of the n-set orbit contains a representative of the m-set orbit.

We first claim that every m-set orbit appears in some X_α: take a representative of an m-set orbit, say $Y \subset \Omega$. Adjoin a further $n-m$ elements to get an n-set $\bar{X}$. This n-set lies in some orbit, so there is a representative of this orbit in one of the X_α, say $X \subset X_\alpha$. Then this X_α contains a representative of our m-set orbit by the above argument, as we wished to show.

Now if $Y \subset \Omega$ is a representative of an m-set orbit, we set

$$\beta_Y = \min\{\, \alpha : g(Y) \subset X_\alpha \text{ for some } g \in G \,\}.$$

Note that this implies that the m-set orbit containing Y is contained in c_{β_Y} but not in c_α for any $\alpha < \beta_Y$. We set $\mathcal{B} = \{\, \beta_Y : Y \subset \Omega \text{ and } |Y| = m \,\}$, and if Y is an m-set, then we set d_{β_Y} to be the orbit of Y. We claim that $\mathcal{B}$ satisfies the conditions of the proposition with this orbit labelling. Certainly an m-set in the orbit d_{β_Y} appears in X_{β_Y} for each Y, by construction, and for each $\alpha \in \mathcal{A}$, X_α contains no m-sets in the orbit d_β for $\beta > \alpha$, again by construction. However, for d_{β_Y} to be well-defined, we require that $\beta_{Y_1} \neq \beta_{Y_2}$ if Y_1 and Y_2 lie in distinct orbits. We now show this to be the case by demonstrating that given any $\alpha_0 \in \mathcal{A}$, there can only be one m-set orbit appearing in c_{α_0} which has not appeared in any c_α with $\alpha < \alpha_0$.

So let $\alpha_0 \in \mathcal{A}$, and let $X \subseteq X_{\alpha_0}$ have size $m+n$ and contain an n-set in the orbit c_{α_0}. By the observation we made above, namely that the m-set orbits appearing in an n-set are independent of the choice of the n-set in its n-set orbit, it suffices to show that our set X contains at most one new m-set orbit. To use the lemma, we colour the m-subsets of X as follows. If Y is an m-set with $\beta_Y < \alpha_0$, then Y is given colour 1. Those $Y \subset X$ with $\beta_Y = \alpha_0$ are given the colours 2, 3, ..., with a distinct colour per m-set orbit. (Note that any $Y \subset X$ has $\beta_Y \leqslant \alpha_0$, as all n-subsets of X_{α_0} lie in orbits c_α with $\alpha \leqslant \alpha_0$.)

We now consider the possible colour-types of the n-sets of X. Note first that since the m-sets in a given m-set orbit all have the same colour, the colour-type of an n-set depends only upon the n-set orbit in which it lies. There is some n-subset of X in the orbit c_{α_0} by construction, and this has a certain colour-type. Any other n-subset $\tilde{X} \subset X$ is either in the same orbit c_{α_0}, and so has the same colour-type, or it is in some other orbit c_α with $\alpha < \alpha_0$. In the latter case, every m-subset $Y \subset \tilde{X}$ must have $\beta_Y \leqslant \alpha < \alpha_0$, and so it has colour 1. Thus the colour-type of such an n-set must be the multiset $[1, 1, \ldots, 1]$.

If every n-subset of X is in the orbit c_{α_0}, then there is only one colour-type, and so there can only be one m-set colour in X by the lemma, that is, only one m-set orbit with $\beta_Y = \alpha_0$. On the other hand, if X contains an n-set in an orbit c_α with $\alpha < \alpha_0$, then there are at most two colour-types in X: the all-1 colour-type and the colour-type of c_{α_0}. Thus, by the lemma, X contains at most

two m-set colours. Colour 1 appears in c_α, and so there is at most one other colour present, that is, there is at most one m-set orbit with $\beta_Y = \alpha_0$. Thus d_{β_Y} is well-defined on m-set orbits, and we are done. □

4.2 The Ramsey-ordering conjecture

Let G be a permutation group on Ω and let m and n be positive integers. Let d be an m-set orbit and e an n-set orbit. If c is an $(m+n)$-set orbit, then we say that c *contains a* $d \cup e$ *decomposition* if an $(m+n)$-set X in the orbit c can be written as $X = X_m \cup X_n$ with X_m in d and X_n in e. We can easily show using a theorem of P. M. Neumann that if G has no finite orbits, then for every pair (d, e), there exists an $(m+n)$-set orbit c containing a $d \cup e$ decomposition, as follows.

Neumann [9] proved the following: Let G be a permutation group on Ω with no finite orbits, and let Δ be a finite subset of Ω. Then there exists $g \in G$ with $g\Delta \cap \Delta = \varnothing$. It follows trivially that if Y and Z are finite subsets of Ω, then there exists $g \in G$ with $gY \cap Z = \varnothing$ (just take $\Delta = Y \cup Z$). In our case, let X_m and X_n be representatives of d and e respectively. Then there exists $g \in G$ with $gX_m \cap X_n = \varnothing$, and $gX_m \cup X_n$ is an $(m+n)$-set with the required decomposition, hence we can take c to be its orbit.

We will be considering groups G which have a Ramsey ordering on their $(m+n)$-set orbits. Let c_α, $\alpha \in \mathcal{A}$ be the ordering on $(m+n)$-sets, and let d_β, $\beta \in \mathcal{B}$ and e_γ, $\gamma \in \mathcal{C}$ be the induced Ramsey orderings on m- and n-sets respectively (where we assume N is sufficiently large). We then define

$$\beta \vee \gamma = \min\{\, \alpha : c_\alpha \text{ contains a } d_\beta \cup e_\gamma \text{ decomposition} \,\}.$$

Here is our main conjecture.

Conjecture 4.4. Let G be a permutation group on Ω with no finite orbits and for which the orbits on n-sets can be Ramsey ordered for every n. Then given positive integers m and n, there exists some Ramsey ordering of the orbits on $(m+n)$-sets with $N \geqslant 2(m+n)$, say c_α, $\alpha \in \mathcal{A}$ with corresponding sets $X_\alpha \subseteq \Omega$, which induces Ramsey orderings d_β, $\beta \in \mathcal{B}$ and e_γ, $\gamma \in \mathcal{C}$ on the m-set orbits and n-set orbits respectively, and which satisfies the following conditions for all $\beta, \beta' \in \mathcal{B}$ and $\gamma, \gamma' \in \mathcal{C}$:

$$\beta \vee \gamma < \beta' \vee \gamma \text{ if } \beta < \beta' \quad \text{and} \quad \beta \vee \gamma < \beta \vee \gamma' \text{ if } \gamma < \gamma'.$$

Note that the conditions of this conjecture also imply that if $\beta < \beta'$ and $\gamma < \gamma'$, then $\beta \vee \gamma < \beta \vee \gamma' < \beta' \vee \gamma'$, so that $\beta \vee \gamma \leqslant \beta' \vee \gamma'$ implies that either

$\beta < \beta'$ or $\gamma < \gamma'$ or $(\beta, \gamma) = (\beta', \gamma')$.

Given this conjecture, it is easy to show that $A(G)$ is an integral domain for such groups. For if $fg = 0$ with $0 \neq f \in V_m(G)$ and $0 \neq g \in V_n(G)$, let β_0 be such that $f(d_\beta) = 0$ for $\beta < \beta_0$ but $f(d_{\beta_0}) \neq 0$, and let γ_0 be such that $g(e_\gamma) = 0$ for $\gamma < \gamma_0$ but $g(e_{\gamma_0}) \neq 0$. (We write $f(d_\beta)$ to mean the value of $f(Y)$ where Y is any representative of the orbit d_β, and so on.) Letting $\alpha_0 = \beta_0 \vee \gamma_0$, we can consider $fg(c_{\alpha_0})$. Now since $fg = 0$, this must be zero, but we can also determine this explicitly. Letting X be a representative of c_{α_0}, we have

$$fg(c_{\alpha_0}) = fg(X) = \sum_{\substack{Y \subset X \\ |Y| = m}} f(Y) g(X \setminus Y).$$

Every term in the sum is of the form $f(d_\beta)g(e_\gamma)$ where $d_\beta \cup e_\gamma$ is a decomposition of c_{α_0}, so that $\beta \vee \gamma \leqslant \alpha_0 = \beta_0 \vee \gamma_0$. But by the conjecture, this implies that except for terms of the form $f(d_{\beta_0})g(e_{\gamma_0}) \neq 0$, every term either has $\beta < \beta_0$ so that $f(d_\beta) = 0$, or $\gamma < \gamma_0$ so that $g(e_\gamma) = 0$, and hence every one of these terms is zero. Since there exist terms of the form $f(d_{\beta_0})g(e_{\gamma_0})$ by the choice of α_0, we must have $fg(c_{\alpha_0}) \neq 0$. But this contradicts $fg = 0$, and so $A(G)$ is an integral domain.

Recall from section 2 that we can assume $m = n$ when showing that $A(G)$ is an integral domain (that is, $fg = 0$ where $f, g \in V_n(G)$ implies $f = 0$ or $g = 0$); hence we can restrict ourselves to proving the conjecture in the case $m = n$ if this is easier.

5 Special cases (I): Wreath-S-like groups

5.1 Notational conventions

We gather here some notation that we will be using for the rest of this part of the thesis.

We will make use of the lexicographical order on finite sequences and multisets, which we define as follows. Let $(X, <)$ be a totally ordered set. If $x = (x_1, \dots, x_r)$ and $y = (y_1, \dots, y_s)$ are two ordered sequences of elements of X, then we say that x is lexicographically smaller than y, written $x <_{\text{lex}} y$, if there is some t with $x_i = y_i$ for all $i < t$, but either $x_t < y_t$ or $r + 1 = t \leqslant s$. If we now take a finite multiset of elements of X, say M, we write $\text{seq}(M)$ to mean the sequence obtained by writing the elements of M (as many times as they appear in M) in decreasing order. Then if M_1 and M_2 are finite multisets, we define $M_1 <_{\text{lex}} M_2$ to mean $\text{seq}(M_1) <_{\text{lex}} \text{seq}(M_2)$. Note that $<_{\text{lex}}$ is a total order on the set of finite multisets, for $\text{seq}(M_1) = \text{seq}(M_2)$ if and only if $M_1 = M_2$. If we need to explicitly list the elements of a multiset, we will write $[x_1, x_2, \dots]$. We write $M_1 + M_2$ for the multiset sum of the multisets M_1 and M_2, so if $M_1 = [x_1, \dots, x_r]$ and $M_2 = [y_1, \dots, y_s]$, then $M_1 + M_2 = [x_1, \dots, x_r, y_1, \dots, y_s]$.

In the following sections, we will talk about a set of *connected blocks* for a permutation group, the idea being that every orbit will correspond to a multiset or sequence of connected blocks. The choice of terminology will be explained below, and is not related to blocks of imprimitivity. Also, the individual words "connected" and "block" have no intrinsic meaning in the context of the definitions in this thesis. Every connected block has a positive integral weight (for which we write $\text{wt}(\Delta)$), and the weight of a sequence or multiset of connected blocks is just the sum of weights of the individual connected blocks. We well-order the connected blocks of each weight, and denote the connected blocks of weight i by $\Delta_i^{(j)}$, where j runs through some well-ordered indexing set. Without loss of generality, we assume that $\Delta_1^{(1)}$ is the least connected block of weight 1. We then define a well-ordering on all connected blocks by $\Delta_i^{(j)} < \Delta_{i'}^{(j')}$ if $i < i'$ or $i = i'$ and $j < j'$. Using this ordering, we can then talk about the lexicographic ordering on sequences or multisets of connected blocks.

5.2 Wreath-S-like groups

Our prototypical family of groups for this class of groups are those of the form $G = H \text{ Wr } S$, where H is a permutation group on Δ and $S = \text{Sym}(\mathbb{Z})$, the symmetric group acting on a countably infinite set (we take the integers for convenience). The action is the imprimitive one, so G acts on $\Omega = \Delta \times \mathbb{Z}$. We

extract those features of this group which are necessary for the proof below to work.

Definition 5.1. We say that a permutation group G on Ω is *wreath-S-like* if there is a set of connected blocks $\{\Delta_i^{(j)}\}$ and a bijection ϕ from the set of orbits of G on finite subsets of Ω to the set of all finite multisets of connected blocks, with the bijection satisfying the following conditions (where we again blur the distinction between orbits and orbit representatives):

(i) If $Y \subset \Omega$ is finite, then $\mathrm{wt}(\phi(Y)) = |Y|$.

(ii) If $Y \subset \Omega$ is finite and $\phi(Y) = [\Delta_{i_1}^{(j_1)}, \ldots, \Delta_{i_k}^{(j_k)}]$, we can partition Y as $Y = Y_1 \cup \cdots \cup Y_k$ with $|Y_l| = i_l$ for each l. Furthermore, if $Z \subseteq Y$ and $Z = Z_1 \cup \cdots \cup Z_k$, where $Z_l \subseteq Y_l$ for each l, then we can write $\phi(Z)$ as a sum of multisets $\phi(Z) = M_1 + \cdots + M_k$, where $\mathrm{wt}(M_l) = |Z_l|$ for each l and $M_l = [\Delta_{i_l}^{(j_l)}]$ if $Z_l = Y_l$.

Note that condition (ii) implies that $\phi(Y_l) = [\Delta_{i_l}^{(j_l)}]$ for $j = 1, 2, \ldots, k$. Essentially, this condition means that subsets of Y correspond to "submultisets" of $\phi(Y)$ in a suitable sense.

In the case of $G = H$ Wr S mentioned above, we take the connected blocks of weight n to be the orbits of the action of H on n-subsets of Δ. Then every orbit of G can be put into correspondence with a multiset of H-orbits as follows. If $Y \subset \Omega$ is an orbit representative, then $\phi(Y) = [\,\pi_i(Y) : \pi_i(Y) \neq \varnothing\,]$, where the π_i are projections: $\pi_i(Y) = \{\,\delta : (\delta, i) \in Y\,\}$, and we identify orbits of H with orbit representatives. Note that $\mathrm{wt}(\phi(Y)) = |Y|$ as required, and that condition (ii) is also satisfied; in fact, in the notation of the condition, we have $M_l = [\Delta_{i'_l}^{(j'_l)}]$ for each l, for some appropriate i'_l and j'_l.

Another example is the automorphism group of the random graph. The random graph is the unique countable homogeneous structure whose age consists of all finite graphs. It is also known as the Fraïssé limit of the set of finite graphs; see Cameron [3] for more information on homogeneous structures and Fraïssé's theorem. We take the set of connected blocks to be the isomorphism classes of finite connected graphs, where the weight of a connected block is the number of vertices in it. Any orbit can be uniquely described by the multiset of connected graph components in an orbit representative. Condition (i) is immediate, as is condition (ii). Note, however, that there are examples in this scenario where M_l may not be a singleton. For example, if $Y = P_2$ is the path of length 2 (with three vertices), so that $\phi(Y) = [P_2]$, and $Z \subset Y$ consists of the two end vertices of the path, then $\phi(Z) = [K_1, K_1]$.

This prototypical example explains the choice of terminology: the basic units in this example are the connected graphs, so we have called our basic units

connected blocks, both to suggest this example and that of strongly connected components in tournaments as considered in section 6 below.

Cameron [4, Sec. 2] has shown that $A(G)$ is a polynomial algebra if G is an oligomorphic wreath-S-like group, from which it follows that $A(G)$ is an integral domain in this case. It also follows that ε is a prime element, so both Conjectures 1.1 and 1.2 hold in this case. The argument that $A(G)$ is a polynomial algebra in the oligomorphic case is similar to that presented below for wreath-A-like groups, only significantly simpler.

We now show, using a new argument based on Ramsey-orderings, that $A(G)$ is an integral domain in the wreath-S-like case, even without the assumption that G is oligomorphic. This will also provide a basis for the arguments presented in the next section for wreath-A-like groups.

Theorem 5.2. *If G is wreath-S-like, then $A(G)$ is an integral domain.*

Proof. We claim that in such a situation, the conditions of Conjecture 4.4 are satisfied, and hence $A(G)$ is an integral domain.

Following the requirements of the conjecture, let m and n be positive integers and pick any integer $N \geqslant 2(m+n)$. Denote the inverse of ϕ by ψ and let α run through all multisets of connected blocks of total weight $m+n$, then we set $c_\alpha = \psi(\alpha)$ and let X_α be an N-set in the orbit $\psi(\alpha + [\Delta_1^{(1)}, \ldots, \Delta_1^{(1)}])$, where the second multiset has $N-(m+n)$ copies of $\Delta_1^{(1)}$. We claim that this gives a Ramsey ordering of the orbits on $(m+n)$-sets, where the multisets are ordered lexicographically (which gives a well-ordering on the multisets). Firstly, every $(m+n)$-set orbit appears among the list by hypothesis, as ψ is a bijection. Secondly, by construction, there is an $(m+n)$-subset of X_α in the orbit $\psi(\alpha)$, namely partition X_α as in condition (ii) of the definition, and remove all of the elements corresponding to the copies of $\Delta_1^{(1)}$ added. This subset will then map to α under ϕ, by condition (ii). Finally, any $(m+n)$-subset of X_α can be seen to correspond to a multiset lexigraphically less than or equal to α, again using condition (ii) and the fact that $\Delta_1^{(1)}$ is the least connected block, so the subset will be in an orbit c_β with $\beta \leqslant_{\text{lex}} \alpha$, as required.

We note that the induced Ramsey orderings on m-set orbits and n-set orbits are given by precisely the same construction. Specifically, let β be a multiset with $\text{wt}(\beta) = n$. Then the orbit corresponding to the multiset β first appears in X_{α_0} where $\alpha_0 = \beta + [\Delta_1^{(1)}, \ldots, \Delta_1^{(1)}]$. For assume that an n-set Z in the orbit $\psi(\beta)$ appears in X_α. As we have $\phi(Z) = \beta$, β must be a "submultiset" of α in the sense of condition (ii), and it is clear that the lexicographically smallest such α is the one given by adjoining an appropriate number of copies of $\Delta_1^{(1)}$ to β. It is not difficult to show that $\beta \vee \gamma$ is precisely the multiset $\beta + \gamma$, and that $\beta <_{\text{lex}} \beta'$ implies $\beta + \gamma <_{\text{lex}} \beta' + \gamma$, and therefore $\beta \vee \gamma <_{\text{lex}} \beta' \vee \gamma$;

similarly, $\gamma <_{\text{lex}} \gamma'$ implies $\beta \vee \gamma <_{\text{lex}} \beta \vee \gamma'$. (The argument is similar to that of Theorem 6.2 below.) Thus the conditions of the conjecture are satisfied by this Ramsey ordering, and hence $A(G)$ is an integral domain. □

6 Special cases (II): Wreath-A-like groups

We can now apply the same ideas used for the wreath-S-like case to the next class of groups, although the details are more intricate. The only essential difference between these two classes is that here we deal with ordered sequences of connected blocks instead of unordered multisets of connected blocks. We first define this class of groups and show that their algebras are integral domains. We then show that in the oligomorphic case, they have a structure similar to that of shuffle algebras, and deduce that they are polynomial rings. With this information, we then look at some integer sequences which arise from this family of groups.

6.1 Wreath-A-like groups

If we have two finite sequences $S_1 = (x_1, \ldots, x_r)$ and $S_2 = (y_1, \ldots, y_s)$, then we write $S_1 \oplus S_2 = (x_1, \ldots, x_r, y_1, \ldots, y_s)$ for their concatenation.

Definition 6.1. We say that a permutation group G on Ω is *wreath-A-like* if there is a set of connected blocks $\{\Delta_i^{(j)}\}$ and a bijection ϕ from the set of orbits of G on finite subsets of Ω to the set of all finite sequences of connected blocks, with the bijection satisfying the following conditions:

(i) If $Y \subset \Omega$ is finite, then $\mathrm{wt}(\phi(Y)) = |Y|$.

(ii) If $Y \subset \Omega$ is finite and $\phi(Y) = (\Delta_{i_1}^{(j_1)}, \ldots, \Delta_{i_k}^{(j_k)})$, we can partition Y as an ordered union $Y = Y_1 \cup \cdots \cup Y_k$ with $|Y_l| = i_l$ for each l. Furthermore, if $Z \subseteq Y$ and $Z = Z_1 \cup \cdots \cup Z_k$, where $Z_l \subseteq Y_l$ for each l, then we can write $\phi(Z)$ as a concatenation of sequences $\phi(Z) = S_1 \oplus \cdots \oplus S_k$ where $\mathrm{wt}(S_l) = |Z_l|$ for each l, and $S_l = (\Delta_{i_l}^{(j_l)})$ if $Z_l = Y_l$.

As in the wreath-S-like case, condition (ii) implies that $\phi(Y_l) = (\Delta_{i_l}^{(j_l)})$ for $l = 1, 2, \ldots, k$.

Our prototypical family of groups for this class of groups are those of the form $G = H \mathrm{\ Wr\ } A$, where H is a permutation group on Δ, and A is the group of all order-preserving permutations of the rationals. Again, the wreath product action is the imprimitive one, so G acts on $\Omega = \Delta \times \mathbb{Q}$. As before, we take the connected blocks of weight n to be the orbits of the action of H on n-subsets of Δ. Then every orbit of G can be put into correspondence with a unique sequence of H-orbits as follows. If $Y \subset \Omega$ is an orbit representative, we can apply an element of the top group A to permute Y to a set of the form $(\Delta_1 \times \{1\}) \cup (\Delta_2 \times \{2\}) \cup \cdots \cup (\Delta_t \times \{t\})$, where each Δ_i is non-empty. Each of the Δ_i is a representative of some H-orbit, so we set $\phi(Y) = (\Delta_1, \Delta_2, \ldots, \Delta_t)$,

again blurring the distinction between orbits and orbit representatives. It is again easy to see that conditions (i) and (ii) of the definition hold in this case.

Another example is the automorphism group of the random tournament. In this context, a tournament is a complete graph, every one of whose edges is directed, and the random tournament is the Fraïssé limit of the set of finite tournaments. A tournament is called strongly connected if there is a path between every ordered pair of vertices. It can be shown quite easily that every tournament can be decomposed uniquely as a sequence of strongly connected components, where the edges between components are all from earlier components to later ones. So here we take our set of connected blocks to be the isomorphism classes of finite strongly connected tournaments (and again, the weight of a connected block is the number of vertices in it), and if T is a finite subset of the random tournament, we set $\phi(T)$ to be the sequence of strongly connected components of T. Again, it is not difficult to see that conditions (i) and (ii) hold. Also, as in the case of the random graph, it may be that a sub-tournament has more components that the original tournament; for example, the cyclically-oriented 3-cycle is strongly connected, but any 2-element subset of it consists of two strongly connected 1-sets.

A third example is the automorphism group of the "generic pair of total orders". This is the Fraïssé limit of the class of finite sets, where each finite set carries two (unrelated) total orders, which can be taken as $a_1 < a_2 < \cdots < a_n$ and $a_{\pi(1)} \prec a_{\pi(2)} \prec \cdots \prec a_{\pi(n)}$ for some permutation $\pi \in S_n$. Thus orbits of the Fraïssé limit are described by permutations. We can take the connected blocks for this group to be the permutations $\pi \in S_n$ for which there exists no k with $0 < k < n$ such that π maps $\{1, \ldots, k\}$ to itself. The details of this example are not hard to check.

Theorem 6.2. *If G is wreath-A-like, then $A(G)$ is an integral domain.*

Proof. The proof runs along very similar lines to that of Theorem 5.2. If α is a sequence of connected blocks, we write $[\alpha]$ to denote the multiset whose elements are the terms of the sequence with their multiplicities. We define an ordering on sequences by $\alpha < \beta$ if $[\alpha] <_{\text{lex}} [\beta]$ or $[\alpha] = [\beta]$ and $\alpha >_{\text{lex}} \beta$.

Again, we show that the conditions of Conjecture 4.4 are satisfied in this case. Let m and n be positive integers and let N be a positive integer with $N \geqslant 2(m+n)$. Denoting the inverse of ϕ by ψ and letting α run through all sequences of connected blocks of total weight $m+n$, we set $c_\alpha = \psi(\alpha)$ and let X_α be a N-set in the orbit $\psi(\alpha \oplus (\Delta_1^{(1)}, \ldots, \Delta_1^{(1)}))$, where the second sequence has $N-(m+n)$ copies of $\Delta_1^{(1)}$. We claim that this gives a Ramsey ordering of the orbits on $(m+n)$-sets, where the sequences are ordered as described in the previous paragraph. Firstly, every $(m+n)$-set orbit appears in the list by

hypothesis, as ψ is a bijection. Secondly, by construction, there is an $(m+n)$-subset of X_α in the orbit $\psi(\alpha)$, namely partition X_α as in condition (ii) of the definition, and remove all of the elements corresponding to the copies of $\Delta_1^{(1)}$ appended. This subset will then map to α under ϕ, by condition (ii).

To show the final condition of Ramsey orderings, we must show that any $(m+n)$-subset of X_α is in an orbit corresponding to a sequence less than or equal to α. Using the notation of condition (ii), we let $\alpha = (\Delta_{i_1}^{(j_1)}, \ldots, \Delta_{i_k}^{(j_k)})$ and $X_\alpha = X_1 \cup \cdots \cup X_k \cup X_{k+1} \cup \cdots \cup X_r$, where X_{k+1}, ..., X_r correspond to the appended copies of $\Delta_1^{(1)}$. Consider a subset $Y = Y_1 \cup \cdots \cup Y_r \subset X_\alpha$ with $|Y| = m+n$. If $Y_l \neq X_l$ for some l with $X_l \neq \Delta_1^{(1)}$, then clearly $[\phi(Y)] <_{\text{lex}} [\alpha]$, as $\text{wt}(S_l) < i_l$, and the only new connected blocks which can be used are copies of $\Delta_1^{(1)}$, which is the least connected block. So the remaining case to consider is where some of the $\Delta_{i_l}^{(j_l)}$ are equal to $\Delta_1^{(1)}$, and for some or all of those, $Y_l = \varnothing$, whereas $Y_s = X_s$ for some $s > k$. But in such a case, while we have $[\phi(Y)] = [\alpha]$, it is clear that $\phi(Y) \geqslant_{\text{lex}} \alpha$. So in either case, we have $\phi(Y) \leqslant \alpha$, or equivalently $Y \leqslant c_\alpha$, as required.

We note that the induced Ramsey orderings on m-set orbits and n-set orbits are given by precisely the same construction; in particular, the orbit given by the sequence β first appears in X_α, where $\alpha = \beta \oplus (\Delta_1^{(1)}, \ldots, \Delta_1^{(1)})$.

Finally, we must show that the remaining conditions of the conjecture are satisfied by this Ramsey ordering. We will only show that $\beta < \beta'$ implies $\beta \vee \gamma < \beta' \vee \gamma$; the other condition follows identically. We first deduce an explicit description of $\beta \vee \gamma$.

A *shuffle* of two sequences, say $(x_1, \ldots, x_r)$ and $(y_1, \ldots, y_s)$, is a sequence $(z_1, \ldots, z_{r+s})$ for which there is a partition of $\{1, 2, \ldots, r+s\}$ into two disjoint sequences $1 \leqslant i_1 < i_2 < \cdots < i_r \leqslant r+s$ and $1 \leqslant j_1 < j_2 < \cdots < j_s \leqslant r+s$ with $z_{i_k} = x_k$ for $1 \leqslant k \leqslant r$ and $z_{j_k} = y_k$ for $1 \leqslant k \leqslant s$.

We first show that $\beta \vee \gamma$ is the lexicographically greatest shuffle of β with γ; this is not difficult although the argument is a little intricate. We let α_0 be this greatest shuffle and note that $[\alpha_0] = [\beta] + [\gamma]$. Now let α be any sequence of connected blocks for which c_α contains a $d_\beta \cup e_\gamma$ decomposition; we must show that $\alpha_0 \leqslant \alpha$. (Here d_β and e_γ are the orbits on m-sets and n-sets corresponding to β and γ respectively.)

We let $\alpha = (A_1, \ldots, A_k)$ be this sequence of connected blocks, and let Y be a representative of the orbit c_α. Write Y as an ordered union $Y = Y_1 \cup \cdots \cup Y_k$ as in condition (ii) of the definition of wreath-A-like groups. Then any decomposition of c_α into two subsets can be written as

$$c_\alpha = Z \cup Z' = (Z_1 \cup \cdots \cup Z_k) \cup (Z_1' \cup \cdots \cup Z_k'),$$

where $Y_l = Z_l \cup Z'_l$ as a disjoint union for each l. Now if we require $\phi(Z) = \beta$ and $\phi(Z') = \gamma$, this means that the sequences $S_1 \oplus \cdots \oplus S_k$ and $S'_1 \oplus \cdots \oplus S'_k$ corresponding to Z and Z' respectively, as given by condition (ii), must equal β and γ respectively. If $\{Z_l, Z'_l\} = \{Y_l, \varnothing\}$, then $[S_l] + [S'_l] = [A_l]$ by condition (ii), but if not, then $[S_l] + [S'_l] <_{\text{lex}} [A_l]$ by comparing weights. As $M_1 <_{\text{lex}} M_2$ implies $M_1 + M <_{\text{lex}} M_2 + M$ for any multisets M_1, M_2 and M, it follows that $[\beta] + [\gamma] \leqslant_{\text{lex}} [\alpha]$ with equality if and only if $\{Z_l, Z'_l\} = \{Y'_l, \varnothing\}$ for each l, that is, $[\alpha_0] \leqslant_{\text{lex}} [\alpha]$ with equality if and only if α is a shuffle of β and γ. And if α is such a shuffle, then $\alpha \leqslant_{\text{lex}} \alpha_0$ by construction, so $\alpha_0 \leqslant \alpha$, as required.

Given this, we can now show that if $\beta < \beta'$, then $\beta \vee \gamma < \beta' \vee \gamma$. We first consider the case that $[\beta] <_{\text{lex}} [\beta']$, from which it follows that $[\beta] + [\gamma] <_{\text{lex}} [\beta'] + [\gamma]$. Since $[\beta \vee \gamma] = [\beta] + [\gamma]$ and $[\beta' \vee \gamma] = [\beta'] + [\gamma]$, we deduce that $[\beta \vee \gamma] <_{\text{lex}} [\beta' \vee \gamma]$, so $\beta \vee \gamma < \beta' \vee \gamma$.

Now consider the other possible case, namely $[\beta] = [\beta']$ but $\beta >_{\text{lex}} \beta'$. Note that $[\beta \vee \gamma] = [\beta' \vee \gamma]$ in this case, so we must show that $\beta \vee \gamma >_{\text{lex}} \beta' \vee \gamma$. We let $\beta = (\Delta_1, \ldots, \Delta_r)$, $\beta' = (\Delta'_1, \ldots, \Delta'_r)$ and $\gamma = (E_1, \ldots, E_s)$ in the following. We also let $\alpha = \beta \vee \gamma = (A_1, \ldots, A_{r+s})$ and $\alpha' = \beta' \vee \gamma = (A'_1, \ldots, A'_{r+s})$. Recalling that $\beta \vee \gamma$ is the lexicographically greatest shuffle of β and γ, we can construct $\beta \vee \gamma$ by using the following merge-sort algorithm (written in pseudo-code).

function *MergeSort*(β, γ)
  { *We have* $\beta = (\Delta_1, \ldots, \Delta_r)$ *and* $\gamma = (E_1, \ldots, E_s)$ }
  $i \leftarrow 1$
  $j \leftarrow 1$
  while $i \leqslant r$ **or** $j \leqslant s$ **do**
    if $(i > r)$ **then** { $A_{i+j-1} \leftarrow E_j$; $j \leftarrow j + 1$ }
    else if $(j > s)$ **then** { $A_{i+j-1} \leftarrow \Delta_i$; $i \leftarrow i + 1$ }
    else if $(E_j \geqslant \Delta_i)$ **then** { $A_{i+j-1} \leftarrow E_j$; $j \leftarrow j + 1$ }
    else { $A_{i+j-1} \leftarrow \Delta_i$; $i \leftarrow i + 1$ }
  od
  return $\alpha = (A_1, \ldots, A_{r+s})$

Observe what happens if we run the algorithm on the pairs (β, γ) and (β', γ). Assume that $\Delta_i = \Delta'_i$ for $i < i_0$, but that $\Delta_{i_0} > \Delta'_{i_0}$. Then they will run identically as long as $i < i_0$. When $i = i_0$, they will both continue taking terms from γ until $E_j < \Delta_{i_0}$ or γ is exhausted. Once this happens, the (β, γ) algorithm will take Δ_{i_0} next, so $A_{i_0+j-1} = \Delta_{i_0}$, but the (β', γ) algorithm will take $\max\{\Delta'_{i_0}, E_j\}$, so $A'_{i_0+j-1} = \max\{\Delta'_{i_0}, E_j\} < \Delta_{i_0} = A_{i_0+j-1}$. Thus we have $\beta \vee \gamma >_{\text{lex}} \beta' \vee \gamma$, so $\beta \vee \gamma < \beta' \vee \gamma$ as required.

It follows that $A(G)$ is an integral domain, as we wanted. □

6.2 Shuffle algebras

In the oligomorphic case, we can do better: the algebra $A(G)$ is actually a polynomial algebra if G is an oligomorphic wreath-A-like group. We show this by noting strong similarities between our algebra and standard shuffle algebras, and using well-known properties of shuffle algebras, in particular that the Lyndon words form a polynomial basis for the shuffle algebra.

We start by briefly recalling the key facts we will need. We take these results from Reutenauer's book on free Lie algebras [11]. The references to definitions, theorems and so forth are to his book.

Let T be an alphabet. Although Reutenauer sometimes assumes the alphabet to be finite, it will be clear that all of the results we use below work equally well in the infinite case: since words are always of finite length and we only ever work with finitely many words at once, we can always restrict attention to the finite subset of T containing the letters in use.

We write T^* for the set of words in the alphabet T. We write $K\langle T\rangle$ for the K-vector space with basis T^*. If we use the concatenation product (where the product of two words is just their concatenation), then this is the ring of non-commuting polynomials over T. But there is another product that we can define on words, and by extension on $K\langle T\rangle$, called the *shuffle product*. This is explained in section 1.4 of Reutenauer, and we now essentially quote parts of it.

Let $w = a_1 \cdots a_n$ be a word of length n in T^*, and let $I \subseteq \{1, \ldots, n\}$. We denote by $w|I$ the word $a_{i_1} \cdots a_{i_k}$ if $I = \{i_1 < i_2 < \cdots < i_k\}$; in particular, $w|I$ is the empty word if $I = \varnothing$. (Such a word $w|I$ called a *subword* of w.) Note that when

$$\{1, \ldots, n\} = \bigcup_{j=1}^{p} I_j,$$

then w is determined by the p words $w|I_j$ and the p subsets I_j.

Given two words u_1 and u_2 of respective lengths n_1 and n_2, their *shuffle product*, denoted by $u_1 \sqcup\!\sqcup u_2$, is the polynomial

$$u_1 \sqcup\!\sqcup u_2 = \sum w(I_1, I_2),$$

where the sum is taken over all pairs (I_1, I_2) of disjoint subsets of $\{1, \ldots, n\}$ with $I_1 \cup I_2 = \{1, \ldots, n\}$ and $|I_j| = n_j$ for $j = 1, 2$, and where the word $w = w(I_1, I_2)$ is defined by $w|I_j = u_j$ for $j = 1, 2$. Note that $u_1 \sqcup\!\sqcup u_2$ is a sum of words of length n, each with the same multiset of letters, and so is a homogeneous polynomial of degree n. Note also that the empty word, denoted by 1, is the identity for the shuffle product, that the shuffle product is commutative and associative, and that it is distributive with respect to addition. Thus $K\langle T\rangle$ with

the shuffle product is a commutative, associative algebra, called the *shuffle algebra*.

Using the associative and distributive properties of the shuffle product, we can also give an expression for the shuffle product of the words $u_1, \ldots, u_p$, of respective lengths $n_1, \ldots, n_p$; their shuffle product is the polynomial

$$u_1 \sqcup\!\sqcup \cdots \sqcup\!\sqcup u_p = \sum w(I_1, \ldots, I_p),$$

where now the sum is taken over all p-tuples $(I_1, \ldots, I_p)$ of pairwise disjoint subsets of $\{1, \ldots, n\}$ with $\bigcup_{i=1}^{p} I_j = \{1, \ldots, n\}$ and $|I_j| = n_j$ for each $j = 1, \ldots, p$, and where the word $w = w(I_1, \ldots, I_p)$ is defined by $w|I_j = u_j$ for each $j = 1, \ldots, p$.

A word appearing in the shuffle product $u_1 \sqcup\!\sqcup \cdots \sqcup\!\sqcup u_p$ is called a *shuffle* of $u_1, \ldots, u_p$. Note that this is consistent with the definition of shuffle we used in the proof of Theorem 6.2 above. As an example, if $a, b, c \in T$, then $ab \sqcup\!\sqcup ac = abac + 2aabc + 2aacb + acab$, and $aabc$ and $acab$ are both shuffles of ab and ac.

The next definition we need is that of a Lyndon word. Assume that our alphabet T is totally ordered. Then a *Lyndon word* in T^* is a non-empty word which is lexicographically smaller than all of its nontrivial proper right factors; in other words, w is a Lyndon word if $w \neq 1$ and if for each factorisation $w = uv$ (concatenation product) with $u, v \neq 1$, one has $w <_{\text{lex}} v$.

An alternative categorisation of Lyndon words is as follows (Corollary 7.7 in Reutenauer). Given a word $w = a_1 \cdots a_n$ of length n, we can define the rotation operator ρ by $\rho(w) = a_2 \cdots a_n a_1$. Then a word w of length $n \geqslant 1$ is Lyndon if and only if $w <_{\text{lex}} \rho^k(w)$ for $k = 1, \ldots, n-1$, which is to say that w is primitive (it does not have the form $w = u^r$ for some $r > 1$) and that it is lexicographically smaller than any rotation (cyclic permutation) of itself. It follows that Lyndon words are in bijective correspondence with primitive necklaces; see [11, Chap. 7] for more information.

A key property of Lyndon words is that every word $w \in T^*$ can be written *uniquely* as a decreasing product of Lyndon words, so $w = l_1^{r_1} \cdots l_k^{r_k}$, where $l_1 >_{\text{lex}} \cdots >_{\text{lex}} l_k$ and $r_1, \ldots, r_k \geqslant 1$. (This follows from Theorem 5.1 and Corollary 4.4, and can also easily be proved directly—see section 7.3.)

Finally, Theorem 6.1 states that the shuffle algebra $K\langle T\rangle$ is a polynomial algebra generated by the Lyndon words, and that for each word w, written as a decreasing product of Lyndon words $w = l_1^{r_1} \cdots l_k^{r_k}$ as in the previous paragraph,

one has

$$S(w) \stackrel{\text{def}}{=} \frac{1}{r_1! \cdots r_k!}\, l_1^{\sqcup\!\sqcup r_1} \sqcup\!\sqcup \cdots \sqcup\!\sqcup l_k^{\sqcup\!\sqcup r_k} = w + \sum_{\substack{[u]=[w] \\ u <_{\text{lex}} w}} \alpha_u u, \tag{7}$$

for some non-negative integers α_u, where $l^{\sqcup\!\sqcup r}$ means $l \sqcup\!\sqcup \cdots \sqcup\!\sqcup l$ with r terms in the product, and, in this context, $[u]$ means the multiset of letters in the word u.

Note that it is equation (7) which proves that $K\langle T\rangle$ is a polynomial algebra: the set T^* is a K-vector space basis for $K\langle T\rangle$, and given any finite multiset M of elements of T, the matrix relating the basis elements $\{\, w : w \in T^* \text{ and } [w] = M \,\}$ to $\{\, S(w) : w \in T^* \text{ and } [w] = M \,\}$ is unitriangular when the words are listed in lexicographic order, so that $\{\, S(w) : w \in T^* \,\}$ also forms a basis for $K\langle T\rangle$. This argument is true whether T is finite or infinite.

We can now apply this to our case of oligomorphic wreath-A-like permutation groups. Let G acting on Ω be such a group, as in Definition 6.1 above. We obviously take our alphabet T to be the set of connected blocks of the action (as given by the definition of wreath-A-like groups), so that T^* corresponds bijectively to the set of orbits of G on finite subsets of Ω. The alphabet T has the standard ordering defined on connected blocks, and the set T^* can then be ordered either by the lexicographic order (denoted $<_{\text{lex}}$) or by the order we defined at the start of Theorem 6.2 (denoted $<$).

Clearly $A(G)$ can be regarded as a K-vector space, with the set of characteristic functions of finite orbits as basis. We will identify the connected block sequence $w = (\Delta_{i_1}^{(j_1)}, \ldots, \Delta_{i_k}^{(j_k)})$ with the characteristic function of the corresponding orbit, writing w for both. Via this correspondence, we can identify $A(G)$ with $K\langle T\rangle$ as vector spaces. The grading on $A(G)$ induces a grading on $K\langle T\rangle$: the homogeneous component $V_n(G)$ is identified with the subspace of $K\langle T\rangle$ spanned by $\{\, w \in T^* : \mathrm{wt}(w) = n \,\}$. We then consider the product that the vector space $K\langle T\rangle$ inherits via this identification. Let $v \in T^*$ be another connected block sequence. We write $v \mathbin{\overline{\sqcup\!\sqcup}} w$ for the product in $A(G)$ and the induced product in $K\langle T\rangle$. The notation is designed to indicate that this product is related to the shuffle product, as we will see, and we call it the *complete shuffle product*. (It is also somewhat related to the infiltration product on $K\langle T\rangle$; see [11, sect. 6.3].) Recalling the definition of multiplication in $A(G)$, we see that for any finite subset $X \subset \Omega$ with $|X| = \mathrm{wt}(v) + \mathrm{wt}(w)$,

$$(v \mathbin{\overline{\sqcup\!\sqcup}} w)(X) = \sum_{\substack{Y \subseteq X \\ |Y| = \mathrm{wt}(v)}} v(Y) w(X \setminus Y).$$

But $v(Y)$ is none other than the characteristic function which has value 1 if

$\phi(Y) = v$ and 0 otherwise, and similarly for $w(Y \setminus X)$. So we have

$$(v \mathbin{\overline{\sqcup\!\sqcup}} w)(X) = |\{\, Y \subseteq X : \phi(Y) = v,\ \phi(X \setminus Y) = w \,\}|.$$

Thus, setting $u = \phi(X)$ and writing $u \to v \cup w$ if there is a $Y \subseteq X$ with $\phi(Y) = v$ and $\phi(X \setminus Y) = w$, we have

$$v \mathbin{\overline{\sqcup\!\sqcup}} w = \sum_{u \in T^*} \beta_u u,$$

where $\beta_u > 0$ if $u \to v \cup w$ and $\beta_u = 0$ otherwise.

Now we can characterise those u for which $u \to w \cup v$ quite easily. Firstly, consider the case that $[u] = [w] + [v]$, that is, the set of connected blocks of u is the same as those of w and v combined. Then $u \to w \cup v$ if and only if u is a shuffle of w and v, by condition (ii) of Definition 6.1, as in the proof of Theorem 6.2. In fact, the terms in $w \mathbin{\overline{\sqcup\!\sqcup}} v$ with $[u] = [w] + [v]$ will be precisely $w \sqcup\!\sqcup v$, which is easy to see. Now consider those terms with $[u] \neq [w] + [v]$. If $[u] <_{\mathrm{lex}} [w] + [v]$, then it is easy to see that we cannot have $u \to w \cup v$, but it may be possible otherwise. We deduce that our product is given by:

$$w \mathbin{\overline{\sqcup\!\sqcup}} v = w \sqcup\!\sqcup v + \sum_{\substack{\mathrm{wt}(u) = \mathrm{wt}(w) + \mathrm{wt}(v) \\ [u] > _{\mathrm{lex}} [w] + [v]}} \beta_u u \tag{8}$$

for some non-negative integers β_u.

Now given $w = l_1^{r_1} \cdots l_k^{r_k}$ written as a (concatenation) product of decreasing Lyndon words, we can consider the complete shuffle product as we did for the normal shuffle product above:

$$\begin{aligned} \bar{S}(w) &\stackrel{\mathrm{def}}{=} \frac{1}{r_1! \cdots r_k!}\, l_1^{\overline{\sqcup\!\sqcup} r_1} \mathbin{\overline{\sqcup\!\sqcup}} \cdots \mathbin{\overline{\sqcup\!\sqcup}} l_k^{\overline{\sqcup\!\sqcup} r_k} \\ &= \frac{1}{r_1! \cdots r_k!}\, l_1^{\sqcup\!\sqcup r_1} \sqcup\!\sqcup \cdots \sqcup\!\sqcup l_k^{\sqcup\!\sqcup r_k} + \sum_{\substack{\mathrm{wt}(u) = \mathrm{wt}(w) \\ [u] > _{\mathrm{lex}} [w]}} \beta_u u \\ &= w + \sum_{\substack{[u] = [w] \\ u <_{\mathrm{lex}} w}} \alpha_u u + \sum_{\substack{\mathrm{wt}(u) = \mathrm{wt}(w) \\ [u] > _{\mathrm{lex}} [w]}} \beta_u u \\ &= w + \sum_{\substack{\mathrm{wt}(u) = \mathrm{wt}(w) \\ u > w}} \alpha_u u, \end{aligned} \tag{9}$$

where the α_u and the β_u are non-negative integers. To get the second line, we have repeatedly used equation (8) to reduce the complete shuffle product to a normal shuffle product. Observe that $\mathrm{wt}(l_1^{r_1} \cdots l_k^{r_k}) = \mathrm{wt}(w)$, hence the sum is

over words with $\mathrm{wt}(u) = \mathrm{wt}(w)$, and with $[u] >_{\mathrm{lex}} [w]$, since $>_{\mathrm{lex}}$ is transitive and $[u_1] >_{\mathrm{lex}} [u_2]$ implies $[u_1] + [u] >_{\mathrm{lex}} [u_2] + [u]$ for any word u. That the β_u are non-negative is easy to see, and it is not that much harder to see that they are integral, although we do not need this. In the third line, we have used equation (7), and in the last line, we have set $\alpha_u = \beta_u$ in the case that $[u] >_{\mathrm{lex}} [w]$, and used the relation on words (sequences) defined in the previous section, namely $u > w$ if $[u] >_{\mathrm{lex}} [w]$ or $[u] = [w]$ and $u <_{\mathrm{lex}} w$.

It is also important to note that in our case, the set $\{ u : \mathrm{wt}(u) = \mathrm{wt}(w) \}$ is finite, as there are only finitely many connected blocks of each weight, the same number as the number of orbits on sets of size $\mathrm{wt}(w)$, so that the sums in equation (9) are all finite.

We now see, as above, that the matrix relating $\{ w : w \in T^* \text{ and } \mathrm{wt}(w) = n \}$ to $\{ \bar{S}(w) : w \in T^* \text{ and } \mathrm{wt}(w) = n \}$ is unitriangular when the words of weight n are listed in the order we have defined. It follows that the $\bar{S}(w)$ form a vector space basis for $A(G) = K\langle T \rangle$, and hence the set of Lyndon words is a set of polynomial generators for $A(G)$. We summarise these results as a theorem.

Theorem 6.3. *If G is an oligomorphic wreath-A-like permutation group, then $A(G)$ is a polynomial ring, and the generators are those characteristic functions on orbits corresponding to Lyndon words as described above.* □

We can now deduce:

Corollary 6.4. *If G is an oligomorphic wreath-A-like permutation group, then the element $\varepsilon \in V_1(G)$ is prime in $A(G)$.*

Proof. We have $e = \Delta_1^{(1)} + \cdots + \Delta_1^{(r)}$, where the $\Delta_1^{(j)}$ are the orbits on 1-sets. As each of the $\Delta_1^{(j)}$ is a Lyndon word, $A(G) = K[\Delta_1^{(1)}, \ldots, \Delta_1^{(r)}, \Delta_2^{(1)}, \ldots]$. It follows that we can replace the polynomial generator $\Delta_1^{(1)}$ by ε (as they are linearly related), giving $A(G) = K[\varepsilon, \Delta_1^{(2)}, \ldots, \Delta_1^{(r)}, \Delta_2^{(1)}, \ldots]$. It is clear, since we then have $A(G)/(\varepsilon) \cong K[\Delta_1^{(2)}, \ldots, \Delta_1^{(r)}, \Delta_2^{(1)}, \ldots]$, that $A(G)/(\varepsilon)$ is an integral domain, so ε is prime in $A(G)$. □

6.3 Integer sequences, necklaces and free Lie algebras

Theorem 6.3 leads us to revisit some counting questions. Cameron [5] considered the following question. If the algebra $A(G)$ corresponding to an "interesting" oligomorphic group G were polynomial, what would be the sequence counting the number of polynomial generators of each degree? From knowledge of the dimension of each homogeneous component of $A(G)$, the answer can be determined using the inverse Euler transform. Now that we have an explicit description of the polynomial generators in the wreath-A-like case, an examination of the

sequences observed might yield some interesting new information about those sequences.

The two sequences we will consider are those arising from the groups S_2 Wr A and A Wr A, both of which appear in the On-Line Encyclopedia of Integer Sequences [12]. There are some obvious generalisations to other groups, as we observe below. The n-th homogeneous component of the group S_2 Wr A has dimension F_{n+1} (a Fibonacci number, where $F_0 = 0$ and $F_1 = 1$), and so the sequence counting the number of generators of degree n is A006206, beginning 1, 1, 1, 1, 2, 4, 5, 8, 11, 18, ... By our result, the n-th term of this sequence gives the number of Lyndon words of weight n (starting with $n = 1$) in the alphabet $T = \{\Delta_1, \Delta_2\}$, where Δ_1 and Δ_2 have respective weights 1 and 2.

Similarly, for the group A Wr A, the n-th homogeneous component has dimension 2^{n-1} for $n \geqslant 1$, and the sequence counting the number of generators of degree n is A059966, beginning 1, 1, 2, 3, 6, 9, 18, 30, ... (Note that the paper quoted above had sequence A001037 by mistake, this being the inverse Euler transform of the closely related sequence (2^n).) This sequence then counts the number of Lyndon words of weight n in the alphabet $T = \{\Delta_1, \Delta_2, \dots\}$, where Δ_i has weight i.

The Encyclopedia entry gives a different explanation, however: this sequence lists the dimensions of the homogeneous components of the free Lie algebra with one generator of each degree 1, 2, 3, etc. The connection between these two descriptions of this sequence is easy to describe, using [11, Thm. 4.9]. Let T be an alphabet whose letters each have a positive integral degree/weight (we use these terms interchangeably in this section), and where there are only finitely many letters of each possible weight. There is a basis of the free Lie algebra on the alphabet T (viewed as a vector space) given by $\{ P_w : w \in T^* \text{ Lyndon} \}$, where $P_a = a$ if $a \in T$, and $P_w = [P_u, P_v]$ otherwise, where $w = uv$ with v being the lexicographically smallest nontrivial proper right factor of w (see [11, Thm. 5.1]). Note that it trivially follows by induction that the degree of the homogeneous polynomial P_w is $\mathrm{wt}(w)$. Thus the dimension of the homogeneous component of degree n of the free Lie algebra on the alphabet T is the number of Lyndon words in T^* of weight n. It follows that we can also describe the two sequences above as either the number of Lyndon words of weight n in the alphabets $\{\Delta_1, \Delta_2\}$ and $\{\Delta_1, \Delta_2, \dots\}$ respectively, or as the number of primitive necklaces of weight n in these symbols, or as the dimension of the homogeneous component of degree n of the free Lie algebras on these sets. This obviously generalises to other wreath-A-like groups.

We may ask other counting questions based on these ideas. We start with an alphabet of weighted letters T (again with only finitely many letters of each

weight). The primary questions arising are how to transform between the three sequences:

$$\begin{aligned} a_n &= \text{number of letters of weight } n \text{ in } T, \\ w_n &= \text{number of words of weight } n \text{ in } T^*, \\ l_n &= \text{number of Lyndon words of weight } n \text{ in } T^*. \end{aligned}$$

(Of course, l_n can also be regarded as the number of primitive necklaces of weight n in this alphabet.) In our context, a_n is the number of connected blocks of weight n in our wreath-A-like group, w_n gives the dimension of the homogeneous component of weight n in $A(G)$ and l_n gives the number of polynomial generators of weight n in $A(G)$. We use the notation and some of the ideas presented in Bernstein and Sloane's paper on integer sequences [1].

The transformation between (a_n) and (w_n) can be effected by INVERT, as every word is an ordered sequence of letters:

$$1 + \sum_{n=1}^{\infty} w_n x^n = \frac{1}{1 - \sum_{n=1}^{\infty} a_n x^n}.$$

The transformation between (w_n) and (l_n) is performed using EULER, as every word is a product of a decreasing sequence of Lyndon words, so can be identified with a multiset of Lyndon words:

$$1 + \sum_{n=1}^{\infty} w_n x^n = \prod_{n=1}^{\infty} \frac{1}{(1 - x^n)^{l_n}}.$$

It follows that we can transform between (a_n) and (l_n) using a variant of WEIGH:

$$1 - \sum_{n=1}^{\infty} a_n x^n = \prod_{n=1}^{\infty} (1 - x^n)^{l_n}. \tag{10}$$

Most of the six possible conversions between (a_n), (w_n) and (l_n) are straightforward given these formulæ; the two which are harder are converting (w_n) and (a_n) to (l_n). Inverting the EULER transform is explained in [1]; we apply the same idea to convert from (a_n) to (l_n).

Given a sequence (a_n), we introduce the auxiliary sequence (c_n) defined by the equation $1 - \sum_{n=1}^{\infty} a_n x^n = \exp\bigl(-\sum_{n=1}^{\infty} c_n x^n/n\bigr)$. Using the generating functions $A(x) = \sum_{n=1}^{\infty} a_n x^n$ and $C(x) = \sum_{n=1}^{\infty} c_n x^n$, we can perform standard manipulations using the defining equation for (c_n) to deduce that $C(x) =$

$xA'(x) + C(x)A(x)$. It follows that

$$c_n = na_n + \sum_{k=1}^{n-1} c_k a_{n-k}. \tag{11}$$

Now substituting $\exp\bigl(-\sum c_n x^n/n\bigr)$ for $1 - \sum a_n x^n$ in equation (10), taking logarithms and expanding as a power series gives the coefficient of x^n/n to be $c_n = \sum_{d|n} d\, l_d$. Finally, Möbius inversion gives

$$l_n = \frac{1}{n} \sum_{d|n} \mu(n/d) c_d. \tag{12}$$

Thus we have an effective way of calculating the number of Lyndon words of a given weight given the number of letters of each possible weight.

As an interesting example of this process, let us consider our favourite group, $G = S_2$ Wr A. In this case, recall that we have $T = \{\Delta_1, \Delta_2\}$, so $a_1 = a_2 = 1$ and $a_n = 0$ for $n \geqslant 3$. Then the sequence (c_n) is calculated by equation (11): we have $c_1 = 1$ and $c_2 = 3$. For $n \geqslant 3$, we have $c_n = c_{n-1} + c_{n-2}$, so (c_n) is the standard Lucas sequence (L_n): 1, 3, 4, 7, 11, 18, ... We can now calculate the sequence (l_n): the first few terms are as we predicted: 1, 1, 1, 1, 2, 2, 4, 5, ..., and a general formula is $l_n = \frac{1}{n} \sum_{d|n} \mu(n/d) L_d$, as is given in the Encyclopedia entry for A006206. One interesting thing to observe is that if p is prime, then we have $l_p = (\mu(1)L_p + \mu(p)L_1)/p = (L_p - 1)/p$. It follows that the Lucas sequence satisfies $L_p \equiv 1 \pmod{p}$ for all primes p, a known result (see Hoggart and Bicknell [7]), but somewhat surprising in this context.

The description of our sequence A006206 in the Encylcopedia is "aperiodic binary necklaces [of length n] with no subsequence 00, excluding the sequence '0'." Our description is that it counts primitive necklaces of weight n in the alphabet $\{\Delta_1, \Delta_2\}$. These are easily seen to be equivalent: if we replace every Δ_1 by the symbol 1 and every Δ_2 by the symbols 10 (in clockwise order, say), then we will get a primitive (aperiodic) binary necklace with no subsequence 00 whose length equals the weight of the necklace we started with, and we can perform the inverse transformation equally simply (as we are excluding the necklace 0). We can do the same with the group S_n Wr A, enabling us to count the number of primitive binary necklaces of length n with no subsequence $00\cdots0$ (with n zeros) and excluding the necklace 0.

Now let us apply these ideas to the case $G = A$ Wr A. Firstly, the auxilary sequence turns out to be $c_n = 2^n - 1$, and the sequence (l_n) is given by $l_n = \sum_{d|n} \mu(n/d)(2^d - 1)$. This can be simplified using the result $\sum_{d|n} \mu(n/d) = [n = 1]$, where we are using Iverson's convention that if P is a predicate, then

$[P] = 1$ if P is true and 0 otherwise. So we have $l_n = \sum_{d|n} \mu(n/d)2^d - [n = 1]$. The sequence given by $\sum_{d|n} \mu(n/d)2^d$ is sequence A001037, and so our sequence differs from it by 1 in the $n = 1$ term only, yielding the observed sequence A059966. We can also give a necklace description of this sequence as above: it is the number of primitive binary necklaces of length n excluding the necklace 0—the sequence A001037 is essentially the same, but does not exclude the necklace 0, so it it also counts the number of binary Lyndon words of *length* n. (These are the descriptions of this sequence given in the Encyclopedia.) Finally, as above, if we consider the term l_p for p prime, we see that $l_p = ((2^p - 1) - 1)/p = 2(2^{p-1} - 1)/p$, so for $p > 2$, we deduce Fermat's little theorem for base 2, that is $2^{p-1} \equiv 1 \pmod{p}$.

An investigation of those sequences of non-negative integers (b_n) for which $\frac{1}{n}\sum_{d|n} \mu(n/d)b_d$ is a non-negative integer for all n has been undertaken by Puri and Ward [10], who call them exactly realizable. We can thus add to their work a class of exactly realizable sequences: those which are of the form (c_n), where (c_n) is given by equation (11) for some sequence of non-negative integers (a_n). A particular family of such sequences is given by $a_i = 1$ for $1 \leqslant i \leqslant n$ and $a_i = 0$ for $i > n$; these are sometimes known as "generalised Fibonacci sequences", and have been discussed by Du [6] (where this sequence is called ϕ_n). It would be interesting to know whether new congruence identities can be discovered by applying this technique to some of the sequences identified there or to sequences produced by other wreath-A-like groups.

7 Non-oligomorphic groups

Throughout this part of the thesis, we have mostly focused on oligomorphic groups, proving results in general where there was no problem in doing so. In this final section, we consider briefly the issues arising in the non-oligomorphic case.

As has already been pointed out above, the group $\mathbb{Z}$ acting regularly on $\mathbb{Z}$ does not have a Ramsey ordering on 2-sets, so much of what we did above will not help us to understand the algebra $A(\mathbb{Z})$. It is easy to construct other similar examples.

A more difficult question is whether we have even got the "right" definition of the algebra $A(G)$ in the non-oligomorphic case. The definition we have been using was introduced specifically to study the behaviour of oligomorphic groups. There are two finiteness conditions which can be imposed on the algebra we consider.

Firstly, we have taken the direct sum $A(G) = \bigoplus_{n=0}^{\infty} V_n(G)$, which is the direct limit as $N \to \infty$ of the vector spaces $\bigoplus_{n=0}^{N} V_n(G)$ (with the obvious direct maps). We could have instead taken the cartesian sum $\sum_{n=0}^{\infty} V_n(G)$, being the inverse limit of the same family of vector spaces (with the obvious inverse maps).

Secondly, and independently of the first choice, we could either take $V_n(G)$ to be the vector space of all functions from n-subsets of Ω to K which are fixed by G, as we have until now, or we could take it to be the subspace of this consisting of those functions which assume only finitely many distinct values on n-sets. (The latter idea was suggested to me by Peter Cameron.) Note, though, that if there are infinitely many orbits on n-sets, this vector space will still have uncountable dimension. It is not hard to check that if we use the latter definition, the multiplication in the algebra is still well-defined. Also, this distinction does not exist in the oligomorphic case. (Another seemingly plausible choice, those functions in $V_n(G)$ which are non-zero on only finitely many orbits of G, can fail to produce a well-defined multiplication: consider, for example, the case of e^2 with our favourite non-oligomorphic group, $\mathbb{Z}$: it takes the value 2 on every 2-set.)

Thus we have four plausible algebras to choose from, and it is not clear which is the "correct" one to use. More work is still required in this area.

References

[1] M. Bernstein and N. J. A. Sloane, *Some canonical sequences of integers*, Linear Algebra and Applications **226/228** (1995), 57–72.

[2] Peter J. Cameron, *Orbits of permutation groups on unordered sets, II*, J. London Math. Soc. (2) **23** (1981), 249–264.

[3] ———, *Oligomorphic Permutation Groups*, Cambridge University Press, 1990.

[4] ———, *The algebra of an age*, Model Theory of Groups and Automorphism Groups (David M. Evans, ed.), Cambridge University Press, 1997, pp. 126–133.

[5] ———, *Sequences realized by oligomorphic permutation groups*, Journal of Integer Sequences **3** (2000), Article 00.1.5.

[6] B.-S. Du, *A simple method which generates infinitely many congruence identities*, Fibonacci Quart. **27** (1989), 116–124.

[7] V. E. Hoggart Jr. and M. Bicknell, *Some congruences of the fibonacci numbers modulo a prime p*, Math. Mag. **47** (1974), 210–214.

[8] William M. Kantor, *On incidence matrices of finite projective and affine spaces*, Mat Z. **124** (1972), 315–318.

[9] Peter M. Neumann, *The structure of finitary permutation groups*, Arch. Math. (Basel) **27** (1976), 3–17.

[10] Yash Puri and Thomas Ward, *Arithmetic and growth of periodic orbits*, Journal of Integer Sequences **4** (2001), Article 01.2.1.

[11] Christophe Reutenauer, *Free Lie Algebras*, Oxford University Press, 1993.

[12] N. J. A. Sloane, *The On-Line Encyclopedia of Integer Sequences*, available at `http://www.research.att.com/~njas/sequences/`, 2001.

Part II

Parking Functions, Valet Functions and Priority Queues

In which we present a new bijection between parking functions and priority queues, extend it to a bijection between valet functions and priority queues on multisets, and learn of the standard of driving in Boston

This is an edited version of the published paper: Julian D. Gilbey and Louis H. Kalikow, *Parking functions, valet functions and priority queues*, Discrete Mathematics **197/198** (1999), 351–373.

1 Introduction

The combinatorial properties of parking functions have attracted interest for some time. Much is known about these functions and how they relate to other combinatorial structures such as trees. Recently, allowable pairs of permutations of a priority queue have also been studied. In this part of the thesis, we study these two classes of objects, which turn out to be closely related.

After introducing some notation in section 2, we define parking functions in section 3, together with some background material on the subject. The same is done for priority queues and allowable pairs in section 4. In section 5 we define the notion of a "breakpoint" for both parking functions and allowable pairs, and in section 6 we present a new bijection between these objects. In section 7 we introduce valet functions, which turn out to correspond to allowable pairs of permutations of a multiset. We also note an interesting bijection between valet functions and k-way trees, which restricts to a new bijection between parking functions and labelled trees. In section 8 we present, with detailed proof, a bijection between valet functions and allowable pairs for a multiset. This bijection, of which our first bijection is a special case, has the property of being both output and breakpoint preserving. (The output of the various objects involved is defined in sections 3, 4 and 7.) In section 9 we give an alternative description of the bijection of section 6, and use this in section 10 to give an interpretation for allowable pairs of the inversion enumerator for trees, showing that our bijection preserves this too in a suitable sense. We conclude in section 11 by comparing our bijection to other bijections involving parking functions and priority queues.

2 Notation

We write $[n]$ for $\{1, 2, \ldots, n\}$ and $[n]_0$ for $[n] \cup \{0\}$, with the convention that $[0] = \varnothing$. We will often think of a function $p : [n] \to [n]$ as the sequence of its values $p(1), \ldots, p(n)$ (sometimes even omitting the commas). Similarly, we regard a permutation $\sigma \in S_n$ as either a bijection $\sigma : [n] \to [n]$ or as a sequence $\sigma_1\sigma_2 \ldots \sigma_n$ (writing σ_i for $\sigma(i)$). If (a_i) is a sequence, we will also use $\mathbf{a}$ to refer to it. In the case that each $a_i \in \mathbb{N}$, we write $M_{\mathbf{a}}$ for the multiset $\{1^{a_1}, 2^{a_2}, \ldots\}$. Note that 0 is considered a natural number! (Also, we are using standard set notation for multisets, which is different from that used in the first part of this thesis; this turns out to be more convenient here.)

From section 7 onwards, we will be considering functions $p : [k] \to \mathcal{P}([n])$ with $|p(i)| = a_i$ for each i, where $\mathcal{P}(\Omega)$ denotes the power set of Ω. We will list the elements of each $p(i)$ as $p_i(1)$, $p_i(2)$, $\ldots$, $p_i(a_i)$. The order in which we do so

turns out to be unimportant for our results, as we will show below, so without loss of generality, we will assume that they are listed in increasing order, unless stated otherwise. We will also write $p_1(1), \ldots, p_i(j-1)$ as shorthand for the initial subsequence of $p_1(1), \ldots, p_1(a_1), p_2(1), \ldots, p_k(a_k)$ up to, but excluding, $p_i(j)$, even in the case that $j = 1$.

3 Parking functions and major functions

Consider a one-way street with n empty parking spaces in a row. There are n drivers who wish to park in these spaces and they arrive one at a time. Each driver has a preferred parking space, to which she drives. If it is empty, she parks there, but if not, she parks in the next available parking space if there is one. If, however, the rest of the spaces are occupied, she leaves without parking. If all of the cars are able to park, we call the sequence of preferred positions a parking function.

Formalising this description, we define a *parking function* to be a function $p : [n] \to [n]$ for which the following algorithmic function returns `TRUE`, and we write P_n for the set of all parking functions on $[n]$. Our arrays are indexed starting at 1, and we follow the convention that the body of a loop headed by a condition such as "**for** $i := 1$ **to** 0 **do**" is never executed.

function *TestParking*(p, n)
  $L :=$ empty array of length $n + 1$
  for $i := 1$ **to** n **do**
    $l_0 := \min\{\, l : l \geqslant p(i)$ and $L[l]$ is empty $\}$
    if $l_0 = n + 1$ **then**
      return `FALSE`
    else
      $L[l_0] := i$
    fi
  od
  return `TRUE`

If the algorithm returns `TRUE`, we write $\pi_n(p)$ for the resulting permutation $(L[1], L[2], \ldots, L[n])$ of $[n]$, calling it the *output* of p. We note that if p is a parking function and $\tau = \pi_n(p)$, then $\tau^{-1}(i) \geqslant p(i)$ for each i.

Parking functions were introduced in computer science and combinatorics by Konheim and Weiss [12, Sec. 6] as a colourful way to study a hashing problem. (However, the original scenario used would no longer be considered politically correct!) In their paper, they proved that the number of parking functions

on $[n]$ is $(n+1)^{n-1}$. Further proofs of this fact followed, including a beautiful one by Pollak (see Riordan [18, Sec. 2] and Foata and Riordan [5, Sec. 2]), a simple extension of which is used to prove Theorem 7.2 below. (An alternative description of Pollak's proof in group-theoretic terms is given by Stanley in [22, Sec. 2].) Knuth [11, Sec. 6.4] surveys the results about parking functions known to computer science in the early 1970s. (His description is given in terms of a hashing algorithm, but see also exercises 6.4–29 through 6.4–31 in which the parking function description is presented.)

It was recently pointed out to me by Joseph Kung that parking functions have been known to statisticians for a long time, in the context of order statistics. This also leads to a natural generalisation of parking functions. For more information, see Kung and Yan [14].

Several bijections between parking functions on $[n]$ and other sets of combinatorial structures are known. The first published bijection between parking functions and acyclic functions on $[n]$ (which are trivially representable by labelled trees on $[n]_0$) was by Schützenberger [20]. Kreweras [13, Sec. 6] gives a bijection that maps labelled trees with k inversions to parking functions with k probes. (Inversions are described in section 10 below. Note also that our parking functions correspond to Kreweras's *suites majeures* under the bijection $p(i) \mapsto (n+1) - p(i)$.) Moszkowski [16, Sec. 3] gives another bijection, in which a node of the tree with i children corresponds to a parking space in which i cars prefer to park. Pollak (see Riordan [18, Secs. 3 and 4] and Foata and Riordan [5, Sec. 2]) also gives a bijection in which a parking function is associated with a code which, by Prüfer's correspondence, corresponds to a tree. Foata and Riordan [5] also present another bijection from parking functions on $[n]$ to acyclic functions on $[n]$, and Françon [6] has shown how their result may be generalised to a much larger class of selection procedures. Knuth [11, answer to exercise 6.4–31] describes two bijections which are based on those of Foata and Riordan [5] and Kreweras [13], but are in fact different from them. Finally, in section 7 below, we describe a bijection which satisfies the property described in part (a) of Knuth's answer. In addition to these, Stanley [21, 22, 23] has studied how parking functions relate to noncrossing partitions and to hyperplane arrangements. Parking functions have also proved to have interesting algebraic properties. See, for example, the series of conjectures concerning diagonal invariants and parking functions presented by Haiman [10].

We define a *probe* to be an attempt by a car to park in an already occupied space, and the number of probes of a parking function is the total number of probes made by all of the cars. For example, if a car prefers space 3, but parks in space 6 (because spaces 3, 4 and 5 were full), the car makes three probes. In the language of the algorithm *TestParking*, the number of probes of car i is

$l_0 - p(i)$. So, letting $\tau = \pi_n(p)$ as before, car i makes $\tau^{-1}(i) - p(i)$ probes. Thus the number of probes of p is given by

$$\begin{aligned}\sum_{i=1}^{n}(\tau^{-1}(i) - p(i)) &= \sum_{i=1}^{n}\tau^{-1}(i) - \sum_{i=1}^{n}p(i)\\ &= \sum_{j=1}^{n} j - \sum_{i=1}^{n}p(i)\\ &= \tfrac{1}{2}n(n+1) - \sum_{i=1}^{n}p(i),\end{aligned}$$

as τ is a permuation of $[n]$. In particular, the number of probes of a parking function $p \in P_n$ depends only upon the values which the function takes, not the order in which it takes them. This was known already; see for example Peterson [17, page 137] or Gessel and Sagan [7, Sec. 7].

In Konheim and Weiss [12, Sec. 3], an expression is also obtained for the size of the set $S(\tau) = \{\, p \in P_n : \pi_n(p) = \tau \,\}$, defined for any $\tau \in S_n$. Given τ, set $\tau(0) = n + 1$ and define

$$b_\tau(i) = \max\{\, j \in [i-1]_0 : \tau(j) > \tau(i) \,\}.$$

Then we have

$$|S(\tau)| = \prod_{i=1}^{n}(i - b_\tau(i)). \tag{1}$$

To see why this is true, notice that if car m parks in space j, its preferred space could have been any space numbered $i \leqslant j$ as long as the spaces i, $i+1$, …, $j-1$ were occupied before it attempted to park. This will be the case if and only if all of these spaces are occupied by cars numbered less than m.

We finish this section with a very important alternative characterisation of parking functions. We call a function $f : [n] \to [n]$ a *major function* if it satisfies the property

$$|\{\, i : f(i) \leqslant m \,\}| \geqslant m \qquad \text{for } m = 1, 2, \ldots, n.$$

Lemma 3.1. *p is a parking function if and only if p is a major function.*

This is the special case of Lemma 7.3 below with $\mathbf{a} = (1, 1, \ldots, 1)$. It is also straightforward to prove directly, and has been known for a long time. A direct proof can be found in Gessel and Sagan [7, Theorem 7.2].

4 Priority queues and allowable pairs

We follow Atkinson and Thiyagarajah [3] for the following definitions. A *priority queue* is an abstract data type supporting the operations INSERT and DELETEMIN. There is an input data stream $\sigma = \sigma_1\sigma_2\ldots$ and an output data stream $\tau = \tau_1\tau_2\ldots$, where the σ_i are (possibly repeated) elements of a totally ordered set. Each INSERT operation will insert the next element of σ into the queue, and each DELETEMIN operation will remove a minimal element of the queue, placing it in the output stream. We only allow a DELETEMIN operation when the queue is non-empty.

We restrict ourselves throughout this thesis to the case where σ is finite; it follows that τ is also. If σ has length n, then an allowable sequence of n INSERT's and n DELETEMIN's (that is, one with the property that any initial subsequence contains at least as many INSERT's as DELETEMIN's) will be called a *priority queue computation*. If σ is the input and τ is the output of some priority queue computation, we call (σ, τ) an *allowable pair*. We write Q_n for the set of allowable pairs on $[n]$, by which we mean those allowable pairs (σ, τ) for which $\sigma, \tau \in S_n$. The following algorithm from Atkinson and Beals [1] takes as input a pair (σ, τ) of data streams of length n and tests whether it is an allowable pair. (The use of the notation INSERT(σ_j) rather than simply INSERT is for convenience, as it allows us to refer to our location in σ.)

```
function TestPair((σ, τ), n)
    Q := empty priority queue
    i := 1
    for j := 1 to n do
(*)     while τ_j ∉ Q do
            INSERT(σ_i)
            i := i + 1
        od
        if τ_j ≠ min(Q) then
            return FALSE
        else
            DELETEMIN
        fi
    od
    return TRUE
```

If (σ, τ) is an allowable pair, then τ is called the *output* of (σ, τ), and the priority queue computation executed by this algorithm is called the *natural*

computation for (σ, τ).

Many properties of Q_n are known. Atkinson and Thiyagarajah [3, Thm. 1] found that the number of allowable pairs on $[n]$ is $(n+1)^{n-1}$. Moreover, they show in [3, Lemma 5] that the number of allowable pairs (σ, τ) having a given permutation $\tau \in S_n$ as output is given by the same expression as that which counts the number of parking functions having τ as output ($|S(\tau)|$ in equation (1) above). This suggests the possible existence of an interesting bijection between parking functions and allowable pairs on $[n]$ which is output preserving.

As with parking functions, bijections have been found between allowable pairs on $[n]$ and labelled trees on $n+1$ vertices. Atkinson and Beals [1, Sec. 3] define such a bijection inductively, and Gessel and Wang [8] give algorithms for this bijection. A variant of their bijection can be obtained by letting their $\gamma_{(i,m)}$ denote the result of inserting m within γ *before* the symbol i, where i is given the name "root" if m is inserted at the end of γ. A different bijection, also defined by induction, is given by Golin and Zaks [9]. This too has a variant obtained by connecting $*$ in $T_{\pi\to\sigma}$ to the *predecessor* of $\max(\pi_i)$ in π_i.

Atkinson, Linton and Walker [2] generalised the work on allowable pairs by permitting the input and output data streams to be permutations of a multiset $M_{\mathbf{a}} = \{1^{a_1}, 2^{a_2}, \ldots, k^{a_k}\}$. They found that the number of allowable pairs in this case is $\frac{1}{n+1}\prod_{i=1}^{k}\binom{n+1}{a_i}$. This was calculated by constructing a bijection between the allowable pairs and certain k-way trees. (A k-way tree is either an empty tree or a root node with a sequence of k k-way subtrees.) Their bijection, again defined by induction, is a natural extension of the bijection in Atkinson and Beals [1] for allowable pairs on $[n]$. We provide a corresponding extension of parking functions below (sections 7ff), and deduce an alternative way of counting these pairs.

5 Breakpoints

We now define a parallel concept for both functions $[n] \to [n]$ and pairs of permutations of $[n]$, which turns out to be invariant under our bijection and is crucial to our method of proof.

Let $p : [n] \to [n]$. We say that $b \in [n]_0$ is a *breakpoint* of p if we have $|\{\, i : p(i) \leqslant b \,\}| = b$. It is easily checked that in the case $p \in P_n$, this condition is equivalent to $\{L[1], \ldots, L[b]\} = \{\, i : p(i) \leqslant b \,\}$. (We always have $\{L[1], \ldots, L[b]\} \subseteq \{\, i : p(i) \leqslant b \,\}$; then consider the sizes of the sets.) In the car drivers description, this says that every driver who wishes to park in one of the first b spaces succeeds in doing so.

Now let $(\sigma, \tau) \in S_n \times S_n$. We say that $b \in [n]_0$ is a *breakpoint* of (σ, τ) if

$\{\sigma_1, \ldots, \sigma_b\} = \{\tau_1, \ldots, \tau_b\}$. In the case $(\sigma, \tau) \in Q_n$, this is equivalent to saying that, with the natural computation, the queue is empty after outputting τ_b. (This follows, for if b is a breakpoint, then once the first b elements of σ have been read into the queue, the body of the **while** loop in the *TestPair* algorithm will not be executed again until τ_b has been output. The converse is trivial.)

It is clear that 0 and n are always breakpoints of any $p \in P_n$ and any $(\sigma, \tau) \in Q_n$. The following lemma shows that at least one other breakpoint often exists in such cases.

Lemma 5.1. (a) *Let $p \in P_n$, $t = \pi_n(p)$ and $d = t^{-1}(n)$. Then d is a breakpoint of p.*

(b) *Let $(\sigma, \tau) \in Q_n$ and $\delta = \tau^{-1}(n)$. Then δ is a breakpoint of (σ, τ).*

This follows as a special case of Corollary 8.2 when $\mathbf{a} = (1, 1, \ldots, 1)$. It is also very straightforward to prove directly.

6 The bijection between parking functions and allowable pairs

We define functions $\phi_n : P_n \to Q_n$ and $\psi_n : Q_n \to P_n$ inductively. For $n = 0$, the functions are trivial, as the sets have only one element.

For $n \geqslant 1$, given $p \in P_n$, we define $(s, t) = \phi_n(p)$ as follows:

(ϕ1) Set $t = \pi_n(p)$ and $d = t^{-1}(n)$.

(ϕ2) Define $p' \in P_{n-1}$ by setting, for $i < n$,

$$p'(i) = \begin{cases} p(i) - 1 & \text{if } p(i) > d, \\ p(i) & \text{otherwise.} \end{cases}$$

(ϕ3) Set $(s', t') = \phi_{n-1}(p')$.

(ϕ4) We define s by inserting n into the $p(n)$-th position of s'.

And for $n \geqslant 1$, given $(\sigma, \tau) \in Q_n$, we define $q = \psi_n(\sigma, \tau)$ as follows:

(ψ1) Set $q(n) = \sigma^{-1}(n)$ and $\delta = \tau^{-1}(n)$.

(ψ2) Let σ' and τ' be, respectively, σ and τ with n deleted, so $(\sigma', \tau') \in Q_{n-1}$.

(ψ3) Set $q' = \psi_{n-1}(\sigma', \tau')$.

(ψ4) For $i < n$, set

$$q(i) = \begin{cases} q'(i) + 1 & \text{if } q'(i) \geqslant \delta, \\ q'(i) & \text{otherwise.} \end{cases}$$

Theorem 6.1. *The functions ϕ_n and ψ_n are well-defined, mutually inverse bijections between P_n and Q_n, and are output and breakpoint preserving.*

This follows as a special case of Theorem 8.1, where $\mathbf{a} = (1, 1, \ldots, 1)$. It can also be proved directly in a similar manner. Surprisingly, however, we have been unable to find a substantially simpler proof of this result, even when using the results of section 9 below.

7 Valet functions and multiset priority queues

From now on, $\mathbf{a} = (a_i)$ will be a finite sequence of positive integers with k terms, and we set $n = \sum_{i=1}^{k} a_i$. Furthermore, if $k \geqslant 1$, we will let the sequence $\mathbf{b} = (b_i)$ consist of the first $k-1$ terms of $\mathbf{a}$ and set $n' = \sum_{i=1}^{k-1} b_i = n - a_k$.

We define valet functions in a similar way to parking functions. There are again n cars, but this time, there are k types of car and k valets. Each valet is responsible for one type of car and has an appropriately sized preferred subset of the parking spaces in which to park those cars. Each valet tries in turn to park all of his cars, allocating one of his cars to each of his preferred spaces, and parking the cars one by one, using the same rules as before. If all of the cars are able to be parked, the chosen subsets form a valet function.

We again formalise this description. We define a *valet function* on $\mathbf{a}$ to be a function $p : [k] \to \mathcal{P}([n])$, with $|p(i)| = a_i$ for each i, for which the following algorithmic function returns `TRUE`. We write $P_{\mathbf{a}}$ for the set of valet functions on this $\mathbf{a}$. (We recall that we write $p(i) = \{p_i(1), \ldots, p_i(a_i)\}$, although we do not make any assumptions about the order of the elements $p_i(1)$, $\ldots$, $p_i(a_i)$ until after we have proved Lemma 7.1.)

```
function TestValet(p, k, a)
    n := Σ_{i=1}^{k} a_i
    L := empty array of length n + 1
    for i := 1 to k do
        for j := 1 to a_i do
(†)         l_0 := min{ l : l ⩾ p_i(j) and L[l] is empty }
            if l_0 = n + 1 then
                return FALSE
            else
                L[l_0] := i
            fi
        od
    od
    return TRUE
```

The next lemma guarantees that if the algorithm returns `TRUE`, it makes sense to speak of *the* permutation $(L[1], L[2], \ldots, L[n])$ of $M_{\mathbf{a}}$. As before, we write $\pi_{\mathbf{a}}(p)$ for this permutation, calling it the *output* of p. (The proof is delayed until the end of this section.)

Lemma 7.1. *The ordering chosen for the elements of each $p(i)$ does not affect either the return value of the algorithm TestValet or, if the return value is* `TRUE`*, the contents of array L when the algorithm terminates.*

Theorem 7.2. *The number of valet functions on* $\mathbf{a}$ *is*

$$\frac{1}{n+1}\prod_{i=1}^{k}\binom{n+1}{a_i}.$$

Proof. We extend Pollak's proof for the number of parking functions on $[n]$ (see section 3 above) to this case. Consider a circular car park with $n+1$ parking spaces labelled 1, 2, …, $n+1$ in order, and allow each valet to choose a subset of preferred spaces of size a_i from the $n+1$ spaces. As described above, each valet allocates one of their cars to each of their preferred spaces. Each valet now tries to park his cars in turn: for each car, he starts at their preferred space for that car and then, if necessary, drive around the circle (in order) until they find an empty space. (This is always possible as there are sufficiently many spaces available.) The number of possible choices of subsets is given by $\prod_{i=1}^{k}\binom{n+1}{a_i}$. A particular choice of subsets yields a valet function if and only if the empty space left after all n cars park is the $(n+1)$-th space. By symmetry, as there are $n+1$ possible choices for the empty space, this happens for precisely $1/(n+1)$ of the possibilities, giving the stated result. □

This result combined with the bijection in section 8 below yields another method of counting allowable pairs, which together with the bijection given by Atkinson, Linton and Walker [2] yields a new way of counting k-way trees (see Atkinson and Walker [4]). Similarly, Pollak's original proof together with the bijection of Kreweras [13] will yield a proof of Cayley's theorem for the number of labelled trees on $n+1$ vertices. Also, if we use the $(0,1)$ matrices introduced by Atkinson and Walker [4] to represent k-way trees, we can set $p(i) = \{\, l : m_{il} = 1 \,\}$ (where $M(\Gamma) = (m_{ij})$ is the matrix of the k-way tree Γ), giving a bijection between valet functions and k-way trees. This clearly restricts to a bijection between parking functions and labelled trees (treating a labelled tree on $n+1$ vertices as an n-way tree), and it satisfies the property described by Knuth [11] in part (a) of the answer to exercise 6.4–31. This bijection is distinct from all of those described in section 3 above, but it turns out that it is actually related to that of Moszkowski [16]; modifying his bijection by ordering

the vertices of the tree using a depth-first search (that is, preorder) instead of a breadth-first search gives our bijection. The proof of this is straightforward using Lemma 7.3 below and the well-known result that a sequence $f(1)$, …, $f(n+1)$ of natural numbers is the down-degree sequence of some tree on $[n]_0$ rooted at 0, traversed in preorder, precisely when $\sum_{i=1}^{m} f(i) \geqslant m$ for $m = 1, 2, \ldots, n$ and $\sum_{i=1}^{n+1} f(i) = n$. (This latter result is essentially due to Schröter; see Rosenbloom [19, pp. 152–156 and 205] for a proof and references.)

We can also extend the concept of a major function correspondingly. We say that a function $f : [k] \to \mathcal{P}([n])$ satisfying $|f(i)| = a_i$ for each i is a *major function* if it satisfies the property

$$\sum_{i=1}^{k} |f(i) \cap [m]| \geqslant m \qquad \text{for } m = 1, 2, \ldots, n.$$

Just as in the parking function case, we have the following lemma (which is also proved below).

Lemma 7.3. *p is a valet function if and only if p is a major function.*

We similarly define $Q_{\mathbf{a}}$ to be the set of all allowable pairs on the multiset $M_{\mathbf{a}}$, that is all allowable pairs (σ, τ) where σ and τ are (multiset) permutations of $M_{\mathbf{a}}$.

Next, we extend the definition of breakpoints to this case. We say that $b \in [n]_0$ is a breakpoint of a function $f : [k] \to \mathcal{P}([n])$, where $|f(i)| = a_i$ as usual, if $\sum_{i=1}^{k} |f(i) \cap [b]| = b$. If p is a valet function, it follows as before that b is a breakpoint if and only if $\{L[1], \ldots, L[b]\} = \{1^{|p(1)\cap[b]|}, \ldots, k^{|p(k)\cap[b]|}\}$ as multisets. Similarly we say that $b \in [n]_0$ is a breakpoint for a pair (σ, τ) of permutations of $M_{\mathbf{a}}$ if $\{\sigma_1, \ldots, \sigma_b\} = \{\tau_1, \ldots, \tau_b\}$ as multisets, and as before, b is a breakpoint of an allowable pair $(\sigma, \tau) \in Q_{\mathbf{a}}$ if and only if, with the natural computation, the queue is empty after outputting τ_b.

Before we prove Lemma 7.1, we introduce some notation which will make it easier to refer to the progress of the algorithms *TestValet* and *TestPair*. Given any function $p : [k] \to \mathcal{P}([n])$ with $|p(i)| = a_i$ for each i, we execute the algorithm *TestValet*$(p, k, \mathbf{a})$, setting

$$E_i(j) = \{\, l \in [n+1] : l \geqslant p_i(j) \text{ and } L[l] \text{ is empty at the start of loop } (i, j) \,\}$$

and

$$\hat{E}_i(j) = \{\, l \in [n+1] : L[l] \text{ is empty at the start of loop } (i, j) \,\},$$

so that

$$E_i(j) = \hat{E}_i(j) \cap \{p_i(j), \ldots, n+1\},$$

where by "the start of loop (i, j)", we mean the point (†) in the algorithm when

the values of i and j are as given. We will never refer to $E_i(j)$ or $\hat{E}_i(j)$ in cases that such a point is not reached. Note that, by construction, the (i,j)-th car will park in $\min E_i(j)$ if it is less than $n+1$, and will fail to park if $E_i(j) = \{n+1\}$.

Similarly, if (σ, τ) are a pair of permutations of $M_{\mathbf{a}}$, we execute the algorithm *TestPair*$((\sigma,\tau), n)$, and let $\mathcal{Q}(i,j)$ be the contents of the queue $\mathcal{Q}$ at the point $(*)$ where the values of i and j are as given. We will also never refer to $\mathcal{Q}(i,j)$ unless such a point is reached.

Proof of Lemma 7.1. It suffices to show that if we swap the values of $p_i(j)$ and $p_i(j+1)$ (where $1 \leqslant j < a_i$), the output is unaffected, since any permutation of $p(i)$ can be achieved by a sequence of transpositions of this form.

We use a prime to distinguish between the executions of *TestValet* with the original ordering of $p(i)$ and the ordering in which $p_i(j)$ and $p_i(j+1)$ have been swapped (the latter having a prime). We note that $L = L'$ at the start of the loop (i,j), as the algorithms are identical until this point; in particular, $\hat{E}_i(j) = \hat{E}'_i(j)$. Consider first the case $p_i(j) < p_i(j+1)$. Then $E'_i(j) \subseteq E_i(j)$ and one of the following holds:

(i) $E_i(j) = \{n+1\}$.

Thus $E'_i(j) = \{n+1\}$ as well and the algorithm returns `FALSE` in both cases.

(ii) $\min E_i(j) < n+1$ but $E_i(j+1) = \{n+1\}$.

Then we either have that $|\hat{E}_i(j) \cap \{p_i(j), \ldots, n\}| = 1$ or, if not, then $\hat{E}_i(j) \cap \{p_i(j+1), \ldots, n\} = \varnothing$. In the former case, if $\min E_i(j) < p_i(j+1)$, then $E'_i(j) = \{n+1\}$, otherwise $E'_i(j+1) = \{n+1\}$. In the latter case $E'_i(j) = \{n+1\}$. Thus in all of these cases the algorithm returns `FALSE` for both orderings.

(iii) $\min E_i(j) < n+1$ and $\min E_i(j+1) < n+1$.

Then we either have $\min E_i(j) < p_i(j+1)$, in which case $\min E'_i(j) = \min E_i(j+1)$ and $\min E'_i(j+1) = \min E_i(j)$, or $\min E_i(j) \geqslant p_i(j+1)$, in which case $\min E'_i(j) = \min E_i(j)$ and $\min E'_i(j+1) = \min E_i(j+1)$. In either case, neither algorithm returns `FALSE` at this point, and the arrays L and L' are identical after these two executions of the inner **for** loop. Since the rest of the algorithms run identically, the lemma holds in this case.

The case $p_i(j) > p_i(j+1)$ is entirely similar and the lemma is thus established. □

Proof of Lemma 7.3. We show that p is not a valet function if and only if p is not a major function.

Assume that p is not a major function. Then we can find an m satisfying $\sum_{i=1}^{k} |p(i) \cap [m]| < m$, so that $\sum_{i=1}^{k} |p(i) \cap \{m+1, \ldots, n\}| > n - m$.

Noting that each possible value of l_0 can be used at most once in the execution of the *TestValet* algorithm, and that $l_0 \geqslant p_i(j)$ for each (i, j), we see that there are more than $n - m$ values of l_0 greater than m when the algorithm runs, so for some (i, j), we must have $l_0 = n + 1$. Thus p is not a valet function.

Conversely, if p is not a valet function, let (i_0, j_0) be the value of (i, j) at which the algorithm returns `FALSE`, so that $\min E_{i_0}(j_0) = n + 1$. Let m be the last empty space in L (other than $n + 1$) when the algorithm terminates, so $m < p_{i_0}(j_0)$. Then for each (i, j) lexicographically less than (i_0, j_0) with $p_i(j) \leqslant m$, we have $\min E_i(j) < m$, as m remains unoccupied. But as the $n - m$ entries $L[m+1], \ldots, L[n]$ are all occupied, it follows that $n - m$ terms of $p_1(1), \ldots, p_{i_0}(j_0 - 1)$ are greater than m, as is $p_{i_0}(j_0)$.

So we see that $\sum_{i=1}^{k} |p(i) \cap \{m+1, \ldots, n\}| \geqslant n - m + 1$, or equivalently $\sum_{i=1}^{k} |p(i) \cap [m]| \leqslant m - 1$, proving that p is not a major function. □

8 Extending the bijection: valet functions and allowable pairs

This bijection is an extension of the one presented in section 6. We also prove here that this really is a bijection as claimed. Theorem 6.1 follows as a corollary of this theorem.

We define functions $\phi_{\mathbf{a}} : P_{\mathbf{a}} \to Q_{\mathbf{a}}$ and $\psi_{\mathbf{a}} : Q_{\mathbf{a}} \to P_{\mathbf{a}}$ inductively. For $k = 0$, the functions are trivial, as the sets only have one element.

For $k \geqslant 1$, given $p \in P_{\mathbf{a}}$, we define $(s, t) = \phi_{\mathbf{a}}(p)$ as follows:

(ϕ1) Set $t = \pi_{\mathbf{a}}(p)$ and $D = t^{-1}(k)$. We let $d_0 = 0$, $d_{a_k+1} = n + 1$ and $D = \{d_1, \ldots, d_{a_k}\}$, where $d_1 < d_2 < \cdots < d_{a_k}$.

(ϕ2) Define $p' \in P_{\mathbf{b}}$ by setting, for $i < k$,

$$p'(i) = \bigcup_{w=0}^{a_k} \{\, l - w : l \in p(i) \text{ and } d_w < l < d_{w+1} \,\}.$$

(ϕ3) Set $(s', t') = \phi_{\mathbf{b}}(p')$.

(ϕ4) We define s by inserting a_k terms labelled k into s' so that $s(j) = k$ if $j \in p(k)$.

And for $k \geqslant 1$, given $(\sigma, \tau) \in Q_{\mathbf{a}}$, we define $q = \psi_{\mathbf{a}}(\sigma, \tau)$ as follows:

($\psi 1$) Set $q(k) = \sigma^{-1}(k)$ and $\Delta = \tau^{-1}(k)$. Let $\delta_0 = 0$, $\delta_{a_k+1} = n+1$ and $\Delta = \{\delta_1, \ldots, \delta_{a_k}\}$, where $\delta_1 < \delta_2 < \cdots < \delta_{a_k}$.

($\psi 2$) Let σ' and τ' be, respectively, σ and τ with all k's deleted, so $(\sigma', \tau') \in Q_{\mathbf{b}}$.

($\psi 3$) Set $q' = \psi_{\mathbf{b}}(\sigma', \tau')$.

($\psi 4$) For $i < k$, set

$$q(i) = \bigcup_{w=0}^{a_k} \{\, \lambda + w : \lambda \in q'(i) \text{ and } \delta_w < \lambda + w < \delta_{w+1} \,\}.$$

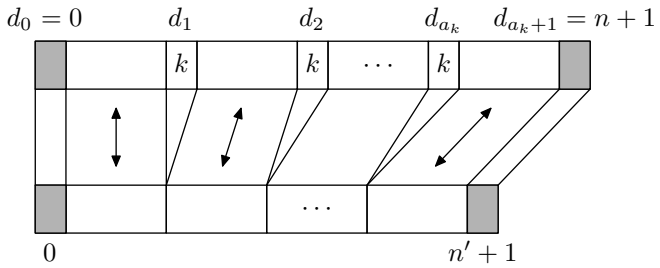
Theorem 8.1. *The functions $\phi_{\mathbf{a}}$ and $\psi_{\mathbf{a}}$ are well-defined, mutually inverse bijections between $P_{\mathbf{a}}$ and $Q_{\mathbf{a}}$, and are output and breakpoint preserving.*

Proof. We prove the result by induction on k, the case $k = 0$ being trivial. We must first show that the algorithms are well-defined. The only problematic parts here are the steps ($\phi 2$) and ($\psi 2$), where we must justify the claims that $p' \in P_{\mathbf{b}}$ and $(\sigma', \tau') \in Q_{\mathbf{b}}$. We then show that both functions preserve breakpoints and identify a special set of breakpoints, from which we deduce that $\phi_{\mathbf{a}}(p) \in Q_{\mathbf{a}}$ and $\psi_{\mathbf{a}}(\sigma, \tau) \in P_{\mathbf{a}}$. Finally, we show that $\phi_{\mathbf{a}}$ and $\psi_{\mathbf{a}}$ are mutually inverse and output preserving. The proof is quite technical in nature.

Throughout the proof we will assume, using Lemma 7.1, that the elements of $p(i)$, $p'(i)$ and the like are listed in increasing order. In particular, this allows us to make statements such as: $p'_i(j) = p_i(j) - w$ where $d_w < p_i(j) < d_{w+1}$.

It is also useful to note here that both $(d_w - w)$ and $(\delta_w - w)$ are non-decreasing sequences, as both (d_w) and (δ_w) are strictly increasing sequences.

We use the following figure to guide us in our thinking, for example when we consider how to obtain t' from t. The top row represents a permutation of $M_{\mathbf{a}}$ and the bottom row represents the corresponding permutation of $M_{\mathbf{b}}$. Alternatively, we can think of the top and bottom rows as representing L and L' respectively, suggesting the relationship between p and p', E and E' and the like. The shaded boxes in the rows represent 0 and $n+1$ or $n'+1$, which are sometimes needed. Many of the steps in the proof given are "obvious" in terms of this picture, but we prefer to give formal proofs in terms of the defining algorithms.



(i) The step ($\phi 2$) is well-defined and $\pi_{\mathbf{b}}(p')$ is $\pi_{\mathbf{a}}(p)$ with all k's deleted.

We could show that step ($\phi 2$) is well-defined by proving that p' is a major function. However, we prefer to show the result directly from the algorithm *TestValet*, for this also allows us to show that $\pi_{\mathbf{b}}(p')$ is $\pi_{\mathbf{a}}(p)$ with all k's deleted. More precisely, if we set $t = \pi_{\mathbf{a}}(p)$ and $t' = \pi_{\mathbf{b}}(p')$, we show that, for $1 \leqslant l' \leqslant n'$,

$$t'(l') = t(l' + w) \qquad \text{where } d_w < l' + w < d_{w+1}. \tag{2}$$

(Note that because of step ($\phi 1$) in the calculation of $\phi_{\mathbf{b}}(p')$, the t' of step ($\phi 3$) really is $\pi_{\mathbf{b}}(p')$.)

We compare the executions of *TestValet*$(p, k, \mathbf{a})$ and *TestValet*$(p', k-1, \mathbf{b})$, distinguishing the variables associated with the latter by using a prime. We claim that, for $i < k$,

$$\hat{E}'_i(j) = \{n' + 1\} \cup \bigcup_{w=0}^{a_k} \{\, l - w : l \in \hat{E}_i(j) \text{ and } d_w < l < d_{w+1} \,\}. \tag{3}$$

Before proving this claim, we show that given (3) for a particular (i, j), it follows that

$$E'_i(j) = \{n' + 1\} \cup \bigcup_{w=0}^{a_k} \{\, l - w : l \in E_i(j) \text{ and } d_w < l < d_{w+1} \,\} \tag{4}$$

for this (i, j). To show this, recall that $E'_i(j) = \hat{E}'_i(j) \cap \{p'_i(j), \ldots, n' + 1\}$. It is thus sufficient to show that for each w we have

$$\begin{aligned} &\{\, l - w : l \in \hat{E}_i(j) \text{ and } d_w < l < d_{w+1} \,\} \cap \{p'_i(j), \ldots, n' + 1\} \\ &\qquad = \{\, l - w : l \in E_i(j) \text{ and } d_w < l < d_{w+1}, \,\}. \end{aligned} \tag{5}$$

Let w_0 be such that $d_{w_0} < p_i(j) < d_{w_0+1}$, so that $p'_i(j) = p_i(j) - w_0$. We consider the three cases $w < w_0$, $w > w_0$ and $w = w_0$ separately. If $w < w_0$, then as $(d_{w+1} - 1) - w \leqslant d_{w_0} - w_0 < p'_i(j)$, the left hand side of (5) is empty, as is the right hand side, since $d_{w+1} \leqslant d_{w_0} < p_i(j) \leqslant \min E_i(j)$.

If $w > w_0$, then $(d_w + 1) - w \geqslant (d_{w_0+1} - (w_0 + 1)) + 1 > p'_i(j)$, so the left hand side is simply $\{\, l - w : l \in \hat{E}_i(j) \text{ and } d_w < l < d_{w+1} \,\}$. But as $p_i(j) < d_w$, it follows that $l \in \hat{E}_i(j)$ if and only if $l \in E_i(j)$ when $l > d_w$. Thus the right hand side is the same.

Finally, if $w = w_0$ we have

$$\begin{aligned}
&\{\, l - w_0 : l \in E_i(j) \text{ and } d_{w_0} < l < d_{w_0+1} \,\} \\
&\qquad = \{\, l - w_0 : l \in \hat{E}_i(j) \text{ and } l \geqslant p_i(j) \text{ and } d_{w_0} < l < d_{w_0+1} \,\} \\
&\qquad = \{\, l - w_0 : l \in \hat{E}_i(j) \text{ and } d_{w_0} < l < d_{w_0+1} \,\} \cap \{p'_i(j), \ldots, n'+1\},
\end{aligned}$$

where the last line follows as $p_i(j) - w_0 = p'_i(j)$, and for any $l < d_{w_0+1}$, we have the inequalities $l - w_0 \leqslant d_{w_0+1} - (w_0+1) \leqslant d_{a_k+1} - (a_k+1) = n'$. Thus our claim about $E'_i(j)$ follows from that about $\hat{E}'_i(j)$.

We now prove our claim about $\hat{E}'_i(j)$. We show that (3) holds every time (†) is reached in the execution of the algorithms. On the first occasion that (†) is reached, we have $(i,j) = (1,1)$, and as $\hat{E}'_i(j) = [n'+1]$ and $\hat{E}_i(j) = [n+1]$, the claim holds.

Assume that (3) holds at (†) when $(i,j) = (i_0, j_0)$, where $i_0 < k$. We have $l_0 = \min E_{i_0}(j_0) \leqslant n$, and we let w_0 be such that $d_{w_0} < l_0 < d_{w_0+1}$, noting that $l_0 \notin D$ when $i < k$. It follows easily from (4) that $l'_0 = \min E'_{i_0}(j_0) = l_0 - w_0$. Thus the next time (†) is reached, $\hat{E}_i(j) = \hat{E}_{i_0}(j_0) \setminus \{l_0\}$ and $\hat{E}'_i(j) = \hat{E}'_{i_0}(j_0) \setminus \{l'_0\}$ (where we have $(i,j) = (i_0, j_0+1)$ or $(i_0+1, 1)$, according to whether $j_0 < a_{i_0}$ or $j_0 = a_{i_0}$). Thus we have

$$\begin{aligned}
\hat{E}'_i(j) &= \hat{E}'_{i_0}(j_0) \setminus \{l'_0\} \\
&= \Big(\{n'+1\} \cup \bigcup_{w=0}^{a_k} \{\, l - w : l \in \hat{E}_{i_0}(j_0) \text{ and } d_w < l < d_{w+1} \,\}\Big) \\
&\qquad \setminus \{l_0 - w_0\} \\
&= \{n'+1\} \cup \bigcup_{w=0}^{a_k} \{\, l - w : l \in \hat{E}_i(j) \text{ and } d_w < l < d_{w+1} \,\},
\end{aligned}$$

proving that (3) holds for (i,j), and hence (3) holds for all (i,j) pairs.

It follows immediately from this that $p' \in P_{\mathbf{b}}$ and that $\pi_{\mathbf{b}}(p')$ is $\pi_{\mathbf{a}}(p)$ with all k's deleted: if $d_w < l' + w = l < d_{w+1}$, then for some (i,j) with $i < k$, we have $l = \min E_i(j)$, and for this (i,j), we see that $l_0 = l$ and $l'_0 = l'$. As $t(l_0) = t'(l'_0) = i$, we have $t'(l') = t(l'+w)$, proving (2). Also, the above showed that for each (i_0, j_0) with $i_0 < k$, we have $l'_0 = l_0 - w_0$, so $l'_0 < d_{w_0+1} - w_0 \leqslant d_{a_k+1} - a_k = n'+1$, so p' is a valet function.

(ii) The step ($\psi 2$) is well-defined.

Consider the natural priority queue computation for (σ, τ). Removing the i-th Insert for each $i \in \sigma^{-1}(k)$ and the j-th DeleteMin for each $j \in \tau^{-1}(k)$ yields a priority queue computation (actually the natural one)

which produces an output of τ' given an input of σ'. Thus $(\sigma', \tau') \in Q_{\mathbf{b}}$.

This argument can be formalised in terms of the algorithm *TestPair*, as in part (vii) below, but we shall not give details here.

(iii) $\phi_{\mathbf{a}}$ preserves breakpoints.

Let b be a breakpoint of p and let w be such that $d_w \leqslant b < d_{w+1}$, so that for $i < k$, $p_i(j) \leqslant b$ if and only if $p'_i(j) \leqslant b - w$. Also, $|p(k) \cap [b]| = w$ as $t^{-1}(k) = \{d_1, \ldots, d_{a_k}\}$ and b is a breakpoint of p. Thus

$$\begin{aligned}\sum_{i=1}^{k-1} |p'(i) \cap [b-w]| &= \sum_{i=1}^{k-1} |p(i) \cap [b]| \\ &= \sum_{i=1}^{k} |p(i) \cap [b]| \ - \ |p(k) \cap [b]| \\ &= b - w,\end{aligned}$$

so $b - w$ is a breakpoint of p'.

By the inductive hypothesis, $b - w$ is a breakpoint of $\phi_{\mathbf{b}}(p') = (s', t')$, so that $\{s'_1, \ldots, s'_{b-w}\} = \{t'_1, \ldots, t'_{b-w}\}$ as multisets. But we noted above that $|p(k) \cap [b]| = w$ and that $t_i = k$ for $i = d_1, d_2, \ldots, d_{a_k}$, so using step ($\phi 4$), we see that

$$\begin{aligned}\{s_1, \ldots, s_b\} &= \{s'_1, \ldots, s'_{b-w}\} \cup \{k^w\} \\ &= \{t'_1, \ldots, t'_{b-w}\} \cup \{k^w\} \\ &= \{t_1, \ldots, t_b\},\end{aligned}$$

showing that b is a breakpoint of (s, t).

(iv) $\psi_{\mathbf{a}}$ preserves breakpoints.

We use an argument similar to that of part (iii) above. Let b be a breakpoint of (σ, τ) and let w be such that $\delta_w \leqslant b < \delta_{w+1}$. Then we have $\{\tau_1, \ldots, \tau_b\} = \{\tau'_1, \ldots, \tau'_{b-w}\} \cup \{k^w\}$. But as b is a breakpoint of (σ, τ), we must have $\{\sigma_1, \ldots, \sigma_b\} = \{\sigma'_1, \ldots, \sigma'_{b-w}\} \cup \{k^w\}$ as well. Thus $\{\sigma'_1, \ldots, \sigma'_{b-w}\} = \{\tau'_1, \ldots, \tau'_{b-w}\}$ and $b - w$ is a breakpoint for (σ', τ'). By the inductive hypothesis, it follows that $b - w$ is a breakpoint for q'.

It is relatively straightforward to show that for $i < k$, $q'_i(j) \leqslant b - w$ if and

only if $q_i(j) \leqslant b$. Also, $|q(k) \cap [b]| = |\sigma^{-1}(k) \cap [b]| = w$, hence

$$\begin{aligned}\sum_{i=1}^{k} |q(i) \cap [b]| &= \sum_{i=1}^{k-1} |q(i) \cap [b]| + |q(k) \cap [b]| \\ &= \sum_{i=1}^{k-1} |q'(i) \cap [b-w]| + w \\ &= (b-w) + w = b,\end{aligned}$$

so b is a breakpoint of q as required.

(v) $d_w - w$ is a breakpoint of p' for each w.

In the execution of *TestValet*$(p, k, \mathbf{a})$, we have $\hat{E}_k(1) = D \cup \{n+1\}$, so that $d_w \in \hat{E}_i(j)$ for all (i, j) with $i < k$. Thus, for such pairs, $p_i(j) < d_w$ if and only if $\min E_i(j) < d_w$. It follows that $\sum_{i=1}^{k-1} |p(i) \cap [d_w]| = d_w - w$, as $\min E_i(j)$ is distinct for distinct (i, j) and $\{ \min E_i(j) : i < k \} = [n] \setminus D$.

Now for $i < k$, if $p_i(j) < d_w$ then $p'_i(j) \leqslant d_w - w$ by the definition of p' and the monotonicity of $d_w - w$. Also, if $p_i(j) > d_w$, then $p'_i(j) > d_w - w$. Thus $p_i(j) < d_w$ if and only if $p'_i(j) \leqslant d_w - w$, and

$$\begin{aligned}\sum_{i=1}^{k-1} |p'(i) \cap [d_w - w]| &= \sum_{i=1}^{k-1} |p(i) \cap [d_w]| \\ &= d_w - w,\end{aligned}$$

as required.

(vi) $\delta_w - w$ is a breakpoint of (σ', τ') for each w.

Consider the natural computation for (σ, τ). Remove the i-th Insert and the j-th DeleteMin for each $i \in \sigma^{-1}(k)$ and $j \in \tau^{-1}(k)$ to get a priority queue computation for (σ', τ'), as in part (ii) above.

Note that when k is output in the priority queue computation for (σ, τ), the queue $\mathcal{Q}$ contains only k's. Thus, in the priority queue computation for (σ', τ'), the queue will be empty at each point at which k would have been output in the corresponding computation for (σ, τ), that is, after each $(\delta_w - w)$-th DeleteMin.

As above, this argument can be formalised by comparing the executions of *TestPair*$((\sigma, \tau), n)$ and *TestPair*$((\sigma', \tau'), n')$. It is similar to, but easier than, the argument in part (vii) below.

(vii) $\phi_{\mathbf{a}}$ produces allowable pairs.

Given $p \in P_{\mathbf{a}}$, we show that $(s,t) = \phi_{\mathbf{a}}(p)$ is an allowable pair. By the inductive hypothesis, we know that $(s',t') = \phi_{\mathbf{b}}(p')$ is an allowable pair, and by step ($\phi 4$) and the result of (i) above, (s,t) is obtained from (s',t') by inserting k's into s' and t' in positions determined by $p(k)$ and D respectively.

To show that (s,t) is an allowable pair, we compare the execution of the algorithms *TestPair*$((s,t),n)$ and *TestPair*$((s',t'),n')$, distinguishing the variables in the two executions by using a prime. We set

$$u(i) = |s^{-1}(k) \cap [i-1]|$$

and

$$v(j) = |t^{-1}(k) \cap [j-1]|,$$

noting that $s_i = s'_{i-u(i)}$ if $s_i \neq k$ and $t_j = t'_{j-v(j)}$ if $t_j \neq k$. As $j \leqslant i$ throughout the execution of the algorithm, and $|p(k) \cap [j]| \geqslant |D \cap [j]|$ for all j, we see that $u(i) \geqslant v(j)$ and $u(i+1) \geqslant v(j+1)$ whenever $(*)$ is reached in the algorithm.

We claim that on each such occasion we have

$$\mathcal{Q}(i,j) = \mathcal{Q}'(i-u(i), j-v(j)) \cup \{k^{u(i)-v(j)}\}. \tag{6}$$

Recall that $\mathcal{Q}(i,j)$ is the content of the queue $\mathcal{Q}$ at the point $(*)$ when i and j are as given. It is certainly true when $(i,j) = (1,1)$. Given this result for $(i,j) = (i_0, j_0)$, we consider four possibilities for the next step in *TestPair*$((s,t),n)$, showing that in each case the claim is true the next time $(*)$ is reached. We assume that the corresponding execution of *TestPair*$((s',t'),n')$ has reached $(*)$ with $(i',j') = (i_0 - u(i_0), j_0 - v(j_0))$; it will follow from this induction argument that this point is indeed reached.

(a) $t_{j_0} \notin \mathcal{Q}$ and $s_{i_0} \neq k$.

We cannot have $t_{j_0} = k$ in this case, for if $k \notin \mathcal{Q}$, then $u(i_0) = v(j_0)$ from (6). As $u(i_0+1) \geqslant v(j_0+1) \geqslant v(j_0)$ but $s_{i_0} \neq k$, we must have $u(i_0+1) = u(i_0)$ and $v(j_0+1) = v(j_0)$, hence $t_{j_0} \neq k$. Thus $t'_{j_0-v(j_0)} \notin \mathcal{Q}'$, so $s_{i_0} = s'_{i_0-u(i_0)}$ is inserted in both algorithms, giving

$$\begin{aligned}
\mathcal{Q}(i_0+1, j_0) &= \mathcal{Q}(i_0, j_0) \cup \{s_{i_0}\} \\
&= \mathcal{Q}'(i_0 - u(i_0), j_0 - v(j_0)) \cup \{k^{u(i_0)-v(j_0)}\} \\
&\qquad \cup \{s'_{i_0-u(i_0)}\} \\
&= \mathcal{Q}'(i_0 + 1 - u(i_0+1), j_0 - v(j_0)) \cup \{k^{u(i_0+1)-v(j_0)}\},
\end{aligned}$$

as $u(i_0+1) = u(i_0)$ in this case.

(b) $t_{j_0} \notin \mathcal{Q}$ and $s_{i_0} = k$.

Then after INSERT(s_{i_0}) is executed in *TestPair*$((s,t),n)$, we have

$$\begin{aligned}\mathcal{Q}(i_0+1, j_0) &= \mathcal{Q}(i_0, j_0) \cup \{k\} \\ &= \mathcal{Q}'(i_0 - u(i_0), j_0 - v(j_0)) \cup \{k^{u(i_0)-v(j_0)}\} \cup \{k\} \\ &= \mathcal{Q}'(i_0 + 1 - u(i_0+1), j_0 - v(j_0)) \cup \{k^{u(i_0+1)-v(j_0)}\},\end{aligned}$$

as $u(i_0+1) = u(i_0)+1$ in this case.

(c) $t_{j_0} \in \mathcal{Q}$ and $t_{j_0} \neq k$.

As $t_{j_0} = t'_{j_0-v(j_0)}$, we see that $t'_{j_0-v(j_0)} \in \mathcal{Q}'(i_0 - u(i_0), j_0 - v(j_0))$ and thus $t'_{j_0-v(j_0)} = \min \mathcal{Q}'$ (as *TestPair*$((s',t'),n')$ does not return `FALSE`). It follows from (6) that $t_{j_0} = \min \mathcal{Q}(i_0, j_0)$, and therefore *TestPair*$((s,t),n)$ does not return `FALSE` at this point either. After the DELETEMIN is executed in each of the algorithms, we have removed $t_{j_0} = t'_{j_0-v(j_0)}$ from both $\mathcal{Q}$ and $\mathcal{Q}'$, so

$$\begin{aligned}\mathcal{Q}(i_0, j_0+1) &= \mathcal{Q}(i_0, j_0) \setminus \{t_{j_0}\} \\ &= \big(\mathcal{Q}'(i_0 - u(i_0), j_0 - v(j_0)) \cup \{k^{u(i_0)-v(j_0)}\}\big) \\ &\qquad \setminus \{t'_{j_0-v(j_0)}\} \\ &= \mathcal{Q}'(i_0 - u(i_0), j_0 + 1 - v(j_0+1)) \cup \{k^{u(i_0)-v(j_0+1)}\},\end{aligned}$$

as $v(j_0+1) = v(j_0)$ in this case.

(d) $t_{j_0} \in \mathcal{Q}$ and $t_{j_0} = k$.

Then $j_0 \in D$, say $j_0 = d_w$, so $v(j_0) = w-1$. Thus we have

$$\begin{aligned}\mathcal{Q}(i_0, j_0) &= \mathcal{Q}'(i_0 - u(i_0), j_0 - v(j_0)) \cup \{k^{u(i_0)-v(j_0)}\} \\ &= \mathcal{Q}'(i_0 - u(i_0), d_w - w + 1) \cup \{k^{u(i_0)-v(j_0)}\}\end{aligned}$$

We must have $u(i_0) > v(j_0)$ as $k \in \mathcal{Q}$, so to show that the execution of *TestPair*$((s,t),n)$ does not return `FALSE` at this point, it suffices to show that $\mathcal{Q}'(i_0 - u(i_0), d_w - w + 1) = \varnothing$. By (v), $d_w - w$ is a breakpoint of p', and by the inductive hypothesis, it follows that $d_w - w$ is a breakpoint of (s',t'). Thus at the point that t'_{d_w-w} was deleted from the queue in *TestPair*$((s',t'),n')$, $\mathcal{Q}'$ was empty. (This has already occurred, as now $j' = d_w - w + 1$.) Let i_1 be the smallest value of i satisfying $i - u(i) = d_w - w + 1$ for which (i_1, j_0) occurred as a value of (i,j) during the execution of *TestPair*$((s,t),n)$. Then $i_1 \leqslant i_0$ and $\mathcal{Q}'(i_1 - u(i_1), d_w - w + 1) = \varnothing$. Now if $u(i_1) - v(j_0) = u(i_1) - w + 1 > 0$,

it follows that $t_{j_0} = k \in \mathcal{Q}$ when $(i,j) = (i_1, j_0)$, hence $i_1 = i_0$ and $\mathcal{Q}'(i_0 - u(i_0), d_w - w + 1) = \varnothing$ as required. If not, then we have $\mathcal{Q}(i_1, j_0) = \varnothing$, which shows that $i_1 = j_0 = d_w$. Now since $v(d_w + 1) \leqslant u(d_w + 1)$, but $v(d_w) = u(d_w)$ and $d_w \in D$, we deduce that $d_w \in p(k)$. Hence $s_{i_1} = k$, so that $i_0 = i_1 + 1$ and $u(i_0) = u(i_1) + 1$, giving $\mathcal{Q}'(i_0 - u(i_0), d_w - w + 1) = \mathcal{Q}'(i_1 - u(i_1), d_w - w + 1) = \varnothing$ as required. Thus *TestPair*$((s,t), n)$ does not return `FALSE` at this point.

After this step, we have $j = j_0 + 1$ and $v(j) = v(j_0) + 1$, so

$$\begin{aligned}\mathcal{Q}(i_0, j_0 + 1) &= \mathcal{Q}(i_0, j_0) \setminus \{k\} \\ &= \big(\mathcal{Q}'(i_0 - u(i_0), j_0 - v(j_0)) \cup \{k^{u(i_0) - v(j_0)}\}\big) \setminus \{k\} \\ &= \mathcal{Q}'(i_0 - u(i_0), j_0 + 1 - v(j_0 + 1)) \cup \{k^{u(i_0) - v(j_0 + 1)}\}.\end{aligned}$$

It follows from the analysis of these four cases that our claim holds. It follows from our proof of cases (c) and (d) that every time the test $t_j \in \mathcal{Q}$ is carried out in *TestPair*$((s,t), n)$, the test succeeds, so (s,t) is an allowable pair.

(viii) $\psi_{\mathbf{a}}$ produces valet functions.

Given $(\sigma, \tau) \in Q_{\mathbf{a}}$, it suffices to demonstrate that $q = \psi_{\mathbf{a}}(\sigma, \tau)$ is a major function (appealing to Lemma 7.3), that is, given $m \in [n]$, we show that $\sum_{i=1}^{k} |q(i) \cap [m]| \geqslant m$. We note that by the inductive hypothesis, we already have $q' \in P_{\mathbf{b}}$, so q' is a major function.

Let w be such that $\delta_w \leqslant m < \delta_{w+1}$. For $i < k$, we can easily deduce that $q_i(j) \leqslant m$ if and only if $q'_i(j) \leqslant m - w$. Thus $\sum_{i=1}^{k-1} |q(i) \cap [m]| = \sum_{i=1}^{k-1} |q'(i) \cap [m - w]| \geqslant m - w$. But we also know that $|q(k) \cap [m]| = |\sigma^{-1}(k) \cap [m]| \geqslant |\tau^{-1}(k) \cap [m]| = w$, and hence $\sum_{i=1}^{k} |q(i) \cap [m]| \geqslant m$. Thus q is a valet function as required.

(ix) $\psi_{\mathbf{a}}\phi_{\mathbf{a}} = \mathrm{Id}_{P_{\mathbf{a}}}$.

Given $p \in P_{\mathbf{a}}$, we set $(s,t) = \phi_{\mathbf{a}}(p)$, then $(\sigma, \tau) = (s,t)$ and finally set $q = \psi_{\mathbf{a}}(\sigma, \tau) = \psi_{\mathbf{a}}\phi_{\mathbf{a}}(p)$. We wish to show that $p = q$.

As $\tau = t$, we have $\Delta = D$, so $\delta_w = d_w$ for each w. As $p_i(j) \notin D$ for $i < k$ and $q(k) = p(k)$, it is clear that if $p' = q'$, then $p = q$. But as $p' = \psi_{\mathbf{b}}(s', t')$ by the inductive hypothesis and $q' = \psi_{\mathbf{b}}(\sigma', \tau')$ by step ($\psi 3$), it suffices to show that $(s', t') = (\sigma', \tau')$. However, we showed in (i) that t' is t with all k's deleted, and clearly s' is s with all k's deleted by step ($\phi 4$). Also, step ($\psi 2$) tells us that σ' and τ' are respectively σ and τ with all k's

deleted. Thus, since $(s,t) = (\sigma,\tau)$, we have $(s',t') = (\sigma',\tau')$, so $p = q$ and $\psi_{\mathbf{a}}\phi_{\mathbf{a}} = \mathrm{Id}_{P_{\mathbf{a}}}$.

(x) $\phi_{\mathbf{a}}\psi_{\mathbf{a}} = \mathrm{Id}_{Q_{\mathbf{a}}}$ and $\pi_{\mathbf{a}}(\psi_{\mathbf{a}}(\sigma,\tau)) = \tau$.

Given $(\sigma,\tau) \in Q_{\mathbf{a}}$, we first set $q = \psi_{\mathbf{a}}(\sigma,\tau)$, then set $p = q$ and finally set $(s,t) = \phi_{\mathbf{a}}(p) = \phi_{\mathbf{a}}\psi_{\mathbf{a}}(\sigma,\tau)$. We show that $\pi_{\mathbf{a}}(q) = \tau$, and then use this to deduce that $(s,t) = (\sigma,\tau)$; these are the two results desired.

We proceed in a manner similar to that used in (i). We know by the inductive hypothesis that $(\sigma',\tau') = \phi_{\mathbf{b}}(q')$, so we also have $\tau' = \pi_{\mathbf{b}}(q')$ by step (ϕ1). Comparing the execution of *TestValet*$(q,k,\mathbf{a})$ with that of *TestValet*$(q',k-1,\mathbf{b})$, we claim that for $i < k$,

$$\hat{E}_i(j) = \{n+1\} \cup \Delta \cup \bigcup_{w=0}^{a_k} \{\, l+w : l \in \hat{E}'_i(j) \text{ and } \delta_w < l+w < \delta_{w+1} \,\}.$$

It follows immediately from this, as in (i), that

$$E_i(j) = \{n+1\} \cup \{\, \delta \in \Delta : \delta \geqslant q_i(j) \,\} \cup \bigcup_{w=0}^{a_k} \{\, l+w : l \in E'_i(j) \text{ and } \delta_w < l+w < \delta_{w+1} \,\}.$$

We prove the claim by showing that it holds each time (†) is reached in the algorithms. The result is trivial on the first occasion, as $(i,j) = (1,1)$, so that $\hat{E}_i(j) = [n+1]$ and $\hat{E}'_i(j) = [n'+1]$. Assume the result to be true at (†) when $(i,j) = (i_0,j_0)$, where $i_0 < k$. We let w_0 be such that $\delta_{w_0} < q_{i_0}(j_0) < \delta_{w_0+1}$, so that $q'_{i_0}(j_0) = q_{i_0}(j_0) - w_0$ satisfies $\delta_{w_0} - w_0 < q'_{i_0}(j_0) \leqslant \delta_{w_0+1} - (w_0+1)$. But $\delta_{w_0+1} - (w_0+1)$ is a breakpoint of q' by part (iv) and the inductive hypothesis, so it follows that $\delta_{w_0} - w_0 < l'_0 = \min E'_{i_0}(j_0) \leqslant \delta_{w_0+1} - (w_0+1)$. Thus we have

$$\begin{aligned} l_0 &= \min E_{i_0}(j_0) \\ &= \min\Big(\{n+1\} \cup \{\, \delta \in \Delta : \delta \geqslant q_{i_0}(j_0) \,\} \\ &\qquad \cup \bigcup_{w=0}^{a_k} \{\, l+w : l \in E'_{i_0}(j_0) \text{ and } \delta_w < l+w < \delta_{w+1} \,\}\Big) \\ &= \min\big(\{n+1\} \cup \{\, \delta \in \Delta : \delta \geqslant q_{i_0}(j_0) \,\} \cup \{l'_0 + w_0\}\big). \end{aligned}$$

But $\delta_{w_0} < l'_0 + w_0 \leqslant \delta_{w_0+1} - 1$, so $l_0 = l'_0 + w_0$. Thus the next time that

(†) is reached, we have $\hat{E}'_i(j) = \hat{E}'_{i_0}(j_0) \setminus \{l'_0\}$ and

$$\begin{aligned}\hat{E}_i(j) &= \hat{E}_{i_0}(j_0) \setminus \{l_0\} \\ &= \Big(\{n+1\} \cup \Delta \cup \bigcup_{w=0}^{a_k} \{\, l+w : l \in \hat{E}'_{i_0}(j_0) \text{ and } \delta_w < l+w < \delta_{w+1} \,\}\Big) \\ &\qquad \setminus \{l'_0 + w_0\} \\ &= \{n+1\} \cup \Delta \cup \bigcup_{w=0}^{a_k} \{\, l+w : l \in \hat{E}'_i(j) \text{ and } \delta_w < l+w < \delta_{w+1} \,\},\end{aligned}$$

as required, where $(i,j) = (i_0, j_0+1)$ or $(i_0+1, 1)$ according as $j_0 < a_{i_0}$ or $j_0 = a_{i_0}$.

In particular, this proof shows that at the end of the loop $i = k-1$, we have, for each w,

$$(L[\delta_w + 1], \ldots, L[\delta_{w+1} - 1]) = (L'[\delta_w - w + 1], \ldots, L'[\delta_{w+1} - (w+1)]),$$

and $L[\delta_w]$ is still empty. Thus during the loop $i = k$ in *TestValet*$(q, k, \mathbf{a})$, $\min E_k(j) \in \Delta$ for each j (using (viii)), so that $\pi_{\mathbf{a}}(q)$ is $\pi_{\mathbf{b}}(q') = \tau'$ with k's inserted into the a_k positions determined by Δ. Thus $\pi_{\mathbf{a}}(q) = \tau$ as stated.

We are now able to show that $(s,t) = (\sigma, \tau)$. Having shown that $\pi_{\mathbf{a}}(q) = \tau$, and noting that $t = \pi_{\mathbf{a}}(p)$, we deduce that $t = \tau$, as $p = q$. It follows that $D = t^{-1}(k) = \tau^{-1}(k) = \Delta$. By construction, $q_i(j) \notin \Delta$ for $i < k$, so steps ($\phi 2$) and ($\psi 4$) now yield $p' = q'$. But then, by the inductive hypothesis, $(s', t') = \phi_{\mathbf{b}}(p') = \phi_{\mathbf{b}}(q') = (\sigma', \tau')$, so $s' = \sigma'$. As s is s' with k's inserted in the positions determined by $p(k) = q(k)$, and σ is σ' with k's inserted in the positions determined by $\sigma^{-1}(k) = q(k)$, it follows that $s = \sigma$, hence $(s,t) = (\sigma, \tau)$ and $\phi_{\mathbf{a}}\psi_{\mathbf{a}} = \mathrm{Id}_{Q_{\mathbf{a}}}$.

(xi) $\phi_{\mathbf{a}}$ and $\psi_{\mathbf{a}}$ are both output preserving.

That $\phi_{\mathbf{a}}$ is output preserving is clear from step ($\phi 1$), and $\psi_{\mathbf{a}}$ is output preserving by the result of part (x) above. □

Corollary 8.2. (a) *Let $p \in P_{\mathbf{a}}$, $t = \pi_{\mathbf{a}}(p)$ and $D = t^{-1}(k)$. Then $\max D$ is a breakpoint of p.*

(b) *Let $(\sigma, \tau) \in Q_{\mathbf{a}}$ and $\Delta = \tau^{-1}(k)$. Then $\max \Delta$ is a breakpoint of (σ, τ).*

Proof. (a) In part (v) of the proof, we noted that $\sum_{i=1}^{k-1} |p(i) \cap [d_w]| = d_w - w$ for each w. In particular, when $w = a_k$, so that $d_w = \max D$, we have $|p(k) \cap [d_w]| = a_k = w$ (as all the cars in $p(k)$ park in the spaces in D), so $\sum_{i=1}^{k} |p(i) \cap [d_w]| = d_w$ as required.

(b) In part (vi) of the proof, we noted that when k is output in the computation of (σ, τ), the queue $\mathcal{Q}$ contains only k's. Thus when the final k is output, $\mathcal{Q}$ must be empty, so $\max \Delta$ is a breakpoint of (σ, τ). □

9 Alternative descriptions of the bijections

It is possible to calculate all of ϕ_n, ψ_n (as defined in section 6), $\phi_{\mathbf{a}}$ and $\psi_{\mathbf{a}}$ non-inductively, as we now demonstrate.

Given $(\sigma, \tau) \in Q_n$, we define for each $j \in [n]$

$$S(\sigma, j) = |\{\, l \in [j] : \sigma_l \leqslant \sigma_j \,\}|$$

and

$$T(\tau, j) = |\{\, l \in [j] : \tau_l > \tau_j \,\}|.$$

We then set $q(i) = S(\sigma, \sigma^{-1}(i)) + T(\tau, \tau^{-1}(i))$ and claim that $q = \psi_n(\sigma, \tau)$.

We can extend this to multisets as follows. Given $(\sigma, \tau) \in Q_{\mathbf{a}}$, for each $i \in [k]$ we list the elements of $\sigma^{-1}(i)$ and $\tau^{-1}(i)$ in increasing order as $\bar{\sigma}_i(1), \ldots, \bar{\sigma}_i(a_i)$ and $\bar{\tau}_i(1), \ldots, \bar{\tau}_i(a_i)$ respectively. Setting

$$q(i) = \{\, S(\sigma, \bar{\sigma}_i(j)) + T(\tau, \bar{\tau}_i(j)) : j \in [a_i] \,\} \tag{7}$$

gives $q = \psi_{\mathbf{a}}(\sigma, \tau)$, as we now show by induction.

The statement is vacuously true if $k = 0$. For $k \geqslant 1$, we assume this result to be true for $k - 1$, so using the notation of the previous section, we have, for $i < k$,

$$q'(i) = \{\, S(\sigma', \bar{\sigma}'_i(j)) + T(\tau', \bar{\tau}'_i(j)) : j \in [a_i] \,\}.$$

We consider the relationship between $S(\sigma, \bar{\sigma}_i(j))$ and $S(\sigma', \bar{\sigma}'_i(j))$, noting that $\sigma_{\bar{\sigma}_i(j)} = i$ by definition of $\bar{\sigma}_i(j)$. We have

$$S(\sigma, \bar{\sigma}_i(j)) = |\{\, l \in [\bar{\sigma}_i(j)] : \sigma_l \leqslant i \,\}|$$

and

$$S(\sigma', \bar{\sigma}'_i(j)) = |\{\, l \in [\bar{\sigma}'_i(j)] : \sigma'_l \leqslant i \,\}|.$$

But as σ' is just σ with all of the k's deleted, we see that $\{\sigma_1, \ldots, \sigma_{\bar{\sigma}_i(j)}\} = \{\sigma'_1, \ldots, \sigma'_{\bar{\sigma}'_i(j)}\} \cup \{k^r\}$ as multisets for some r. Thus $S(\sigma, \bar{\sigma}_i(j)) = S(\sigma', \bar{\sigma}'_i(j))$. This can be proven formally, but we do not do so here.

A similar argument also shows that

$$T(\tau, \bar{\tau}_i(j)) = |\{\, l \in [\bar{\tau}_i(j)] : \tau_l > i \,\}|$$

and

$$T(\tau', \bar{\tau}'_i(j)) = |\{\, l \in [\bar{\tau}'_i(j)] : \tau'_l > i \,\}|$$

differ by the number of k's in τ which appear before the $\bar{\tau}_i(j)$ position, and this is given by w, where w satisfies $\delta_w < \bar{\tau}'_i(j) + w < \delta_{w+1}$. Thus the $q(i)$ given by equation (7) satisfies $q_i(j) = q'_i(j) + w$ where $\delta_w < q'_i(j) + w < \delta_{w+1}$. Therefore the $q_i(j)$, and hence also the $q(i)$, are the same as those produced by step ($\psi 4$) of the bijection.

It remains to show that $q(k)$ is the same as in our original bijection. But this is easy: we have $S(\sigma, \bar{\sigma}_k(j)) = |\{\, l \in [\bar{\sigma}_k(j)] : \sigma_l \leqslant k \,\}| = \bar{\sigma}_k(j)$ and $T(\tau, \bar{\tau}_k(j)) = |\{\, l \in [\bar{\tau}_k(j)] : \tau_l > k \,\}| = 0$, so $q(k) = \{\, \bar{\sigma}_k(j) : j \in [a_k] \,\} = \sigma^{-1}(k)$ as required.

Thus this really is another description of $\psi_{\mathbf{a}}$.

Next, given $p \in P_n$, we can calculate $\phi_n(p) = (s, t)$ in the following way. We already know that $t = \pi_n(p)$, the output of p. To find s, we use a modified method of parking cars, which we will call *Boston parking*. As in the regular scenario, the cars wish to park on our one-way street, arriving in the same order as before. But now, when a car arrives, it *insists* on parking in its preferred space. If this space is empty, it simply parks there. If not, it displaces the car currently there to the next space, possibly setting off a chain of displacements until some car is pushed into an empty space or beyond the end of the row of spaces. A function $p : [n] \to [n]$ is called a *Boston parking function* if no car is displaced beyond the n-th space. It is trivial to check that a function is a Boston parking function if and only if it is a (normal) parking function, as at each step during the parking process, the same space is filled, albeit with a possibly different car. (The name is indicative of the perceived standards of driving etiquette in Boston; the legal aspects of this algorithm will be left to those better versed in that subject!)

As an example, consider the parking function 3, 1, 4, 4, 3, 2 in P_6. Car 1 parks in space 3. Then car 2 parks in space 1 and car 3 parks in space 4. When car 4 arrives, it pushes car 3 over to space 5 and parks in space 4. Subsequently, car 5 pushes cars 1, 3 and 4 along one space, parking itself in space 3. Finally, car 6 parks in space 2, which was still empty. The permutation of Boston-parked cars is 265143. Under the usual rules for parking cars, the permutation obtained is 261345. Therefore, $\phi_6(314432) = (265143, 261345)$.

This alternative description of ϕ_n can easily be extended to the valet functions

and multiset case. Here, each valet parks each of his cars following the Boston parking rules. However, unlike normal parking, the output here does depend upon the ordering of each $p(i)$. We require that the elements of each $p(i)$ are ordered in increasing order so that, for example, the cars of valet k end up parked in the spaces given by $p(k)$.

The proof that this bijection is the same as ϕ_n or $\phi_{\mathbf{a}}$ is then straightforward by induction, once we note that the breakpoints of Boston parking functions are identical to those of normal parking functions, and that $d_w - w$ is a breakpoint of p' for each w.

10 Tree inversions

Using the alternative description of the bijection ϕ_n, we can give an interpretation for allowable pairs of the inversion enumerator for trees, $I_n(x)$, which was first described by Mallows and Riordan [15]. An *inversion* in a rooted labelled tree is a pair (b, a) with $b > a$ for which the (unique) path from the root to vertex a passes through b. The coefficient of x^k in $I_n(x)$ is the number of trees on $[n]_0$ rooted at 0 with k inversions.

Kreweras [13] used his bijection between parking functions on $[n]$ and labelled trees on $[n]_0$ to prove that the coefficient of x^k in $I_n(x)$ is the number of parking functions on $[n]$ with k probes. (See section 3 above for the definition of a probe.) To interpret $I_n(x)$ for allowable pairs, we define an *inversion* of an allowable pair (σ, τ) to be a pair $(b, a) \in [n] \times [n]$ with $b > a$, where b appears before a in σ but after a in τ. It follows that the number of inversions of (σ, τ) is the number of inversions of σ (defined in the usual sense for a permutation) minus the number of inversions of τ. We show that our bijections ϕ_n and ψ_n map parking functions with k probes to allowable pairs with k inversions and vice versa.

Theorem 10.1. *The map ψ_n maps allowable pairs with k inversions to parking functions with k probes.*

Proof. Let $p = \psi_n(\sigma, \tau)$ and denote the number of probes of car i by $k(i)$. We prove the theorem by showing that for each i, the number of inversions of (σ, τ) of the form (i, j) is equal to $k(i)$. We use our alternative description of ψ_n in this proof.

We say that j moved after i if j appears before i in σ, but after i in τ; similarly, we say that j moved before i if j appears after i in σ, but before i in τ. It is clear that if j moved after i, then $j > i$, and if j moved before i, then $j < i$.

Given $(\sigma, \tau) \in Q_n$, note that

$$\begin{aligned}\tau^{-1}(i) = {} & |\{\, j \in [n] : j > i \text{ and } j \text{ is before } i \text{ in } \sigma \,\}| \\ & - |\{\, j \in [n] : j > i \text{ and } j \text{ moved after } i \,\}| \\ & + |\{\, j \in [n] : j = i, \text{ or } j < i \text{ and } j \text{ is before } i \text{ in } \sigma \,\}| \\ & + |\{\, j \in [n] : j < i \text{ and } j \text{ moved before } i \,\}|.\end{aligned}$$

From the results of section 9 above, we have $p(i) = S(\sigma, \sigma^{-1}(i)) + T(\tau, \tau^{-1}(i))$. By some simple manipulations, we see that

$$\begin{aligned}S(\sigma, \sigma^{-1}(i)) &= |\{\, l \in [\sigma^{-1}(i)] : \sigma_l \leqslant i \,\}| \\ &= |\{\, j \in [n] : j = i, \text{ or } j < i \text{ and } j \text{ is before } i \text{ in } \sigma \,\}|\end{aligned}$$

and

$$\begin{aligned}T(\tau, \tau^{-1}(i)) &= |\{\, l \in [\tau^{-1}(i)] : \tau_l > i \,\}| \\ &= |\{\, j \in [n] : j > i \text{ and } j \text{ is before } i \text{ in } \tau \,\}| \\ &= |\{\, j \in [n] : j > i \text{ and } j \text{ is before } i \text{ in } \sigma \,\}| \\ &\qquad - |\{\, j \in [n] : j > i \text{ and } j \text{ moved after } i \,\}|,\end{aligned}$$

since anything greater than i and before it in τ must have been before it in σ also. We then deduce that

$$\begin{aligned}\tau^{-1}(i) &= S(\sigma, \sigma^{-1}(i)) + T(\tau, \tau^{-1}(i)) \\ &\qquad + |\{\, j \in [n] : j < i \text{ and } j \text{ moved before } i \,\}| \\ &= p(i) + |\{\, j \in [n] : j < i \text{ and } j \text{ moved before } i \,\}| \\ &= p(i) + |\{\, j \in [n] : (i, j) \text{ is an inversion of } (\sigma, \tau) \,\}|.\end{aligned}$$

Recalling from section 3 that $k(i) = \tau^{-1}(i) - p(i)$ for each i, we deduce that $k(i)$ equals the number of inversions of (σ, τ) of the form (i, j), and the theorem is proven. □

Corollary 10.2. *The coefficient of x^k in $I_n(x)$ equals the number of allowable pairs in Q_n with k inversions.* □

This can also be proved directly using the recurrence for $I_n(q)$ given by Mallows and Riordan [15, p. 94] and the fact that any $(\sigma, \tau) \in Q_n$ can be written in the form $(\gamma_{i,n}\delta, \alpha n \beta)$, where (γ, α) and (δ, β) are allowable pairs, and $\gamma_{i,n}$ means γ with n inserted in the i-th position. (Note that this is different from the meaning of $\gamma_{(i,m)}$ in Atkinson and Beals [1].)

11 Comparison with other bijections

It is worth considering whether our bijection is simply the composition of a known bijection between parking functions and trees together with one between trees and allowable pairs. However, considering the parking function 3, 1, 4, 1, 5, 9, 2, 6, 5 in P_9, with $\phi_9(314159265) = (472193856, 241357896)$, we find that the trees produced by the bijections described in section 3 (not considering the family described by Françon [6]) and those produced by the bijections described in section 4 are all distinct. Thus our bijection cannot be written as a composition of any pair of the previously known bijections. It would be interesting to find some natural bijections between trees and parking functions or allowable pairs which provide such a composition.

It would also be interesting to find extensions of some of the known bijections between parking functions and trees to bijections between valet functions and k-way trees, and especially to find ones which preserve some generalisation of inversions and probes.

References

[1] M. D. Atkinson and R. Beals, *Priority queues and permutations*, SIAM J. Computing **23** (1994), 1125–1230.

[2] M. D. Atkinson, S. A. Linton, and L. A. Walker, *Priority queues and multisets*, Electron. J. Combin. **2** (1995), no. Research Paper 24, 18pp.

[3] M. D. Atkinson and M. Thiyagarajah, *The permutational power of a priority queue*, BIT **33** (1993), 2–6.

[4] M. D. Atkinson and Louise Walker, *Enumerating k-way trees*, Information Processing Letters **48** (1993), 73–75.

[5] D. Foata and J. Riordan, *Mappings of acyclic and parking functions*, Æquationes Math. **10** (1974), 10–22.

[6] Jean Françon, *Acyclic and parking functions*, J. Comb. Th. (A) **18** (1975), 27–35.

[7] I. M. Gessel and B. E. Sagan, *The Tutte polynomial of a graph, depth-first search, and simplicial complex partitions*, The Foata Festschrift, Electron. J. Combin. **3** (1996), no. 2, Research Paper 9, 36pp.

[8] I. M. Gessel and K.-Y. Wang, *A bijective approach to the permutational power of a priority queue*, unpublished.

[9] M. Golin and S. Zaks, *Labelled trees and pairs of input-output permutations in priority queues*, Graph-Theoretic Concepts in Computer Science (Herrsching, 1994), Lecture Notes in Comput. Sci., no. 903, Springer, Berlin, 1995, pp. 282–291.

[10] Mark D. Haiman, *Conjectures on the quotient ring by diagonal invariants*, Journal of Algebraic Combinatorics **3** (1994), 17–76.

[11] D. E. Knuth, *Sorting and Searching*, 2nd ed., The Art of Computer Programming, vol. 3, Addison–Wesley, Reading, MA, 1998.

[12] A. G. Konheim and B. Weiss, *An occupancy discipline and applications*, SIAM J. Applied Math. **14** (1966), 1266–1274.

[13] G. Kreweras, *Une famille de polynômes ayant plusieurs propriétés énumeratives*, Periodica Mathematica Hungarica **11** (1980), 309–320.

[14] Joseph P. S. Kung and Catherine Yan, *Gončarov polynomials and parking functions*, preprint.

[15] C. L. Mallows and J. Riordan, *The inversion enumerator for labeled trees*, Bull. Amer. Math. Soc. **74** (1968), 92–94.

[16] P. Moszkowski, *Arbres et suites majeures*, Periodica Mathematica Hungarica **20** (1989), no. 2, 147–154.

[17] W. W. Peterson, *Addressing for random access storage*, IBM J. Res. Develop. **1** (1957), 130–146.

[18] J. Riordan, *Ballots and trees*, J. Comb. Th. **6** (1969), 408–411.

[19] Paul C. Rosenbloom, *The elements of mathematical logic*, Dover, New York, 1950.

[20] M. P. Schützenberger, *On an enumeration problem*, J. Comb. Th. **4** (1968), 219–221.

[21] Richard P. Stanley, *Hyperplane arrangements, intervals orders, and trees*, Proc. Nat. Acad. Sci. U.S.A. **93** (1996), no. 6, 2620–2625.

[22] ———, *Parking functions and noncrossing partitions*, The Wilf Festschrift (Philadelphia, PA, 1996), Electron. J. Combin. **4** (1997), no. 2, Research Paper 20, 14pp.

[23] ———, *Hyperplane arrangements, parking functions, and tree inversions*, Mathematical Essays in Honor of Gian-Carlo Rota (B. Sagan and R. Stanley, eds.), Birkhäuser, Boston/Basel/Berlin, 1998, pp. 259–375.