QUASI-MONOTONIC SEQUENCES:

THEORY, ALGORITHMS AND APPLICATIONS

by

A. Ehrenfeucht, J. Haemer* and D. Haussler**

CU-CS-277-84 August, 1984

All correspondence to David Haussler.

Department of Computer Science, University of Colorado, Boulder, Colorado 80309

*Interactive Systems, Inc. 351 Moraine Ave. Estes Park, CO

**Department of Mathematics and Computer Science, University of Denver, Denver, CO 80208

Quasi-Monotonic Sequences:

Theory, Algorithms and Applications

by

Andrzej Ehrenfeucht, Jeffrey Haemer* and David Haussler**

Dept. of Computer Science University of Colorado Boulder, Colo. 80302

*Interactive Systems, Inc. 351 Moraine Ave. Estes Park, Colo.

**Department of Mathematics and Computer Science University of Denver Denver, Colo. 80208

All correspondence to D. Haussler.

Author A. Ehrenfeucht gratefully acknowledges the support of NSF grant MCS-83-05245.

ABSTRACT

We present a simple algebraic theory which allows us to solve a variety of combinatorial problems, including the problem of finding convex hulls in two dimensions, the "Trip Around the Moon" problem, a version of the ballot problem, and the problem of enumerating and randomly generating ordered trees of a given size. Individual problems are solved by applying general algorithms and theorems developed within this algebraic theory.

INTRODUCTION

Imagine yourself standing between a pair of adjacent elements in a sequence of reals. If the sequence is monotonically increasing, then regardless of your exact position within the sequence, every element to your left is less than every element to your right (or vice versa). In this paper we develop the theory of a different, but related kind of sequence. These are *quasi-increasing* sequences, in which the "average" of all the elements to your left is less than the "average" of the elements to your right (or vice versa). Consider as an example an "ideal business year", in which the average of the monthly profits for the remaining months of the year always exceeds the current average.

We develop the theory of quasi-increasing sequences (and the other forms of quasi-monotonic sequences) using the general notion of an *averaging function*. An averaging function is a mapping μ from nonempty sequences over an arbitrary set into some linearly ordered range which satisfies one basic axiom: for any two sequences U and V, $\mu(UV)$ and $\mu(VU)$ must lie between $\mu(U)$ and $\mu(V)$. (Note that we do not demand that $\mu(UV) = \mu(VU)$). Many of the commonly used measures of central tendency satisfy this basic condition, and are thus averaging functions in the sense that we use this term.

With respect to a given averaging function μ, a sequence S is quasi-increasing if $\mu(U) < \mu(V)$ for every pair of nonempty sequences U and V such that $S = UV$. Quasi-nondecreasing, quasi-decreasing and quasi-nonincreasing sequences are defined analogously. Section 1 gives a brief introduction to the theory of quasi-monotonic sequences. This theory is related to the theory of Viennot factorizations (see [*Lot 83*], [*Vie 78*]]), but our basic approach is somewhat different. As in [*Vie 78*], we obtain interesting generalizations of earlier work in [*Spi 56*], and of the work on Lyndon words (see [*Lot 83*]). However, our primary concentration is on the applications of the theory to the solution of various combinatorial problems. These include the problem of finding convex

hulls in the plane (see e.g. [*Prep 79*]), the "Trip Around the Moon" problem ([*Gra 83*]), a generalized version of the ballot problem ([*Tak 67*]), and the problem of enumerating and randomly generating ordered trees of a given size (see e.g. [*Der 80*]). Detailed descriptions of these problems can be found in Sections 2.3, 3.2, 3.4 and 3.5 respectively. While these problems are certainly not new or unsolved, until this point they have not been cast and solved within a general algebraic framework. Sections 2 and 3 are devoted to this task. Each begins with a general result in the theory of quasi-monotonic sequences, followed by a development of an algorithm based on this result with explicit applications.

In Section 2, we demonstrate that for any fixed averaging function μ, every sequence can be uniquely decomposed into a series of maximal quasi-increasing segments, called *upward trends* (Theorem 2.1.1). We give a general algorithm for computing this decomposition, prove that it is correct and demonstrate that it is optimal (linear time) for a certain class of averaging functions which we call *constant time merging*. Using a suitable constant time merging averaging function, the convex hull of a sorted sequence of points in the plane can be viewed as a pair of decompositions of the sequence of line segments between adjacent points, where one decomposition gives the upward trends and the other gives the downward trends. Since we have a linear time decomposition algorithm for this averaging function, this gives a linear time algorithm for finding the convex hull of a set of points sorted on one coordinate, and an $O(n \log n)$ algorithm if initial sorting is required. Both algorithms are optimal ([*Yao 79*]).

In Section 3 we demonstrate that every sequence has a cyclic conjugate that is quasi-nondecreasing (Theorem 3.1.2). We give an algorithm for finding this cyclic conjugate which is also optimal for constant time merging averaging functions. This algorithm can be used to solve the "Trip Around the Moon" problem, variants of which are discussed in [*Tak 67*] and [*Duo 80*] in the context of queuing theory and data storage and retrieval techniques for magnetic bubble memories.

Using a canonical mapping from ordered trees to sequences given in [*Read 72*], an extension of the the above result (Theorem 3.3.4) can also be used to obtain formulas enumerating the number of various types of ordered trees by size ([*Der 80*]), and the above algorithm can be used to randomly generate ordered trees of various sizes and types. These results are given in Section 3.4. In addition to these applications, we can also obtain solutions to some generalized forms of the classic ballot problem (see Section 3.5).

SECTION 1. BASICS

1.1 Notation

Throughout this paper, italicized upper-case letters denote finite sequences, and the corresponding lower-case letters denote their elements. Thus a typical sequence is denoted $S = s_1 \cdots s_n$. The *length* of S is n, denoted $|S|$. Sets will be denoted with upper case Greek letters. If Ω is a finite set, then $|\Omega|$ denotes the cardinality of Ω. For any set Σ, Σ^+ denotes the set of all nonempty sequences formed from the elements of Σ. If U and V are sequences, UV denotes the sequence resulting from the concatenation of U and V, and U^k denotes the sequence resulting from the concatenation of U with itself k times. If $S = UWV$, then W is a *segment* of S. If, in addition, U is the empty sequence, then W is a *prefix* of S and if V is the empty sequence, then W is a *suffix* of S. Any segment of S is *proper* if it is not empty and it is not all of S.

1.2 Averaging Systems

The basic framework underlying the theory of quasi-monotonic sequences can be described as follows.

Definition. Let Σ and Γ be arbitrary sets, the latter being linearly ordered by a relation $\leq$. Let μ be and arbitrary function from Σ^+ into Γ which satisfies the following, where U and V are arbitrary sequences in Σ^+.

Interpolation Property.

If $\mu(U) < \mu(V)$ then $\mu(U) < \mu(UV)$, $\mu(VU) < \mu(V)$ and
if $\mu(U) = \mu(V)$ then $\mu(U) = \mu(UV) = \mu(VU) = \mu(V)$.

Such a function μ is called an *averaging function* and the system Σ, Γ, $\leq$, μ is called an *averaging system.* ▪

This general notion of an averaging function encompasses several measures of central tendency which are commonly used, and some not so commonly used. The following are a few examples of averaging functions.

Definition. Given a sequence of real numbers $S = s_1 \cdots s_n$, the *arithmetic mean* of S is $\frac{\sum_{i=1}^{n} s_i}{n}$. The *geometric mean* of S is $(s_1 s_2 \cdots s_n)^{\frac{1}{n}}$. The *harmonic mean* of S is $\frac{n}{\sum_{i=1}^{n} \frac{1}{s_i}}$. We restrict the geometric and harmonic means to sequences

of positive reals. For any real number $\alpha > 0$, the *α-weighted mean* of S is $\frac{s_1 + \alpha s_2 + \alpha^2 s_3 + \cdots + \alpha^{n-1} s_n}{1 + \alpha + \alpha^2 + \cdots + \alpha^{n-1}}$. For any sequence of pairs of real numbers $T = (x_1, y_1) \cdots (x_n, y_n)$ where $x_i > 0$, $1 \le i \le n$, the *gradient mean* of T is $\frac{\sum_{i=1}^{n} y_i}{\sum_{i=1}^{n} x_i}$.

■

Note that the arithmetic mean is a special case of the α-weighted mean with $\alpha = 1$, and (essentially) a special case of the gradient mean with $x_i = 1$ for all i, $1 \le i \le n$. Another useful special case of the α-weighted mean is obtained by taking α to be an infinitesimal. For μ defined in this manner, given nonempty sequences of reals U and V we have $\mu(U) < \mu(V)$ if and only if UV lexicographically precedes VU, i.e. if and only if $UV = XaZ$ and $VU = XbZ'$ where a and b are reals with $a < b$ and X, Z, Z' are (possibly empty) sequences of reals. We will call this the *lexicographic mean.* This is a good example of a mean which depends on the order of the elements in the sequence, in contrast to the other functions given above.

The fact that all of the above functions are averaging functions rests primarily on one elementary arithmetic result.

Lemma 1.2.1. For any real numbers a, b, c, d with b, $d > 0$,

(1) If $\frac{a}{b} < \frac{c}{d}$ then $\frac{a}{b} < \frac{a+c}{b+d} < \frac{c}{d}$ and

(2) if $\frac{a}{b} = \frac{c}{d}$ then $\frac{a}{b} = \frac{a+c}{b+d} = \frac{c}{d}$.

Proof. This follows easily from well-known arithmetic rules for manipulating fractions. ■

Lemma 1.2.2. The Interpolation Property holds for all of the above means.

Proof. That the arithmetic, harmonic and gradient means satisfy the Interpolation Property (under the restrictions given in their definitions) follows directly from the above lemma. In the case of the α-weighted mean, we notice that if $X = x_1 \cdots x_n$ and $Y = y_1 \cdots y_m$, then $\mu(XY)$ is

$$\frac{(x_1 + \cdots + \alpha^{n-1} x_n) + \alpha^n (y_1 + \cdots + \alpha^{m-1} y_m)}{(1 + \cdots + \alpha^{n-1}) + \alpha^n (1 + \cdots + \alpha^{m-1})}.$$

Hence again we can use the above lemma.

That the geometric mean satisfies the Interpolation Property actually follows from the fact that the arithmetic mean satisfies this property. This is because the logarithm of the geometric mean of a sequence of positive reals is the arithmetic mean of their logarithms, and the logarithm is monotonic. ■

Since it is our intention to proceed as rapidly as possible to the theory of quasi-monotonic sequences and its applications, we will not give an extensive axiomatic treatment of the theory of averaging systems here. However we will pause to note a few general properties of averaging systems which will be useful in what follows. From this point on, μ will denote an arbitrary averaging function, and all sequences will be assumed to be sequences over the domain of μ (i.e. nonempty sequences) unless otherwise noted.

One property of averaging systems which agrees well with our intuitive notion of an "average" is the following.

Lemma 1.2.3. (Balancing Lemma) Let $U_1, U_2, \ldots, U_k, V_1, V_2, \ldots, V_l$ be sequences, where $k, l > 0$. If $\mu(U_i) \leq \mu(V_j)$ for all i and j, $1 \leq i \leq k$ and $1 \leq j \leq l$, then $\mu(U_1 \cdots U_k) \leq \mu(V_1 \cdots V_l)$. If in addition $\mu(U_i) < \mu(V_j)$ for some i and j, $1 \leq i \leq k$ and $1 \leq j \leq l$, then $\mu(U_1 \cdots U_k) < \mu(V_1 \cdots V_l)$.

Proof. Let $\alpha = \max_{1 \leq i \leq k} \mu(U_i)$ and let $\beta = \min_{1 \leq j \leq l} \mu(V_j)$. By the Interpolation Property (repeatedly), it follows that $\mu(U_1 \cdots U_k) \leq \alpha$ and $\mu(V_1 \cdots V_l) \geq \beta$. Since by our basic assumption above, $\alpha \leq \beta$, it follows that $\mu(U_1 \cdots U_k) \leq \mu(V_1 \cdots V_l)$. If the additional assumption above is also valid, then a similar argument shows that $\mu(U_1 \cdots U_k) < \mu(V_1 \cdots V_l)$. ▪

Another useful property of averaging systems can be derived from the fact that Γ, the range of μ, is linearly ordered. While we will occasionally make tacit use of this fact, for the most of our results we need only refer to the following.

Lemma 1.2.4. (Strong Interpolation Property)
For any sequences U and V, the following are equivalent:

(1). $\mu(U) < \mu(V)$,
(2). $\mu(U) < \mu(UV)$,
(3). $\mu(U) < \mu(VU)$,
(4). $\mu(UV) < \mu(V)$ and
(5). $\mu(VU) < \mu(V)$,

and the following are equivalent:

(a). $\mu(U) = \mu(V)$,
(b). $\mu(U) = \mu(UV)$ and
(c). $\mu(U) = \mu(VU)$.

Proof. That (1) implies (2)-(5) and (a) implies (b) and (c) is precisely the content of our basic axiom, the Interpolation Property. For the reverse implications, e.g. (2) -> (1), (3) -> (1), etc. , we note that since the range of μ is linearly ordered, we must have either $\mu(U) < \mu(V)$, $\mu(U) = \mu(V)$ or $\mu(U) > \mu(V)$. Yet

these latter two relationships violate conditions (2)-(5), by the Interpolation Property, and the first and third relations violate conditions (b) and (c) by the same property. ▪

1.3 Quasi-Monotonic Sequences

In this general framework we have outlined, the notion of a quasi-monotonic sequence can be given as follows.

Definition. A sequence S is *quasi-increasing (quasi-nondecreasing)* if $\mu(U) < \mu(V)$ $(\mu(U) \leq \mu(V))$ for all nonempty sequences U, V such that $S = UV$. *Quasi-decreasing* and *quasi-nonincreasing* sequences are defined analogously. S is *quasi-monotonic* if it is a sequence of any of these four types. ▪

Example. If μ is the arithmetic mean, then

2 1 3 4 is quasi-increasing,

3 4 2 1 is quasi-decreasing,

1 4 2 3 is quasi-non-decreasing,

3 2 4 1 is quasi-non-increasing and

1 4 3 2 is none of the above. ▪

When μ is the lexicographic mean, the set of quasi-increasing sequences is the set of *Lyndon words* over the reals (see e.g. [*Lot 83*]).

Two useful variants of the definition of a quasi-monotonic sequence are given in the following lemma. Here, and in several subsequent lemmas, we give only the quasi-increasing and/or quasi-nondecreasing versions, since the other cases follow by a similar argument, by simply reversing the sense of the inequalities.

Lemma 1.3.1. (Prefix/Suffix Lemma) Let S be a sequence. The following are equivalent.

(i). S is quasi-increasing (quasi-nondecreasing).

(ii). $\mu(U) < \mu(S)$ $(\mu(U) \leq \mu(S))$ for every proper prefix U of S.

(iii). $\mu(S) < \mu(V)$ $(\mu(S) \leq \mu(V))$ for every proper suffix V of S.

Proof. This follows directly from the definition of a quasi-increasing (quasi-nondecreasing) sequence, using the Strong Interpolation Property. ▪

It follows that for a quasi-increasing sequence S, $\mu(U) < \mu(V)$ for any proper prefix U and proper suffix V of S, even if they overlap or are separated by some nonempty middle segment of S.

The four classes of quasi monotonic sequences have intersection properties similar to those of normal monotonic sequences, as is demonstrated in the following lemma.

Definition. A sequence $S = s_1 \cdots s_n$ is *constant* if $\mu(s_i) = \mu(s_j)$ for all i and j, $1 \le i, j \le n$. ▪

Lemma 1.3.2. (Constant Sequence Lemma)

(1). A sequence is both quasi-nondecreasing and quasi-nonincreasing if and only if it is constant.

(2). A sequence is both quasi-increasing and quasi-decreasing if and only if it has only one element.

Proof. If S is both quasi-nondecreasing and quasi-nonincreasing, then by the Prefix/Suffix Lemma above, for any proper prefix U of S, $\mu(U) \le \mu(S)$ and $\mu(U) \ge \mu(S)$, i.e. $\mu(U) = \mu(S)$. Hence by the Strong Interpolation Property (repeatedly),it follows that $\mu(s_i) = \mu(S)$ for all i, $1 \le i \le n$, and thus S is constant. For the second part, if S is both quasi-decreasing and quasi-increasing then for any proper prefix U of S, $\mu(U) < \mu(S)$ and $\mu(U) > \mu(S)$, hence S has no proper prefixes, i.e. S has only one element. ▪

We close this introductory section by briefly examining the conditions under which quasi-monotonic sequences can be combined to form larger quasi-monotonic sequences. As above, we will restrict our attention to quasi-increasing and quasi-nondecreasing sequences. Our first lemma deals with sequences formed by concatenating quasi-nondecreasing sequences.

Lemma 1.3.3. (Construction Lemma) Let $S_1, \ldots, S_k$ be quasi-nondecreasing sequences, where $k > 1$, and let $S = S_1 \cdots S_k$. S is quasi-increasing (quasi-nondecreasing) if and only if $\mu(S_1 \cdots S_i) < \mu(S)$ ($\mu(S_1 \cdots S_i) \le \mu(S)$) for all i, $1 \le i < k$.

Proof. We will prove only the "quasi-increasing part" of this result, since the other part is analogous. Further, since the "only if" implication of this part follows directly from the Prefix/Suffix Lemma, we need only verify the "if" implication.

Using the Prefix/Suffix Lemma, it suffices to show that $\mu(S_1 \cdots S_i U) < \mu(S)$ for any i, $0 \le i < k$, and any proper prefix U of S_{i+1}. If $i = 0$, then $\mu(S_{i+1}) = \mu(S_1) < \mu(S)$ since $k > 1$. Furthermore, since S_1 is quasi-nondecreasing, $\mu(U) \le \mu(S_1)$. Hence $\mu(S_1 \cdots S_i U) = \mu(U) < \mu(S)$, establishing the result. Thus we may assume that $i > 0$. Now if $\mu(U) \le \mu(S)$ then since $\mu(S_1 \cdots S_i) < \mu(S)$ as well, we have $\mu(S_1 \cdots S_i U) < \mu(S)$ by the Balancing

Lemma. Hence we may also assume that $\mu(S) < \mu(U)$. Let V be the sequence such that $S_{i+1} = UV$. Since S_{i+1} is quasi-nondecreasing, $\mu(U) \leq \mu(V)$. Hence $\mu(S) < \mu(V)$. Now since either $S_{i+2} \cdots S_k$ is empty or $\mu(S) < \mu(S_{i+2} \cdots S_k)$ (by the Strong Interpolation Property), we have $\mu(S) < \mu(VS_{i+2} \cdots S_k)$ by the Balancing Lemma. Thus since $S = S_1 \cdots S_i UVS_{i+2} \cdots S_k$, by the Strong Interpolation Property, $\mu(S_1 \cdots S_i U) < \mu(S)$. ▪

A useful special case of the above result is the following.

Corollary 1.3.4. (Construction Corollary) If $S_1, \ldots, S_k$ are quasi-nondecreasing sequences, where $k > 1$, and $\mu(S_1) < \mu(S_2) < \cdots < \mu(S_k)$ $(\mu(S_1) \leq \mu(S_2) \leq \cdots \leq \mu(S_k))$ then $S_1 \cdots S_k$ is quasi-increasing (quasi-nondecreasing).

Proof. This follows directly from the above lemma using the Balancing Lemma. ▪

We also consider sequences obtained by overlapping quasi-nondecreasing sequences.

Lemma 1.3.5. (Overlap Lemma) Let T, U, and V be sequences with U nonempty. If TU and UV are quasi-increasing (quasi-nondecreasing), then $S = TUV$ is quasi-increasing (quasi-nondecreasing).

Proof. Again we prove only the result for the quasi-increasing case, the other case being entirely analogous.

Let $S = XY$, where X and Y are nonempty sequences. We show that $\mu(X) < \mu(Y)$. Consider three cases.

(a) $X = TL$ and $Y = RV$ for some nonempty L and R, (thus $U = LR$).
Since TU is quasi-increasing, by the Prefix/Suffix lemma, $\mu(TL) < \mu(TU) < \mu(U)$. Similarly, $\mu(U) < \mu(UV) < \mu(RV)$. Thus, $\mu(X) = \mu(TL) < \mu(RV) = \mu(Y)$.

(b) $X = L$ and $Y = RUV$ for some nonempty L, (thus $T = LR$).
Following the reasoning used in (a), $\mu(L) < \mu(TU) < \mu(U), \mu(RU)$. Also $\mu(U) < \mu(UV) < \mu(V)$. Thus, since $\mu(L) < \mu(RU)$ and $\mu(L) < \mu(V)$, by the Balancing Lemma $\mu(X) = \mu(L) < \mu(RUV) = \mu(Y)$.

(c) $X = TUL$ and $Y = R$ for some nonempty R, (thus $V = LR$).
This case is a mirror image of case (b), and has a parallel proof. ▪

SECTION 2. TRENDS

2.1 The Decomposition Theorem

In Lemma 1.3.3 (the Construction Lemma) we have considered the conditions under which quasi-increasing sequences can be concatenated to form larger quasi-increasing sequences. In this section, we consider the related problem of how we can decompose an arbitrary sequence into quasi-increasing segments. A related approach to decompositions of this type is given in [*Vie 78*].

Since each sequence element is itself a quasi-increasing segment, we restrict our attention to segments which are quasi-increasing and of maximal length.

Definition. Given a sequence $S = s_1 \cdots s_n$, a segment $U = s_i \cdots s_j$, $1 \le i \le j \le n$, is *maximal quasi-increasing* if U is quasi-increasing and no extension $s_h \cdots s_k$ of U, where $1 \le h \le i \le j \le k \le n$ and $h < i$ or $j < k$, is quasi-increasing. A maximal quasi-increasing segment is called an *upward trend.* Downward trends are defined analogously. A sequence of sequences $S_1, \ldots, S_k$ is a *decomposition of S into upward (downward) trends* if $S = S_1 \cdots S_k$ and S_i is an upward (downward) trend of S for each i, $1 \le i \le k$. ▪

Example. If μ is the arithmetic mean, then the sequence 1 2 1 5 4 3 1 2 0 can be decomposed into upward trends as 1 2 1 5 4 3, 1 2, 0 and into downward trends as 1, 2 1, 5 4 3 1 2 0 . ▪

As in the previous section, we will state many of our results only in their quasi-increasing and/or quasi-nondecreasing versions. Unless otherwise indicated, the word *trend* indicates an upward trend and a *decomposition of S* indicates a decomposition of S into upward trends. Our main result is the following.

Theorem 2.1.1. (Decomposition Theorem) Any sequence S can be uniquely decomposed into upward trends, and every upward trend of S is a member of this decomposition.

Proof. Let $S = s_1 \cdots s_n$. For each element s_i of S, find the maximal quasi-increasing segment of S which contains s_i. This segment is unique by the Overlap Lemma. Let $S_1, \ldots, S_k$ be the list of distinct segments found by successively considering elements $s_1, \ldots, s_n$. Again by the Overlap Lemma, these must form a decomposition of S into upward trends, and every trend of S must appear in this decomposition. ▪

When μ is the lexicographic mean, the decomposition into upward trends is known as the Lyndon factorization. When μ is the arithmetic mean, this decomposition has been called Spitzer's factorization (see [*Lot 83*], [*Spi 56*]).

The general relationship among the segments in the decomposition of S is outlined in the next lemma.

Lemma 2.1.2. (Trend Mean Lemma)

(1) $S_1, \ldots, S_k$ is the decomposition of S into upward trends if and only if $S = S_1 \cdots S_k$, S_i is quasi-increasing for all i, $1 \le i \le k$, and $\mu(S_1) \ge \mu(S_2) \ge \cdots \ge \mu(S_k)$.

(2) Let $S_1, \ldots, S_k$ be the decomposition of S into upward trends. If S is quasi-nondecreasing, then $\mu(S_1) = \mu(S_2) = \cdots = \mu(S_k) = \mu(S)$, otherwise $\mu(S_1) > \mu(S) > \mu(S_k)$.

Proof. ad (1). If $S_1, \ldots, S_k$ is the decomposition of S, then by definition we must have $S = S_1 \cdots S_k$ and S_i quasi-increasing, $1 \le i \le k$. Further, we cannot have $\mu(S_i) < \mu(S_{i+1})$ for any i, $1 \le i < k$, for by the Construction Corollary, this would imply that S_iS_{i+1} is quasi-increasing, contradicting the maximality of the trends. Hence $\mu(S_1) \ge \mu(S_2) \ge \cdots \ge \mu(S_k)$. For the other direction, assume that these three conditions hold. Suppose that S_i is not maximal for some i, $1 \le i \le k$. Thus $S = ULS_iRV$ where U,L,R,V are sequences (either L or R may be empty, but not both) and LS_iR is maximal quasi-increasing. From the Overlap Lemma, it follows that $LS_iR = S_h \cdots S_l$ for some $1 \le h \le i \le l \le k$, where $h < i$ or $i < l$. However, then by the Prefix/Suffix Lemma, we must have $\mu(S_h) < \mu(S_l)$, a contradiction to the third condition of (1). Hence each S_i must be maximal, and thus $S_1, \ldots, S_k$ is the decomposition of S.

ad (2). If S is quasi-nondecreasing, then by the Prefix/Suffix Lemma, $\mu(S_1) \le \mu(S) \le \mu(S_k)$. Hence from part (1), $\mu(S_1) = \mu(S_2) = \cdots = \mu(S_k) = \mu(S)$. On the other hand, if S is not quasi-nondecreasing, then it cannot be the case that $\mu(S_1) = \mu(S_2) = \cdots = \mu(S_k)$ (by the Construction Corollary), hence we must have $\mu(S_1) > \mu(S_k)$ by part (1). Again from part (1), using the Balancing Lemma, it follows further that $\mu(S_1) > \mu(S) > \mu(S_k)$. ∎

2.2 The Collect-and-Merge Algorithm

We now turn our attention to the problem of computing the decomposition of a given sequence. We will present an algorithm which produces the decomposition of a given sequence on-line in linear time, under certain general assumptions concerning the computation of μ. We will use a model of computation in which all integers and real numbers to arbitrary precision occupy constant space, and all normal arithmetic operations, including addition, subtraction, multiplication and division, on these numbers take constant time. This is known as the *uniform cost RAM model (see e.g. [Aho 74])*. The key element in our algorithm is the following abstract data type.

Definition. A *block* is a data type which represents specific information about an arbitrary segment of a sequence $S = s_1 \cdots s_n$. This data type supports the following functions, where b, b_1, b_2 are arbitrary blocks representing segments T, U, V respectively.

(1). *location* (b) returns the index in S of the first letter in the segment T.

(2). *length* (b) returns the length of T.

(3). $\mu(b)$ returns $\mu(T)$.

(4). *merge* (b_1, b_2) returns a block b representing the segment UV if the segment V occurs immediately following U in S, otherwise it returns some special error value.

(5). *makeblock* (T) returns a block representing the segment T.

An averaging system and its associated averaging function are *constant time merging* if for any sequence S, there is an implementation of the data type "block" for segments of S in which each block occupies constant space, each of the functions (1)-(4) defined above take constant time, and for any segment T of length 1, *makeblock* T takes constant time. ▪

Lemma 2.2.1. Using the uniform cost RAM model, the arithmetic and gradient means are constant time merging.

Proof. If μ is the arithmetic mean, then a block representing a segment T can be implemented as a record which consists of the index in S of the first element of T, the length of T, and the real number which gives the sum of the elements of T. It is clear that under the model of computation we are using, this data structure occupies constant space, and all of functions associated with a block can be computed from it in the required time. For the gradient mean, we can use a similar data structure which includes both the numerator and the denominator of the fraction that defines $\mu(S)$. ▪

Our algorithm to find the decomposition of a sequence $S = s_1 \cdots s_n$ will create a stack of blocks $b_1, \ldots, b_k$ representing the trends $S_1, \ldots, S_k$ of this decomposition. This will be accomplished by successively computing the stacks representing the decomposition of $s_1 \cdots s_i$ for $i = 1$ to n. Thus we will need a procedure to update an existing stack of trends when a new element is added on the end of the sequence. We give this procedure in the following general format.

The Procedure Coalesce (Q,b)

input: a sequence SB with B quasi-increasing, a stack of blocks $Q = b_1, \ldots, b_k$ (with b_k at the top) representing the decomposition $S_1, \ldots, S_k$ of S into upward trends and a block b representing the

segment B. (We allow the possibility that Q is the empty stack and S is the empty sequence).

output: a stack of blocks $Q = t_1, \ldots, t_l$ (with t_l at the top) representing the decomposition $T_1, \ldots, T_l$ of SB into upward trends.

```
begin
    while Q is not empty and μ(top(Q)) < μ(b) do
    begin
        pop the top block b_top from Q;
        let b = merge(b_top, b);
    end
    push b onto Q;
end.
```

Lemma 2.2.2. The procedure *coalesce* is correct.

Proof. If Q is empty then the while loop of *coalesce* is not executed, and it is obvious that the procedure is correct. Otherwise, we claim the while loop has the following invariant.

(1) $Q = b_1, \ldots, b_j$, for some j, $0 \le j \le k$, where the segment represented by b_h is quasi-increasing, $1 \le h \le j$, and $\mu(b_1) \ge \mu(b_2) \ge \cdots \ge \mu(b_j)$,

(2) the segment represented by b is quasi-increasing and

(3) SB is represented by $b_1 \cdots b_j b$.

It is easily verified that this invariant holds before the first execution of the loop. In this case b represents B, which is quasi-increasing by assumption, and $b_1, \ldots, b_j$ represents the decomposition of S. Thus the segment represented by b_h is quasi-increasing for all h, $1 \le h \le j$, $b_1 \cdots b_j b$ represents SB and by the Trend Mean Lemma, $\mu(b_1) \ge \mu(b_2) \ge \cdots \ge \mu(b_j)$. That it is preserved by the execution of the loop body follows directly from the Construction Corollary, since b_j and b are merged only when $\mu(b_j) < \mu(b)$, and in this case the segment represented by $b_j b$ must be quasi-increasing. Since each time the loop is executed, the size of Q is reduced by one, the loop will terminate. Upon termination, in addition to conditions (1)-(3) we will have either

(4) Q is empty (i.e. $j=0$) or

(5) $j > 0$ and $\mu(b_j) \ge \mu(b)$.

In either case, $b_1, \ldots, b_j, b$ represents the correct decomposition for SB by the Trend Mean Lemma, and hence Q is correct following the last statement

of the procedure. ▪

The algorithm to compute the decomposition of a sequence can now be given.

The Collect-and-Merge Algorithm.
input: a nonempty sequence $S = s_1 \cdots s_n$.
output: a decomposition $S_1, \ldots, S_k$ of S into upward trends.
data structures: a stack Q of blocks.

```
begin
    let Q be empty;
    for i = 1 to n do
    begin
        let b_new = makeblock(s_i);
        coalesce(Q, b_new);
    end;
    return a list of segments represented by the elements of Q
  ordered from bottom to top;
end.
```

Given the correctness of *coalesce*, it is obvious that the Collect-and-Merge algorithm is correct. We briefly analyze the time and space requirements of this algorithm.

Theorem 2.2.3. Using the uniform cost RAM model, for any averaging system which is constant time merging, the space and time requirements of the Collect-and-Merge algorithm are $O(n)$, where n is the length of the input sequence.

Proof. It is clear that the space requirements are $O(n)$. To analyze the time requirements, let us for the moment discount the while loop in *coalesce*. What remains are the first and last statements of the algorithm (which take time $O(n)$), and a group of middle statements which constitute a loop which is executed n times and takes constant time for each execution. Hence the total time used is $O(n)$. In the course of all executions of the middle loop, n blocks are created by calling the function *makeblock*. Now consider the while loop we omitted. One execution of this loop also takes constant time. Furthermore, every time it is executed, the number of blocks in use is reduced by one. Since n blocks are introduced during the course of execution of the entire algorithm, at least one remains when the algorithm terminates, and there are no other means

of reducing the number of blocks, this implies that the while loop is executed at most $n - 1$ times. Thus the total running time of the algorithm is $O(n)$. ▪

Before continuing to the applications of the Collect-and-Merge algorithm, we pause to consider another use of the procedure *coalesce*. Let us assume that we have already computed the decompositions for two sequences S_1 and S_2. We can combine these decompositions into a single decomposition for the sequence S_1S_2 by the following procedure.

The Procedure Combine (T_1, T_2)

input: two stacks of blocks U and V representing the decompositions of S_1 and S_2 into upward trends, where U is ordered from bottom to top and V is ordered from top to bottom.

output: a sequence of blocks Q representing the decomposition of S_1S_2.

```
begin
    while V is not empty and μ(top(U)) < μ(top(V)) do
    begin
        pop the block b from the top of V;
        coalesce(U, b);
    end;
    return U (ordered from bottom to top) concatenated with V
        (ordered from top to bottom);
end.
```

By arguments similar to those given above, it is clear that this procedure is correct, and that in the worst case it takes time and space proportional to the total number of blocks in the decompositions of S_1 and S_2 for an averaging system which is constant time merging. The procedure *combine* might be used in a divide-and-conquer approach to finding decompositions. However, it is clear that since the Collect-and-Merge algorithm is already optimal for averaging systems which are constant time merging, this approach will not be useful in this case. It may be the case though, that this procedure can be used to at least improve the expected time in certain cases when the averaging system is not constant time merging. In other cases, it appears that a more direct approach, taking advantage of special features of the averaging function μ, will yield the most efficient decomposition algorithm. An example of this is Duval's decomposition algorithm for the lexicographic mean ([*Duv 83*]).

2.3 Finding Convex Hulls

As an example of the application of the Collect-and-Merge Algorithm, consider the problem of finding the convex hull of a set of points on the $x-y$ plane.

Assume that we are given a sequence of points $T = (x_0, y_0), \ldots, (x_n, y_n)$, where $n > 1$, with distinct x coordinates, sorted in increasing order on the x coordinate. The convex hull of T is the smallest (minimal area) closed convex polygon which contains all of the points of T. The vertices of this polygon form a subset of T known as the set of *extremal* points of T. It is clear that the set of extremal points of T must include the first and last points of T, and these extremal points will form a degenerate polygon only in the case that all of the points of T lie on the line between the first and last points of T. The edges of the convex hull connecting the extremal points which lie on or above the line from the first to the last point of T will be called the *upper part* of the convex hull, and those connecting the extremal points which lie on or below this line will be called the *lower part*. These sets of edges are disjoint, unless the convex hull is degenerate, in which case they are identical (we generalize this observation later).

The convex hull of T can be determined as follows. From the sequence T, derive a sequence of line segments $S = (s_1, t_1) \cdots (s_n, t_n)$ where $s_i = x_i - x_{i-1}$ and $t_i = y_i - y_{i-1}$ for $1 \le i \le n$. Since T is sorted and all points have distinct x coordinates, $s_i > 0$ for all i, $1 \le i \le n$. Let μ be the gradient mean, as defined above in Section 1. By Lemma 2.2.1, μ is an averaging function which is constant time merging.

Now consider and arbitrary segment $U = (x_k, y_k) \cdots (x_{k+l}, y_{k+l})$ of T, where $l > 1$, and the corresponding sequence of line segments $V = (s_{k+1}, t_{k+1}) \cdots (s_{k+l}, t_{k+l})$. By the Prefix/Suffix Lemma, V is quasi-increasing if and only if

$$\mu((s_{k+1}, t_{k+1}) \cdots (s_{k+i}, t_{k+i})) = \frac{\sum_{j=1}^{i} t_{k+j}}{\sum_{j=1}^{i} s_{k+j}} < \mu(V) \text{ for all } i,\ 1 \le i < l.$$

This is obviously equivalent to the condition that the slope of the line from (x_k, y_k) to (x_{k+i}, y_{k+i}) is less than the slope from (x_k, y_k) to (x_{k+l}, y_{k+l}) for all i, $1 \le i < l$, i.e. that all of the points between (x_k, y_k) and (x_{k+l}, y_{k+l}) lie below the line between these two points. Similarly, V is quasi-decreasing if and only if all intermediate points lie above the line determined by the endpoints of U. It follows easily that the decomposition of S into upward trends defines the upper

part of the convex hull of T, and that the decomposition into downward trends defines the lower part. Thus using the Collect-and-Merge algorithm, the convex hull of T can be computed on-line in linear time.

This algorithm is clearly optimal in situations where the points are given in sorted order with distinct coordinates in one dimension, e.g. in applications where the points are evaluations of a function $f(x)$ for successive values of x taken at discrete intervals. Even if two points can have the same x coordinate, we can usually get around this by perturbing the points slightly within their error range. Here, as in general, care must be taken when applying this algorithm to avoid the accumulation of round-off errors.

If the points of T are not originally given sorted on their x coordinates, then to use the Collect-and-Merge algorithm, it requires $O(n \log n)$ time to sort them, giving a total running time $O(n \log n)$. This the best possible time bound that can be achieved in this situation [*Yao 79*], and there are several algorithms which achieve it, either by sorting first and then applying a hull finding procedure (which in some cases appears to be a special case of the Collect-and-Merge algorithm, e.g. [*And 79*]), or by using divide-and-conquer techniques (e.g. [*Ben 78*]). Many of these latter algorithms are appealing because they run in $O(n)$ expected time for a variety of point distributions. Here it should be noted that the algorithm above, and some of the other techniques based on sorting will also run in $O(n)$ expected time if an $O(n)$ expected time sort can be used (see [*Mei 80*] for an example of a sort which achieves this expected time for a wide class of distributions).

2.4 The Trend Boundary Theorem

The relationship between the convex hull and the corresponding decompositions given above suggests other properties of decompositions which have not yet been explored. For example, as we mentioned above, it is intuitively obvious that the set of extremal points of $T = (x_0, y_0), \ldots, (x_n, y_n)$ between (x_0, y_0) and (x_n, y_n) on which the upper part of the convex hull of T is defined is always disjoint from the set on which the lower part is defined, unless the convex hull is degenerate. This is a general phenomenon that occurs when decompositions into upward trends are compared with decompositions into downward trends. Loosely stated, our result is that internal boundaries are never shared between elements of these decompositions, unless the sequence is constant.

Theorem 2.4.1. (Trend Boundary Theorem) Let S be a sequence decomposed into upward trends by $I_1, \ldots, I_k$ and into downward trends by $D_1, \ldots, D_l$. If $I_1 \cdots I_r = D_1 \cdots D_s$ for any r, s, $1 \le r < k$ and $1 \le s < l$, then S is constant.

Proof. Let $U = I_1 \cdots I_r = D_1 \cdots D_s$ and $V = I_{r+1} \cdots I_k = D_{s+1} \cdots D_l$. By the Trend Mean Lemma,

(1) $\mu(I_1) \ge \mu(I_2) \ge \cdots \ge \mu(I_k)$ and

(2) $\mu(D_1) \le \mu(D_2) \le \cdots \le \mu(D_l)$.

Using the Balancing Lemma, from (1) we have $\mu(U) \ge \mu(V)$ and from (2) we have $\mu(U) \le \mu(V)$. Thus $\mu(U) = \mu(V)$. However, now again using the Balancing Lemma, this implies that none of the inequalities in (1) or (2) above can be strict. Hence

(3) $\mu(I_n) = \mu(D_m) = \mu(S)$ for all $1 \le n \le k$ and $1 \le m \le l$.

Now since I_1 is quasi-increasing and $\mu(D_1) = \mu(I_1)$, D_1 cannot be a proper prefix of I_1. Similarly, I_1 cannot be a proper prefix of D_1, hence $I_1 = D_1$. Continuing in this manner, it follows that $k = l$ and $I_n = D_n$ for all $1 \le n \le k$, i.e. the upward and downward decompositions of S must be identical. Finally, since only a single element sequence can be both quasi-increasing and quasi-decreasing (Lemma 1.3.2), by (3), S must be constant. ▪

SECTION 3. CYCLIC CONJUGATES

3.1 The Rotate-and-Merge Algorithm

Many more applications of the theory of quasi-monotonic sequences can be obtained by considering the families of sequences obtained by taking all cyclic conjugates of a sequence.

Definition. Given a sequence $S = s_1 \cdots s_n$, the set of *cyclic conjugates* of S is $\{S\} \cup \{s_i \cdots s_n s_1 \cdots s_{i-1} : 1 < i \le n\}$. ▪

We will show that every sequence has a cyclic conjugate which is quasi-nondecreasing, and a cyclic conjugate which is quasi- nonincreasing. The key idea is given in the following.

Lemma 3.1.1. (Trend Rotation Lemma) Let S be decomposed into upward trends by $S_1, \ldots, S_k$. If $\mu(S_1) > \mu(S_k)$, then $T = S_2 \cdots S_k S_1$ has fewer trends than S.

Proof. Since $S_1, \ldots, S_k$ is a decomposition of S into upward trends, each S_i, $1 \le i \le k$, is a quasi-increasing sequence. Since $\mu(S_k) < \mu(S_1)$, by the Con-

struction Corollary, $S_k S_1$ is also quasi-increasing. Hence S_2, ..., S_{k-1}, $S_k S_1$ is a decomposition of T into $k-1$ quasi-increasing segments. Thus by the Overlap Lemma, the decomposition of T into upward trends cannot have more than $k-1$ members. ▪

Theorem 3.1.2. (Cyclic Conjugate Theorem) Every nonempty sequence S has a quasi-nondecreasing cyclic conjugate and a quasi-nonincreasing cyclic conjugate.

Proof. From the Trend Mean Lemma, if a sequence S is not quasi-nondecreasing, the decomposition of S has more than one trend and μ of the final trend is less than μ of the first trend. Hence the Trend Rotation Lemma implies that whenever S is not quasi-nondecreasing, a cyclic conjugate T of S with fewer trends in its decomposition can be found by rotating the first trend in the decomposition of S to the back of S. By iterating this procedure, we must eventually reach a cyclic conjugate of S which has only one trend, or one with two or more trends, such that μ of the first trend is equal to μ of the last trend. In either case, this cyclic conjugate of S will be quasi-nondecreasing. A similar argument holds for quasi-nonincreasing cyclic conjugates. ▪

The proof of this theorem also provides us with a simple algorithm for finding a quasi-nondecreasing cyclic conjugate of an arbitrary sequence S. This algorithm is presented below. We will use the abstract data type *block* introduced in the previous section and the procedures and terminology associated with it, under the assumption that these definitions have been extended to allow us treat a sequence S as if it was circular, so that we can merge segments at the end of the sequence with segments at the front. We also assume that the procedure *coalesce* has been extended from a stack of blocks to a queue of blocks in a natural manner, taking the end of this queue as the top of the stack.

The Rotate-and-Merge Algorithm.
input: a nonempty sequence $S = s_1 \cdots s_n$.
output: an index j such that $s_j s_{j+1} \cdots s_n s_1 \cdots s_{j-1}$ is quasi-nondecreasing.
data structures: a queue Q of blocks.

```
begin
    apply the Collect-and-Merge algorithm to S to obtain a decomposition
        S_1, ..., S_k of S into upward trends;
    let Q = b_1, ..., b_k be a queue of blocks representing these trends with
        front(Q) = b_1 and back(Q) = b_k;
    while μ(back(Q)) < μ(front(Q)) do
```

```
    begin
        remove the block at the front of Q and call it b;
        coalesce(Q, b);
    end;
    return location(front(Q));
end.
```

The correctness of this algorithm is easily established, as described above. Under the assumption that μ is constant time merging, and using a uniform cost RAM model as described in the previous section, the timing analysis is also easy. It is simply an extension of the analysis of the Collect-and-Merge algorithm given in Theorem 2.2.4. Again the critical factor is the total number of merges executed in during course of the computation. The same reasoning of Theorem 2.2.4 applied to the Rotate-and-Merge algorithm shows that the total number of merges executed during the first step, where the Collect-and-Merge algorithm is called, combined with those in the remaining steps is exactly $n-1$. Hence the Rotate-and-Merge algorithm is $O(n)$.

3.2 Trip Around the Moon

As an application of these results, consider the following problem, known as "Trip Around the Moon" ([*Gra 83*]).

You are to make one trip around the Moon in a circular path. At various points along this path, there are n fueling stations $t_1, \ldots, t_n$ with fuel amounts $f_1, \ldots, f_n$, such that the total amount of fuel available is sufficient to make one circular trip. You are not guaranteed however, that the amount of fuel available in each station is sufficient to cover the distance to the next station. You begin at the station of your choice with an empty fuel tank. By choosing the right starting station, can you make the entire trip without running out of fuel?

The answer to this question is always yes, independently of the given configuration of fueling stations. We can demonstrate this as follows.

Let $d_1, \ldots, d_n$ be the distances between stations, where d_i is the distance between t_i and t_{i+1}, $1 \le i < n$ and d_n is the distance between t_n and t_1. Assume that the units chosen are such that we can travel distance d with fuel amount f if and only if $f \ge d$. Let $S = (f_1, d_1) \cdots (f_n, d_n)$ and let μ be the gradient mean, defined in Section 1. Let $T = (f_i, d_i) \cdots (f_n, d_n)(f_1, d_1) \cdots (f_{i-1}, d_{i-1})$ be a quasi-nonincreasing cyclic conjugate of S, as guaranteed by the Cyclic Con-

jugate Theorem. By the basic assumption of the problem, $\mu(T) = \frac{\sum_{j=1}^{n} f_j}{\sum_{j=1}^{n} d_j} \geq 1$. Let U be any proper prefix of T. Since T is quasi-nonincreasing, $\mu(U) \geq \mu(T) \geq 1$. Hence the sum of the fuel available in the stations of U is greater than or equal to the total distance spanned by U. Since this holds for every proper prefix U of T, the trip can be made starting at station t_i.

Furthermore, since the gradient mean is constant time merging (Lemma 2.2.1), we can apply the Rotate-and-Merge algorithm to find station t_i in time proportional to the number of fuel stations. Thus we can solve the Trip Around the Moon problem in optimal time.

Notice that in our argument, we have implicitly used the fact that the distances d_i are positive, but not the fact that the fuel amounts f_i are (presumably) positive. In fact the problem has a solution even when the f_i are allowed to be negative, since the gradient mean is still an averaging function in this case (Lemma 1.2.2). Our results may be summarized as follows.

Theorem 3.2.1. For any nonempty sequence $S = (f_1, d_1) \cdots (f_n, d_n)$, where f_i, d_i are real numbers, $d_i > 0$, $1 \leq i \leq n$, and $\sum_{i=1}^{n} f_i \geq \sum_{i=1}^{n} d_i$, there is a cyclic conjugate $T = (f'_1, d'_1) \cdots (f'_n, d'_n)$ of S such that $\sum_{i=1}^{j} f'_i \geq \sum_{i=1}^{j} d'_i$ for all j, $1 \leq j \leq n$. Such a cyclic conjugate T can be found in time proportional to n, using a uniform cost RAM model of computation. ▪

A related result appears in a recent article by Dvornicich ([*Dvo 80*]) which presents some results used to derive efficient algorithms for handling data in magnetic bubble memories.

Theorem 3.2.2. (Dvornicich) Let $S = s_1 \cdots s_n$ be a sequence of real numbers such that $\sum_{i=1}^{n} s_i \geq 0$. Then there is a cyclic conjugate $T = t_1 \cdots t_n$ of S such that $\sum_{i=1}^{j} t_i \geq 0$ for all j, $1 \leq j \leq n$.

Proof. Let T be a quasi-nonincreasing cyclic conjugate of S using the arithmetic mean. Thus $\mu(t_1 \cdots tj) \geq \mu(T) = 0$ for all j, $1 \leq j \leq n$, and the result follows. ▪

The Dvornicich result can also be derived as a corollary to Theorem 3.2.1, or from the more general results presented in [*Gra 63*] and [*Tak 67*] (Theorem 2, pg.1). We have not determined if these later results can also be derived within

the framework we have presented.

3.3 Unbalanced Sequences and the Counting Theorem

We can obtain stronger results along the lines of the Cyclic Conjugate Theorem by undertaking a more detailed analysis of the structure of the set of cyclic conjugates of an arbitrary sequence with respect to μ. We take up this task presently.

Definition. Each of the cyclic conjugates of $S = s_1 \cdots s_n$ defines the same *circular sequence* S', derived by forming the letters of S into a clockwise circular arrangement with s_1 following s_n. Segments of S' will be denoted by ranges $s_i \cdots s_j$. When $i \leq j$, this corresponds to the standard notation. When $i > j$, $s_i \cdots s_j = s_i \cdots s_n s_1 \cdots s_j$. ▪

Given a circular sequence S, the set of cyclic conjugates that form it can be obtained from the set of possible cuts of S.

Definition. Given a circular sequence S formed from $s_1 \cdots s_n$, $C_S = \{c_1, \ldots, c_n\}$ is the set of *cuts* of S, where c_i is the cut between s_i and s_{i+1} for $1 \leq i < n$ and c_n is the cut between s_n and s_1. ▪

Two distinct cuts c_i and c_j in the circular sequence S formed from $s_1 \cdots s_n$ define a pair of opposing segments $s_{i+1} \cdots s_j$ and $s_{j+1} \cdots s_i$. In our basic structural result below, we express the relationship between opposing segments (with respect to μ) as a relationship between their corresponding cuts.

Definition. Given a circular sequence $S = s_1 \cdots s_n$ with cuts $C_S = \{c_1, \ldots, c_n\}$, the relation $\leq$ on C_S is defined as follows. $c_i \leq c_j$ if and only if $i = j$, or $i \neq j$ and $\mu(s_{i+1} \cdots s_j) \leq \mu(s_{j+1} \cdots s_i)$. ▪

We explore the properties of the relation $\leq$ on C_S.

Definition. Given a set A and a binary relation $\leq$ on A, $\leq$ is a *preorder* if it is reflexive and transitive. $\leq$ is a *linear preorder* if in addition, $a \leq b$ or $b \leq a$ for any a, b in A. ▪

Our basic structural result is the following.

Lemma 3.3.1. (Cut Order Lemma) For any circular sequence S, $\leq$ is a linear preorder on C_S.

Proof. Obviously $\leq$ is reflexive and since the range of μ is linearly ordered, for any i, j, $1 \leq i, j \leq n$, either $c_i \leq c_j$ or $c_j \leq c_i$ (or both). Now assume that $c_i \leq c_j$ and $c_j \leq c_k$. If i, j and k are not pairwise distinct, then it is obvious that $c_i \leq c_k$. Otherwise, we may assume without loss of generality that $i < j < k$. Let

$X = s_{i+1} \cdots s_j$, $Y = s_{j+1} \cdots s_k$ and $Z = s_{k+1} \cdots s_i$. Since $c_i \le c_j$, $\mu(X) \le \mu(YZ)$. Since $c_j \le c_k$, $\mu(Y) \le \mu(ZX)$. Thus by the Interpolation Property (twice) we have $\mu(X) \le \mu(YZX) \le \mu(ZX)$. Hence $\mu(X) \le \mu(Z)$ by Strong Interpolation. Thus $\mu(ZX) \le \mu(Z)$, which implies that $\mu(Y) \le \mu(Z)$. Hence by the Balancing Lemma, $\mu(XY) \le \mu(Z)$, i.e. $c_i \le c_k$. It follows that $\le$ is transitive, and thus $\le$ is a linear preorder. ▪

A stronger Cyclic Conjugate Theorem will be obtained for sequences in which $\le$ is a linear ordering on the set of cuts. By the above Lemma, these are sequences for which $\le$ is antisymmetric (i.e. $c_i \le c_j$ and $c_j \le c_i$ implies that $i = j$). This class of sequences can be easily characterized.

Definition. A nonempty sequence S is *unbalanced* if $\mu(U) \ne \mu(S)$ for any proper prefix U of S. S is *cyclically unbalanced* if every cyclic conjugate of S is unbalanced. ▪

Lemma 3.3.2. For any nonempty sequence S, $\le$ is a linear ordering on C_S if and only if S is cyclically unbalanced.

Proof. This follows easily from the Cut Order Theorem, using the Strong Interpolation Property. ▪

To state the stronger version of the Cyclic Conjugate Theorem that holds for cyclically unbalanced sequences, we introduce the following notation.

Definition. Given a nonempty sequence $S = s_1 \cdots s_n$,
$\psi(S)$ = the number of indices i, $1 \le i < n$, such that $\mu(s_1 \cdots s_i) \ge \mu(S)$ and
$\psi^*(S)$ = the number of indices i, $1 \le i < n$, such that $\mu(s_1 \cdots s_i) > \mu(S)$. ▪

This notation actually provides a slightly more general framework for the theory of quasi-monotonic sequences.

Lemma 3.3.3. For any nonempty sequence S,
S is quasi-increasing $\Leftrightarrow \psi(S) = 0$,
S is quasi-nondecreasing $\Leftrightarrow \psi^*(S) = 0$,
S is quasi-decreasing $\Leftrightarrow \psi^*(S) = n-1$ and
S is quasi-nonincreasing $\Leftrightarrow \psi(S) = n-1$.

Proof. This is obvious. ▪

Theorem 3.3.4. (Strong Cyclic Conjugate Theorem) If $S = s_1 \cdots s_n$ is cyclically unbalanced then

(1) $\psi(T) = \psi^*(T)$ for every cyclic conjugate T of S and

(2) for every value of k, $0 \le k \le n-1$, there is a unique cyclic conjugate T of S such that $\psi(T) = k$.

Proof. The first part is obvious. For the second part, since S is cyclically unbalanced, $\leq$ is a linear order on C_S by Lemma 3.3.2. Let $C_{i_1}, \ldots, C_{i_n}$ be the cuts of S listed in increasing order. For any j, $1 \leq j \leq n$, $C_{i_j} \leq C_{i_m}$ for exactly $n-j$ cuts C_{i_m} distinct from C_{i_j}. Thus if T_j is the cyclic conjugate of S formed by the cut C_{i_j}, $1 \leq j \leq n$, then $\psi(T_j) = \psi^*(T_j) = (n-1) - (n-j) = j-1$, $1 \leq j \leq n$. The result follows. ∎

The following corollary of this result will be useful.

Theorem 3.3.5. (Counting Theorem) If Ω is a set of unbalanced sequences of length n which is closed under cyclic conjugation, then for each k, $0 \leq k \leq n-1$, there are exactly $\frac{|\Omega|}{n}$ sequences S in Ω such that $\psi(S) = k$ (equivalently, $\psi^*(S) = k$).

Proof. Obviously, by these assumptions the sequences of Ω must be cyclically unbalanced. Further, by the above result, every sequence in Ω has n distinct cyclic conjugates, all of which are in Ω. It follows that Ω can be partitioned into $m = \frac{|\Omega|}{n}$ classes of n sequences each, where each class is the set of all cyclic conjugates of a given sequence S. Additionally from the above result, for each possible value of ψ there is one element of each class on which ψ has this value. Thus for any particular value of ψ, ψ has this value on exactly m sequences of Ω. ∎

When μ is the lexicographic mean, it is clear that $\mu(X) = \mu(Y)$ if and only if $XY = YX$, i.e. if and only if there exists a nonempty sequence Z and $i, j > 0$ such that $X = Z^i$ and $Y = Z^j$ (see e.g. [*Lot 83*]). It follows that a sequence S is unbalanced with respect to the lexicographic mean if and only if it is primitive, i.e. if and only if there exists no nonempty sequence Z and $i > 1$ such that $S = Z^i$. In this case S will be cyclically unbalanced as well, since every cyclic conjugate of a primitive sequence is primitive. Thus when μ is the lexicographic mean, Theorem 3.3.4 holds for every primitive sequence and Theorem 3.3.5 holds for every set of primitive sequences closed under cyclic conjugation. In fact this is true of any μ which has the property that $\mu(X) = \mu(Y)$ if and only if $XY = YX$.

3.4 Counting and Randomly Generating Ordered Trees

The Counting Theorem can be applied to many types of enumeration problems, in particular, to many of those involving objects enumerated by the well-known Catalan numbers, $C_n = \frac{1}{n+1}\binom{2n}{n}$ (see e.g. [*Gar 76*], [*Sing 79*], [*Der 80*]).

As an example, consider the following list of objects given in [*Der 80*].

Definition.

T_n is the set of rooted ordered trees with n edges (i.e. $n+1$ nodes).

P_n is the set of *legal* sequences of n open and n closed parentheses. A parenthetic expression is called "legal" if each open parenthesis has a matching close parenthesis.

I_n is the set *dominating* sequences $S = s_1 \cdots s_{n+1}$ of $n+1$ nonnegative integers which sum to n, such that $\sum_{j=1}^{i} s_j \geq i$ for all i, $1 \leq i \leq n$. (Because of a misprint in [*Der 80*], we follow the definition given in [*Read 72*] here (see also [*Zak 79*])).

L_n is the set of *admissible* paths from the point (0,0) to (n,n) in an $n \times n$ lattice. All steps in a lattice path are either up or to the right; a path is "admissible" if it does not pass below the the diagonal $y = x$.

B_n is the set of *full* binary trees with n internal nodes. A rooted ordered tree is "full binary" if all nodes are either of degree 0 (leaves) or of degree 2 (have exactly two successors). ▪

Using one-to-one correspondences between these objects, by showing that the dominating sequences are enumerated by the Catalan numbers, Dershowitz and Zaks show that all of the above objects are enumerated by the Catalan numbers. We demonstrate briefly how the Counting Theorem can be applied to achieve this result. We will use the following general property of the arithmetic mean for sequences of integers.

Lemma 3.4.1. Let μ be the arithmetic mean and S be a sequence of integers of length n which sums to $t \neq 0$. If n and t are relatively prime then S is unbalanced.

Proof. Let U be any proper prefix of S and let t' be the sum of U. If $\mu(U) = \mu(S)$ then $\frac{t'}{|U|} = \frac{t}{n}$, which is impossible because n and t are relatively prime and $|U| < n$. Thus S is unbalanced. ▪

Lemma 3.4.2. Let μ be the arithmetic mean and let $S = s_1 \cdots s_{n+1}$ be a sequence of $n+1$ nonnegative integers which sums to n. Then S is a dominating sequence if and only if S is quasi-decreasing.

Proof. If S is a dominating sequence then $\frac{\sum_{j=1}^{i} s_j}{i} \geq 1 > \frac{n}{n+1} = \mu(S)$ for every i, $1 \leq i \leq n$. Hence S is quasi-decreasing. On the other hand, if S is quasi-

decreasing, then $\frac{\sum_{j=1}^{i} s_j}{i} > \frac{n}{n+1} > \frac{i-1}{i}$ for every i, $1 \le i \le n$. Hence $\sum_{j=1}^{i} s_j \ge i$ for any i, $1 \le i \le n$. Thus S is a dominating sequence. ∎

Theorem 3.4.3. $I_n = C_n$ for all $n \ge 1$.

Proof. Let Ω be the set of all sequences of $n+1$ nonnegative integers which sum to n and let μ be the arithmetic mean. By the Lemma 3.4.1, Ω is a set of unbalanced sequences of length $n+1$ which is closed under cyclic conjugation. Hence by the Counting Lemma, exactly $\frac{1}{n+1}|\Omega|$ sequences from this set are quasi-decreasing. Thus we need only show that $|\Omega| = \binom{2n}{n}$ and the result will follow from Lemma 3.4.2. This latter fact is easily established by showing that every sequence $S = s_1 \cdots s_{n+1}$ in Ω can be uniquely represented by a sequence S' of n 0's and n 1's where $S' = 1^{s_1}01^{s_2}0 \cdots 01^{s_{n+1}}$, and vice-versa that every such sequence represents a member of Ω. ∎

Since by the correspondence of [*Der 80*], the dominating sequence associated with a given tree is simply the sequence of outdegree of its nodes visited in preorder, we can also use these techniques to count trees whose nodes have any specific spectrum of outdegrees. If t is a tree with $n+1$ nodes and n_i is the number of nodes with outdegree i for $1 \le i \le k$, where k is the maximal outdegree of any node, then we must have

(1) $n+1 = n_0+n_1+\cdots+n_k$

{ The total number of nodes is $n+1$. }

(2) $n = n_1+2n_2+\cdots+kn_k$

{ The total number of edges is n. }

Theorem 3.4.4. The number of rooted oriented trees with $n+1$ nodes and n_i nodes of outdegree i for $0 \le i \le k$, where the n_i satisfy (1) and (2) above, is $\frac{1}{n+1}\left(\frac{(n+1)!}{n_0!n_1!\cdots n_k!}\right)$.

Proof. Consider the set Ω of all sequences with n_0 0's, n_1 1's, ..., n_k k's. By (1) these sequences are of length $n+1$ and by (2) they sum to n. Thus as in the previous theorem, exactly $\frac{1}{n+1}|\Omega|$ of these sequences are dominating sequences, i.e. represent legitimate trees. The result follows. ∎

One application of these results is in the generation of random trees. Using the technique from the proof of Theorem 3.4.3, we can obtain a random tree with $n+1$ nodes by generating a random binary sequence with n 0's and n 1's, viewing it as a sequence of $n+1$ numbers in unary separated by 0's, obtaining the

quasi-increasing cyclic conjugate of this sequence of numbers (under the arithmetic mean) by the Rotate-and-Merge algorithm, and finally interpreting the resulting sequence as the preorder traversal of a tree. To generate trees whose nodes have a specific spectrum of outdegrees, the initial sequence of $n+1$ numbers can be chosen to reflect these constraints. This general method is well-suited for efficient implementation, and thus should prove practical in situations where rapid generation of random trees is needed.

The techniques for counting trees and other objects given above are not unrelated to the specific techniques given in [*Der 80*] and other techniques for Catalan enumerations, for example those in [*Sil 69*], [*San 78*] and [*Sing 78*]. Since the literature on Catalan numbers and their relatives is extensive (a bibliography of 470 references is given in [*Gou 76*]), no attempt has been made to cover the applications of the present theory in this area in any detail. Hence this remains an interesting area for further research.

3.5 Ballot Problems

Our final application of the Counting Theorem involves a version of the classic ballot problem [*Ber 87*].

A typical ballot problem may be described as follows. Suppose that an election is held, and candidate A receives a votes, while candidate B receives b votes. Let S be the sequence of votes as they are received, and suppose that all $\binom{a+b}{a}$ possible arrangements of S are equally likely. For a given γ, let $\Delta_\gamma(S)$ denote the number of times during the election that the ratio of votes for A to the total votes is greater than or equal to γ. For any given k, $1 \le k \le a+b$, what is the probability that $\Delta_\gamma(S) = k$?

To illustrate the application of the Counting Theorem to this type of problem, we will derive the following result from [*Sri 79*], originally due to Takacs ([*Tak 62*]). Our version is a minor rewording of that given in [*Sri 79*].

Theorem 3.5.1. If a and b are relatively prime and $\gamma = \frac{a}{a+b}$ (i.e. when γ is the final ratio of votes for A), then the probability that $\Delta_\gamma(S) = k$ is $\frac{1}{a+b}$ for all k, $1 \le k \le a+b$.

Proof. Let S be given as a sequence of integers where each vote for A is represented by 1 and each vote for B is represented by 0. Let μ be the arithmetic mean. Thus $\mu(S) = \frac{a}{a+b} = \gamma$. Let U be a proper prefix of S of length r and let α be the number of 1's in U. The ratio of votes for A in U is greater than or

equal to γ if and only $\mu(U) = \frac{a}{r} \geq \gamma = \mu(S)$. Thus $\Delta_\gamma(S) = \psi(S) + 1$, since the ratio of votes for A is always greater than or equal to γ at the end of the election. Furthermore, since a and b are relatively prime, a and $a+b$ are relatively prime, and thus by Lemma 3.4.1, S is unbalanced. Since the set of all possible voting records is obviously closed under cyclic conjugation, by the Counting Theorem, all values of Δ_γ between 1 and $a+b$ are equally likely on this set, and the result follows. ▪

We can also obtain another related, but more general theorem of Takacs' ([*Tak 67*], Theorem 1, page 162) using this method.

Theorem 3.5.2. (Takacs) Let $S = s_1 \cdots s_n$ be a sequence of integers which sums to 1. For each j, $1 \leq j \leq n$, there is exactly one cyclic conjugate of S for which exactly j of its partial sums are positive.

Proof. Let μ be the arithmetic mean. Then by Lemma 3.4.1, S is cyclically unbalanced. Let $U = s_1 \cdots s_i$ be any proper prefix of S. If the partial sum $t = \sum_{k=1}^{i} s_k$ is positive, then $\mu(U) = \frac{t}{|U|}$ must be greater than $\mu(S) = \frac{1}{n}$. On the other hand, if $\mu(U) > \frac{1}{n}$, then clearly t must be positive. It follows that exactly j partial sums of S are positive if and only if $\psi^*(S) = j$. Thus the result follows from the Strong Cyclic Conjugate Theorem. ▪

Theorem 2.1 of [*Spi 56*] can be derived from the Strong Cyclic Conjugate Theorem in a similar manner.

FURTHER RESEARCH

We have already alluded to a few possible directions for further research, among them being a detailed axiomatic investigation of averaging systems and the theory of quasi-monotonic sequences, and a more extensive investigation of the time bounds for the major algorithms used in this theory, using more general assumptions concerning μ. We hope to present results of the first type in a future paper. Under the latter topic, we note that the question of parallel algorithms for finding decompositions and quasi-nondecreasing cyclic conjugates of sequences remains to be explored as well. This is an area in which we have done almost no work at this time. If good parallel algorithms are found, further applications in the area of loop or ring-structured networks (see e.g. [*Dol 82*]) might be explored.

We also hope to use this theory to investigate certain aspects of the structure of random sequences. Using a technique similar to the one used in the proof of Theorem 2.1 in [*Spi 56*], in certain cases we can find correspondences between the trends in the decomposition of a sequence and the cycles in a permutation of that sequence. This allows us to show, for example, that the expected number of trends (using the arithmetic mean) in a sequence of n reals randomly chosen in the interval between 0 and 1 is the same as the expected number of cycles in an arbitrary permutation of that sequence, which is $ln(n)$. We hope to present this and other results in a future paper as well.

ACKNOWLEDGEMENTS

We would like to thank Ron Graham for telling us of the "Trip Around the Moon" problem, and pointing out the applications of this theory to this problem and to the results of Takacs. We are also indebted to Joel Cohen for showing one of us a simple convex hull finding algorithm which another of us "rediscovered" in the setting this theory.

References

[Aho 74] Aho, A., Hopcroft, J., Ullman, J., *The design and analysis of computer algorithms*, Addison Wesley, Reading Mass., 1974.

[And 79] Andrew, A. M., "Another efficient algorithm for convex hulls in two dimensions," *Inf. Proc. Let.*, 9(5) (1979), 216-219.

[Ben 78] Bentley and Shamos , "Divide and conquer for linear expected time," *Inf. Proc. Let.*, 7 (1978), 87-91.

[Ber 87] Bertrand, J., "Solution d'un probleme," *Comples Rendus des Seances de l'Academie des Sciences*, Paris, 105 (1887), 369.

[Der 80] Dershowitz, N. and Zaks, S., "Enumerations of ordered trees," *Discrete Mathematics*, 31 (1980), 9-28.

[Dol 82] Dolev, D., Klawe, M., Rodeh, M., "An O(nlogn) unidirectional distributed algorithm for extrema finding in a circle," *J. Algorithms*, 3 (1982), 245-260.

[Duv 83] Duval, J. P., "Factorizing words over an ordered alphabet," *J. Algorithms*, 4 (1983), 363-381.

[Dvo 80] Dvornicich, R., "On a problem of cyclic permutations of integers," *Discrete Applied Mathematics*, 2 (1980), 353-355.

[Gar 76] Gardner, M., "Catalan numbers: An integer sequence that materializes in unexpected places," *Scientific American*, 243 (June 1976), 120-125.

[Gou 76] Gould, H. W., "Research bibliography of two special number sequences," *rev. ed. (1976), (Available from author at 1239 College A., Morgantown*, W. V. 26505, U.S.A.

[Gra 63] Graham, R. L., "A combinatorial theorem for partial sums," *Ann. Mat. Statist.*, 34 (1963), 1600-1602.

[Gra 83] Graham, R. L., personal communication.

[Lot 83] Lothaire, M., "Combinatorics on words," *Encyclopedia of Mathematics and its Applications*, Vol. 17, Addison-Wesley, 1983.

[Mei 80] Meijer, H. and Akl, S., "Design and analysis of a new hybrid sorting algorithm," *Inf. Proc. Let.*, 10(4,5) (1980), 213-218.

[Pre 79] Preparata, F. P., "An optimal real time algorithm for planar convex hulls," *CACM*, 22(7) (1979), July, 402-404.

[Read 72] Read, R. C., "The coding of various kinds of unlabeled trees," *Graph Theory and Computing*, R. C. Read, ed., Academic Press (1972), 153-182.

[San 78] Sands, A. D., "On generalized Catalan numbers," *Discrete Mathematics*, 21 (1978), 219-221.

[Sil 69] Silberger, D.M., "Occurrences of the integer (2n-2)!/n!(n-1)!," *Math. Proca. Mat.* 13 (1969), 91-96.

[Sin 78] Singmaster, D., "An elementary evaluation of the Catalan numbers," *Amer. Math. Monthly*, 85 (1978), May, 366-368.

[Sin 79] Singmaster, D., "Some Catalan correspondences," *J. London Math. Soc. (2)*, 19 (1979), 203-206.

[Spi 56] Spitzer, F., "A combinatorial lemma and its applications to probability theory," *Trans. Amer. Math. Soc.*, (82) (1956), 323-339.

[Sri 79] Srinivasan, R., "On some results of takacs in ballot problems," *Discrete Mathematics*, 28(2) (1979), 213-218.

[Tak 67] Takacs, L., "Combinatorial methods in the theory of stochastic processes," John Wiley, N.Y., (1967).

[Tak 62] Takacs, L., "Ballot problems," *Z. Wahrscheinlichkeitstheorie*, 1 (1962), 154-158.

[Vie 78] Viennot, G., "Algebres de Lie Libres et Monoides Libres," *Lect. Notes in Math.*, Vol. 691, Springer-Verlag, 1978.

[Yao 79] Yao, A., "A lower bound to finding convex hulls," Tech. Rep. STAN-CS-79-733, Stanford Univ., 1979.

[Zak 79] Zaks, S. and Richards, D., "Generating trees and other combinational objects lexicographically," *SIAM J. Comput.*, 8(1) (1979), 73-81.