THE CONSTRUCTION OF HUFFMAN CODES IS A SUBMODULAR (‘CONVEX’) OPTIMIZATION PROBLEM OVER A LATTICE OF BINARY TREES

D. STOTT PARKER* AND PRASAD RAM†

Abstract. We show that the space of all binary Huffman codes for a finite alphabet defines a *lattice*, ordered by the imbalance of the code trees. Representing code trees as path-length sequences, we show that the imbalance ordering is closely related to a majorization ordering on real-valued sequences that correspond to discrete probability density functions. Furthermore, this tree imbalance is a partial ordering that is consistent with the total orderings given by either the external path length (sum of tree path lengths), or the entropy determined by the tree structure. On the imbalance lattice, we show the weighted path-length of a tree (the usual objective function for Huffman coding) is a *submodular* function, as is the corresponding function on the majorization lattice. Submodular functions are discrete analogues of convex functions. These results give perspective on Huffman coding and suggest new approaches to coding as optimization over a lattice.

Key words. Huffman coding, adaptive coding, prefix codes, enumeration of trees, lattices, combinatorial optimization, convexity, submodular functions, entropy, tree imbalance, Schur convex functions, majorization, Moebius inversion, combinatorial inequalities, Fortuin-Kasteleyn-Ginibre (FKG) inequality, quadrangle inequality, Monge matrices, dynamic programming, greedy algorithms.

AMS subject classifications. 94A15, 94A24, 94A29, 94A45, 90C25, 90C27, 90C39, 90C48, 52A41, 68Q20, 68R05, 05A05, 05A20, 05C05, 05C30, 06A07, 26B25, 26D15

1. Introduction.

The Huffman algorithm has been used heavily to produce efficient binary codes for almost half a century now. It has inspired a large literature with diverse theoretical and practical contributions. A comprehensive, very recent survey is [1]. Although the algorithm is quite elegant, it is tricky to prove correct and to reason about. While there may be little hope of improving on the $O(n \log n)$ complexity of the Huffman algorithm itself,[1] there is still room for improvement in our understanding of the algorithm.

There is also plenty of room for improvement in our understanding of variants of Huffman coding. Although the Huffman algorithm is remarkably robust in general and has widespread use, it is far from optimal in many real applications. Huffman coding is optimal only when the symbols to be coded are random and occur with fixed probabilities. Time-varying dependencies are not captured by the Huffman coding model, and optimal encoding of finite messages is not captured either.

Our motivation came from analysis of dynamic Huffman coding, a specific extension of Huffman coding in which the code used evolves over time. Recently dynamic coding algorithms have been studied heavily. Our initial idea was to define “rebalancing” operations on code trees and to use these dynamically (“on the fly”) in producing better codes, in situations where the distribution of symbols to be coded varies over time and/or is not accurately predictable in advance.

This paper reconstructs Huffman coding as an optimization over the space of binary trees. A natural representation for this space is sequences of ascending path-lengths, since this captures what is significant in producing optimal codes.

We show that the set of path-length sequences representing binary trees forms a lattice, which we call the **imbalance lattice**. This lattice orders trees by their imbalance and gives an organization for them that is useful in optimization. Our belief is that having a better

*UCLA Computer Science Department, University of California, Los Angeles, CA 90095-1596 (stott@cs.ucla.edu).

†Xerox Corporation, El Segundo, CA 90245 (prasad@cp10.es.xerox.com).

[1] The algorithm is closely related to sorting, in the sense that the sorted sequence of a sequence of integer values $\langle x_1 \cdots x_n \rangle$ is obtainable directly from the optimal code tree for the values $\langle 2^{x_1} \cdots 2^{x_n} \rangle$ (e.g., [26, p.335]).

mathematical (and not purely procedural) understanding of coding will ultimately pay off in improved algorithms.

The imbalance lattice and its imbalance ordering on trees depend on **majorization** in an essential way. Majorization is an important ordering on sequences that has many applications in pure and applied mathematics [27]. We have related it to greedy algorithms directly [33]. Earlier majorization was recognized as an important property of the internal node weights produced by the Huffman algorithm [13, 32], and in this work we go further to clarify its pervasive role.

By viewing the space of trees as a lattice, a variety of new theorems and algorithms become possible. For example, the objective functions commonly used in evaluating codes are **submodular** on this lattice. Submodular functions are closely related to convex functions (as we explain later; see Theorem 4.5) and are often easy to optimize [6, 9, 23, 24, 25]. Huffman coding gives a significant example of the importance of submodularity in algorithms.

2. Ordered Sequences, Rooted Binary Trees, and Huffman Codes.

2.1. Ordered Sequences. By a **sequence** we mean an ordered collection of nonnegative real values

$$\mathbf{x} \quad = \quad \langle\ x_1\ x_2\ \cdots\ x_n\ \rangle.$$

Repetition of values in the sequence is permitted: the values x_j need not be distinct. The **length** of this sequence is n, and for simplicity we also refer to the set of such sequences with the vector notation $\Re_+{}^n$.

We introduce several useful operators on sequences:

ascending sort	$\mathrm{sort}{\uparrow}(\,\mathbf{x}\,)$	$=$	$\langle$ $\mathbf{x}$ put in ascending order $\rangle$
descending sort	$\mathrm{sort}{\downarrow}(\,\mathbf{x}\,)$	$=$	$\langle$ $\mathbf{x}$ put in descending order $\rangle$
sequence exponential	$2^{-\mathbf{x}}$	$=$	$\langle\ 2^{-x_1}\ \cdots\ 2^{-x_n}\ \rangle$
sequence logarithm	$-\log_2(\mathbf{x})$	$=$	$\langle\ -\log_2(x_1)\ \cdots\ -\log_2(x_n)\ \rangle.$

A **density sequence** is a nonnegative real-valued sequence whose entries sum to 1.

A **distribution sequence** is an ascending nonnegative sequence whose final entry is 1.

For simplicity, throughout this paper **many sequences are implicitly sorted**:

$\boldsymbol{\ell}$, $\mathbf{s}$, $\mathbf{t}$, $\mathbf{u}$ denote **ascending** sequences of positive **integer values** whose sequence exponentials $2^{-\boldsymbol{\ell}}$, $2^{-\mathbf{s}}$, $2^{-\mathbf{t}}$ are density sequences.

$\mathbf{w}$ denotes a **descending** sequence of positive **real values**.

$\mathbf{v}$ denotes an **ascending** distribution sequence.

$\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}$ denote **descending** density sequences.

Note since $\boldsymbol{\ell}$ is ascending, $2^{-\boldsymbol{\ell}}$ is descending; and since $\mathbf{x}$ is descending, $-\log_2(\mathbf{x})$ is ascending.

We also allow **sequences** to be operated upon as **vectors**. Thus, if $\mathbf{x}$ is a sequence (vector) of length n and A is an $n \times n$ matrix, then $A\,\mathbf{x}$ is a sequence (vector). Treating sequences as vectors allows us to define several useful operators using matrix algebra.

2.2. Rooted Binary Trees and Path-Length Sequences. Rooted binary trees here are binary trees with a root node, in which every node is either a leaf node, or an internal node having one parent and two children. The order of the leaves is insignificant, so a given tree is determined (up to permutation of the leaves) by the lengths of the paths from the root node to each leaf node (the distance of the leaf from the root). Thus we can represent equivalence classes of the rooted binary trees with n leaves by sequences of n nonnegative integers, which give the path-length of each leaf. For example, the path-length sequence

$$\langle\ 1\ 3\ 3\ 4\ 4\ 4\ 4\ \rangle$$

represents a binary tree with $n = 7$ leaves, of which one has path-length 1, two have path-length 3, and four have path-length 4; it is shown in Figure 2.1.

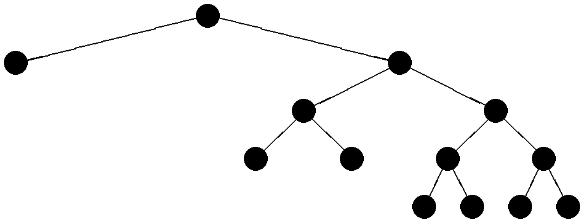


FIG. 2.1. *A binary tree having path-length sequence* $\langle\, 1\,3\,3\,4\,4\,4\,4 \,\rangle$

Path-length sequences obey what we call the **Kraft equality**, a special case of the Kraft inequality of noiseless coding theory (see e.g., [10], p.45).

THEOREM 2.1. *For all* $n \geq 1$, $\langle\, \ell_1 \,\cdots\, \ell_n \,\rangle$ *is the sequence of path-lengths in a rooted binary tree* **iff**

$$\sum_{i=1}^{n} 2^{-\ell_i} \quad = \quad 1.$$

Thus $\boldsymbol{\ell}$ *is a path-length sequence* **iff** $2^{-\boldsymbol{\ell}}$ *is a density sequence.*

Proof. Easily proven by induction on n. For the basis, with $n = 1$ we must have $\ell_1 = 0$. The induction step follows by noticing that the two principal subtrees of any binary tree must have sequences $\langle \ell'_1 \,\cdots\, \ell'_p \rangle$ and $\langle \ell''_1 \,\cdots\, \ell''_q \rangle$ satisfying the equality and that their composition has the sequence $\langle\, (\ell'_1+1) \,\cdots\, (\ell'_p+1)\; (\ell''_1+1) \,\cdots\, (\ell''_q+1) \,\rangle$, which again satisfies the equality. □

n	1	2	3	4	5	6	7	8
T_n	1	1	1	2	3	5	9	16
	$\langle\, 0 \,\rangle$	$\langle\, 1\,1 \,\rangle$	$\langle\, 1\,2\,2 \,\rangle$	$\langle\, 1\,2\,3\,3 \,\rangle$ $\langle\, 2\,2\,2\,2 \,\rangle$	$\langle\, 1\,2\,3\,4\,4 \,\rangle$ $\langle\, 1\,3\,3\,3\,3 \,\rangle$ $\langle\, 2\,2\,2\,3\,3 \,\rangle$	$\langle\, 1\,2\,3\,4\,5\,5 \,\rangle$ $\langle\, 1\,2\,4\,4\,4\,4 \,\rangle$ $\langle\, 1\,3\,3\,3\,4\,4 \,\rangle$ $\langle\, 2\,2\,2\,3\,4\,4 \,\rangle$ $\langle\, 2\,2\,3\,3\,3\,3 \,\rangle$	$\langle\, 1\,2\,3\,4\,5\,6\,6 \,\rangle$ $\langle\, 1\,2\,3\,5\,5\,5\,5 \,\rangle$ $\langle\, 1\,2\,4\,4\,4\,5\,5 \,\rangle$ $\langle\, 1\,3\,3\,3\,4\,5\,5 \,\rangle$ $\langle\, 1\,3\,3\,4\,4\,4\,4 \,\rangle$ $\langle\, 2\,2\,2\,3\,4\,5\,5 \,\rangle$ $\langle\, 2\,2\,2\,4\,4\,4\,4 \,\rangle$ $\langle\, 2\,2\,3\,3\,3\,4\,4 \,\rangle$ $\langle\, 2\,3\,3\,3\,3\,3\,3 \,\rangle$	$\vdots$

TABLE 2.1
All path-length sequences of trees with n *leaves, for small values of* n

Henceforth we assume that **tree path-length sequences are in ascending sorted order**. Table 2.1 shows a lexicographic tabulation of all possible sequences for $1 \leq n \leq 7$, along with T_n, the total number of inequivalent sequences of length n. T_n is enumerated as sequence M0710 (A002572) in [39]. An upper bound on T_n can be obtained from the Catalan number C_n, which computes the number of unordered binary trees: for $n \geq 3$, $T_n \leq \frac{1}{2} C_n \leq 2^{n-3}$. Gilbert [12], using the notation $g(N)$ for T_N, points out that T_n is well approximated for $n \leq 30$ by

$$T_n \quad \simeq \quad 0.148\;(1.791)^n.$$

2.3. Huffman Codes are Optimal Path-Length Sequences. A Huffman code for a given positive weight sequence

$$w_1 \;\geq\; w_2 \;\geq\; \cdots \;\geq\; w_n$$

consists of a binary tree, i.e., a path-length sequence $\boldsymbol{\ell} = \langle\, \ell_1\, \ell_2 \,\cdots\, \ell_n \,\rangle$, which we evidently want to be in ascending order,

$$\ell_1 \;\leq\; \ell_2 \;\leq\; \cdots \;\leq\; \ell_n,$$

so that the weighted path-length

$$g_{\mathbf{w}}(\boldsymbol{\ell}) \quad = \quad \sum_{i=1}^{n} w_i\,\ell_i$$

is minimal. Beyond the Kraft equality of Theorem 2.1, it is difficult to characterize what it is that makes $\boldsymbol{\ell}$ optimal. For example, Table 2.2 shows all feasible codes and costs for the weight sequence $\mathbf{w} = \langle\ 189\ \ 95\ \ 73\ \ 71\ \ 28\ \ 23\ \ 21\ \rangle$, with $n = 7$.

$\boldsymbol{\ell}$	$g_{\mathbf{w}}(\boldsymbol{\ell})$
$\langle$ 1 2 3 4 5 6 6 $\rangle$	*1286*
$\langle$ 1 2 3 5 5 5 5 $\rangle$	*1313*
$\langle$ 1 2 4 4 4 5 5 $\rangle$	*1287*
$\langle$ 1 3 3 3 4 5 5 $\rangle$	$\boxed{\textit{1238}}$
$\langle$ 1 3 3 4 4 4 4 $\rangle$	*1265*
$\langle$ 2 2 2 3 4 5 5 $\rangle$	*1259*
$\langle$ 2 2 2 4 4 4 4 $\rangle$	*1286*
$\langle$ 2 2 3 3 3 4 4 $\rangle$	*1260*
$\langle$ 2 3 3 3 3 3 3 $\rangle$	*1311*

TABLE 2.2
Path-length sequences $\boldsymbol{\ell}$ and their weighted path-length $g_{\mathbf{w}}(\boldsymbol{\ell})$ for $\mathbf{w} = \langle$ 189 95 73 71 28 23 21 $\rangle$

Huffman's breakthrough [18] was to identify an efficient algorithm that finds an optimal tree, avoiding a search over the exponentially large space of trees. The algorithm repeatedly combines the two tree leaves with least weight, whose sum becomes the weight of a new leaf. The Huffman (optimal) tree in Table 2.2 has path lengths $\boldsymbol{\ell} \ = \ \langle\ 1\ 3\ 3\ 3\ 4\ 5\ 5\ \rangle$ and total weighted path-length 1238. The Huffman algorithm reflects a divide-and-conquer structure that has interesting properties on the space of trees, but because of its procedural nature does little to characterize optimal trees.

3. The Imbalance Lattice of Binary Trees.

The optimality of a Huffman code is determined by the match between the balance (or imbalance) between the code tree and the weights of the symbols to be coded. In this section we show ternary balancing exchanges give an imbalance ordering on binary trees that defines a lattice.

The idea of using using lattices in coding dates back at least to Shannon in 1950 [38]. However, we have not found the lattice characterization of tree imbalance elsewhere. Following considerable work in the early 1980s on enumeration of trees, Pallo classified trees by their *rotational* structure (e.g., [30, 31]) and showed that they then form a lattice. Our work differs from Pallo's in that we classify trees by their *path-length* (imbalance) structure.

3.1. Important Properties of Tree Path-Length Sequences.

THEOREM 3.1. *Every path-length sequence $\boldsymbol{\ell}$ has the form*

$$\boldsymbol{\ell} \quad = \quad \langle \quad \cdots \quad (q-j) \quad \overbrace{q \quad \cdots \quad q}^{2k} \quad \rangle,$$

a sequence including $2k$ copies of its largest value q (where $j,\ k > 0$). Also, j is at most the largest exponent of 2 in $2k$, and therefore $j \leq \log_2(2k)$.

Proof. $\boldsymbol{\ell}$ must include $2k$ copies of its largest value q since otherwise $\left(2^q \cdot \sum_{i=1}^{n} 2^{-\ell_i}\right)$ is odd, contradicting the Kraft equality. Using this argument again on the shorter path-length sequence obtained by replacing the $2k$ copies of q with k copies of $(q-1)$, the Kraft equality requires not only that $j > 0$ but also that j be at most the number of times 2 divides $2k$. $\square$

THEOREM 3.2. *Except for the sequence $\langle\ 1\ 2\ 3\ \ldots\ (n-2)\ (n-1)\ (n-1)\ \rangle$, any path-length sequence contains at least three identical values.*

Proof. By induction on the length n of the sequence. For the basis, when $n = 3$ the only sequence is $\langle\ 1\ 2\ 2\ \rangle$, satisfying the theorem. For the induction step, suppose $n > 3$, and to the contrary of the theorem that there is a sequence does not have three identical values. Let q be the smallest value in the sequence appearing twice. We may assume $q < (n-1)$, since otherwise the sequence is $\langle\ 1\ 2\ 3\ \ldots\ (n-2)\ (n-1)\ (n-1)\ \rangle$. Construct the sequence of length $n-1$ that results from replacing the two values q with one value $(q-1)$. In this new sequence, q does not appear at all (since there were only two before), and $(q-1)$ appears at most twice. Therefore, by induction, since this sequence does not have three identical values it is $\langle\ 1\ 2\ 3\ \ldots\ (n-3)\ (n-2)\ (n-2)\ \rangle$. But since $q < (n-1)$ and q does not appear in the new sequence, this gives a contradiction. □

3.2. Ternary Exchanges determine Tree Imbalance.

The insight that inspired us to write this paper is that **it is possible to generate all binary tree path-length sequences using ternary exchanges**. Given any path-length sequence

$$\langle\cdots \qquad p \qquad \cdots\cdots\cdots \quad (q+1) \quad (q+1) \quad \cdots\rangle,$$

then the revision

$$\langle\cdots \quad (p+1) \quad (p+1) \quad \cdots\cdots\cdots \qquad q \qquad \cdots\rangle$$

is a path-length sequence also, because

$$2^{-p} + 2^{-(q+1)} + 2^{-(q+1)} \quad = \quad 2^{-p} + 2^{-q} \quad = \quad 2^{-(p+1)} + 2^{-(p+1)} + 2^{-q}.$$

Moreover, if the initial sequence is sorted in ascending order, so $p \leq q$, and we replace the rightmost p and leftmost two $(q+1)$s, then the resulting sequence is still sorted. (When $p = q$ the two sequences are identical.) Dually, this exchange can be applied in reverse; with sorted sequences if we replace the leftmost two $(p+1)$s and the rightmost $(q-1)$, the result will still be sorted in ascending order.

The net effect of this exchange is to transfer two leaves dangling from level q to level p. The two examples in Figure 3.1 show this pictorially.

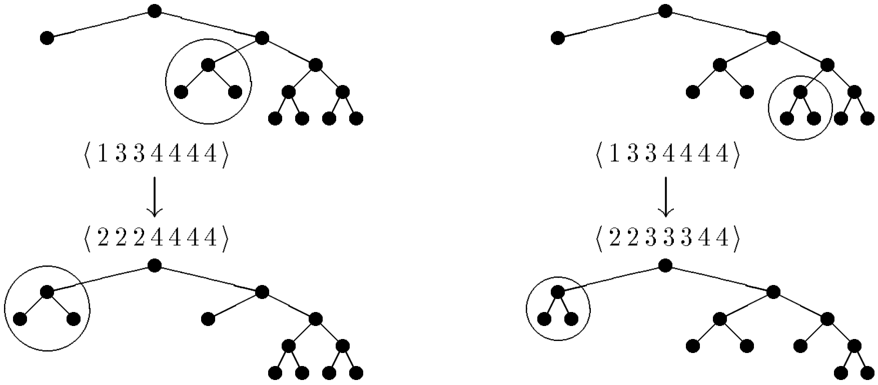


FIG. 3.1. *Balancing exchanges:* $\langle 1334444\rangle \to \langle 2224444\rangle$ *and* $\langle 1334444\rangle \to \langle 2233344\rangle$

DEFINITION 3.3. *Let p, q be integers such that $1 \leq p < q < n$.*
A **balancing exchange** *is a ternary exchange of the form*

$$\begin{array}{cccccccc} \langle\cdots & p & & \cdots\cdots & & (q+1) & (q+1) & \cdots\rangle \\ & \downarrow & \searrow & & \cdots & \searrow & \downarrow & \\ \langle\cdots & (p+1) & (p+1) & & \cdots\cdots & & q & \cdots\rangle \end{array}$$

It is called a **minimal balancing exchange** *if* $(p+1) = q$.
An **imbalancing exchange** *is of the reverse form*

$$\begin{array}{cccccccc} \langle \cdots & (p+1) & (p+1) & & \cdots\cdots & q & & \cdots\rangle \\ & \downarrow & \swarrow & \cdots & & \swarrow & \downarrow & \\ \langle \cdots & p & \cdots\cdots & & (q+1) & (q+1) & & \cdots\rangle \end{array}$$

Finally, we can define partial orders as the reflexive transitive closures of these relations among sequences. Given two sequences s *and* **t**, *we say that* s **is at least as balanced as t**,

$$\mathrm{s} \;\trianglelefteq\; \mathbf{t},$$

if there are sequences $\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_m$ $(m \geq 1)$ *where* $\mathbf{t} = \boldsymbol{\ell}_1$, $\boldsymbol{\ell}_m = \mathrm{s}$, *and for each* i, $1 \leq i < m$, *there is a balancing exchange from* $\boldsymbol{\ell}_i$ *to* $\boldsymbol{\ell}_{i+1}$.

Minimal balancing exchanges, in which $(p+1) = q$, are particularly significant. The balancing exchange $\langle\ 1\ 3\ 3\ 4\ 4\ 4\ 4\ \rangle \rightarrow \langle\ 2\ 2\ 2\ 4\ 4\ 4\ 4\ \rangle$ in Figure 3.1 gives an example. Minimal balancing exchanges are *ternary exchanges of consecutive length values*, so any tree path-length sequence of the form $\langle \cdots\ (q-1)\ \cdots\ (q+1)\ (q+1)\ \cdots\rangle$ determines the more balanced tree path-length sequence $\langle \cdots\ q\ \cdots\ q\ q\ \cdots\rangle$ and vice-versa.

THEOREM 3.4. *If two path-length sequences differ, they differ in at least three values. Also, if they differ in exactly three values, there is a ternary exchange between the sequences.*

Proof. Direct consequence of the Kraft equality. The equality shows that two path-length sequences cannot differ in one value. Similarly, there cannot be sequences s and **t** differing in two values, since if the differences were the disjoint sequences of positive integers $\langle\ s_i\ s_j\ \rangle$ and $\langle\ t_i\ t_j\ \rangle$, then the Kraft equality would imply $2^{-s_i} + 2^{-s_j} = 2^{-t_i} + 2^{-t_j}$, which is false under the disjointness condition. Finally, sequences differing in three integer values $\langle\ s_i\ s_j\ s_k\ \rangle$ and $\langle\ t_i\ t_j\ t_k\ \rangle$ must satisfy $2^{-s_i}+2^{-s_j}+2^{-s_k} = 2^{-t_i}+2^{-t_j}+2^{-t_k}$, and a case analysis shows that this is solved only by $\langle\ s_i\ s_j\ s_k\ \rangle = \langle\ p\ (q{+}1)\ (q{+}1)\ \rangle$ and $\langle\ t_i\ t_j\ t_k\ \rangle = \langle\ (p{+}1)\ (p{+}1)\ q\ \rangle$, corresponding to a ternary exchange. □

THEOREM 3.5. *The path-length imbalance ordering is a partial order.*

Proof. It is reflexive and transitive by construction. Also the imbalance ordering is antisymmetric, because s$\trianglelefteq$**t** and **t**$\trianglelefteq$s together imply s = **t**. Otherwise there would be a sequence of balancing exchanges that transform **t** to s and ultimately back to **t**; this is not possible, as each balancing exchange reduces by at least 1 the sum of the values in the sequence. □

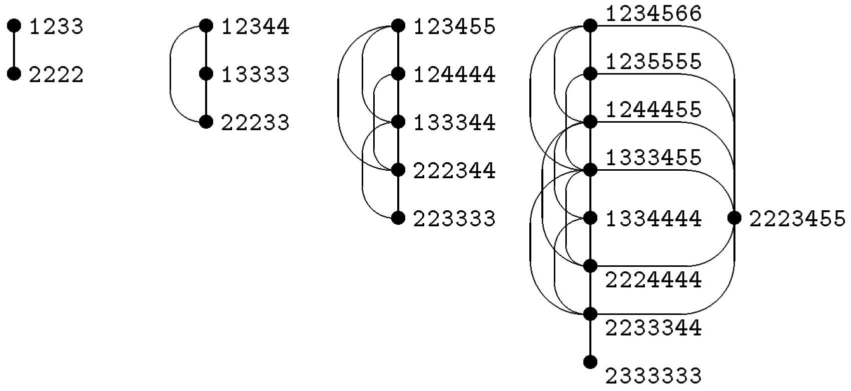
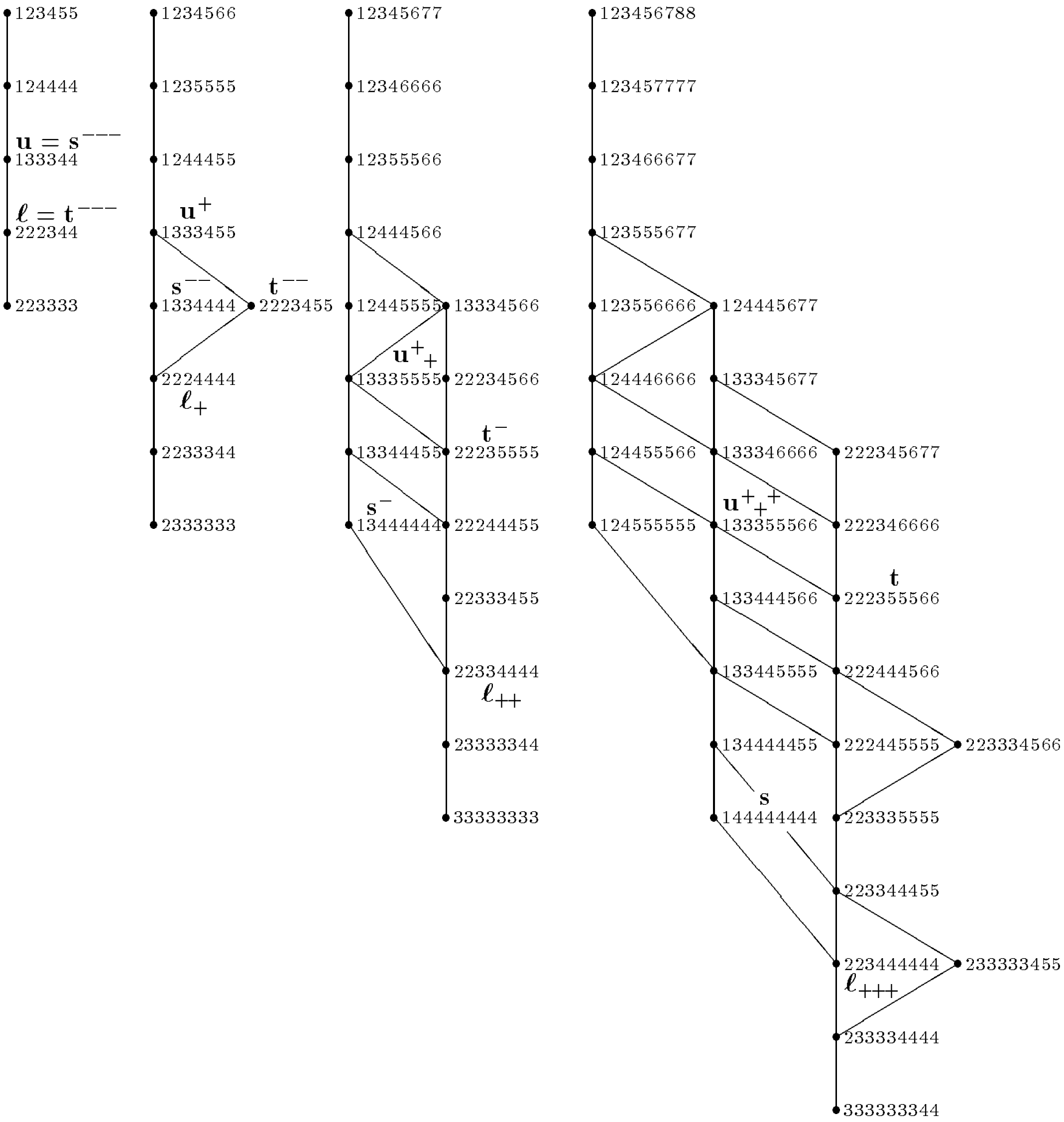


FIG. 3.2. *The path-length imbalance ordering for* $n = 4,\ 5,\ 6,\ 7$*; edges denote ternary exchanges*

The imbalance partial order is straightforward to derive for small values of n. In Figure 3.2, it is displayed for $n = 4, 5, 6, 7$. The most imbalanced sequence appears at the top of the partial order, and an edge from a sequence s down to another **t** means that a balancing exchange is possible from s to **t**. It is evident from Figure 3.2 that the minimal exchanges define the bulk of the ordering. In order to provide a deeper appreciation for its structure,



s	**t**	**s ∨ t**	**s ∧ t**
⟨ 1 4 4 4 4 4 4 4 4 ⟩	⟨ 2 2 2 3 5 5 5 6 6 ⟩	⟨ 1 3 3 3 5 5 5 6 6 ⟩	⟨ 2 2 3 4 4 4 4 4 4 ⟩
⟨ 1 4 4 4 4 4 4 4 4 ⟩	⟨ 2 2 3 3 3 4 5 6 6 ⟩	⟨ 1 3 3 4 4 4 5 6 6 ⟩	⟨ 2 2 3 4 4 4 4 4 4 ⟩
⟨ 1 3 4 4 4 4 4 5 5 ⟩	⟨ 2 2 3 3 3 4 5 6 6 ⟩	⟨ 1 3 3 4 4 4 5 6 6 ⟩	⟨ 2 2 3 3 4 4 4 5 5 ⟩
⟨ 1 2 4 5 5 5 5 5 5 ⟩	⟨ 2 2 3 3 3 4 5 6 6 ⟩	⟨ 1 2 4 4 5 5 5 6 6 ⟩	⟨ 2 2 3 3 3 5 5 5 5 ⟩
⟨ 1 2 4 5 5 5 5 5 5 ⟩	⟨ 2 2 2 3 4 5 6 7 7 ⟩	⟨ 1 2 4 4 4 5 6 7 7 ⟩	⟨ 2 2 2 4 4 5 5 5 5 ⟩
⟨ 1 2 4 4 5 5 5 6 6 ⟩	⟨ 2 2 2 3 4 5 6 7 7 ⟩	⟨ 1 2 4 4 4 5 6 7 7 ⟩	⟨ 2 2 2 3 5 5 5 6 6 ⟩

FIG. 3.3. *The imbalance lattice, showing path-length sequences ordered by imbalance. The sequence* ⟨ 1 2 3 4 ⋯ ⟩ *is maximally imbalanced. The graphs display the (transitively reduced) path-length imbalance ordering for $n = 6, 7, 8, 9$. For clarity only a minimal subset of the imbalance ordering is drawn; orderings in the transitive closure of the minimal set are omitted. The imbalance ordering is also a lattice, with well-defined upper bounds* $\mathbf{s} \vee \mathbf{t}$ *and lower bounds* $\mathbf{s} \wedge \mathbf{t}$ *for every pair of trees* $\mathbf{s}$ *and* $\mathbf{t}$*. Some trees are marked to clarify certain notions (contractions, lower expansions, and upper expansions), and their use in derivation of the first entry in the table of representative upper and lower bounds for $n = 9$.*

Figure 3.3 presents the ordering for $n = 6, 7, 8, 9$. Figures 3.2 and 3.3 suggest a number of results about the imbalance ordering.

THEOREM 3.6. *A sequence is on level k of the imbalance partial order (counting down from 0, the topmost and least balanced level)* **iff** *k minimal balancing exchanges are needed to derive it from the least balanced sequence $\langle\ 1\ 2\ 3\ \cdots\ (n-2)\ (n-1)\ (n-1)\ \rangle$. In this situation the sum of the values in the sequence is*

$$\frac{(n+2)(n-1)}{2} \quad - \quad k.$$

Thus the level of a sequence in the partial order is determined by the sum of its path-length values.

Proof. By induction on k. For the basis $k = 0$, the sum of the path-lengths in the least balanced sequence is $(\sum_{i=1}^{n-1} i) + (n-1) = (n+2)(n-1)/2$. For the induction step, consider a sequence whose sum of values is $\frac{(n+2)(n-1)}{2} - k$ with $k > 0$. By Theorem 3.2, this sequence must contain at least three identical values $\langle\, q\, q\, q\, \rangle$. Thus there is a minimal balancing exchange to this sequence from another that contains $\langle\, (q-1)\, (q+1)\, (q+1)\, \rangle$. This sequence is at level $k-1$ by construction, and by induction it has the stated sum. □

Theorem 3.6 shows the significance of the level of a sequence in the imbalance partial order:

DEFINITION 3.7. *The* **level of balance** *of a path-length sequence* s *is*

$$\frac{(n+2)(n-1)}{2} \quad - \quad (\textit{sum of the path-length values in } \mathrm{s}).$$

3.3. Contractions and Expansions of Path-Length Sequences.

DEFINITION 3.8. *Let $\boldsymbol{\ell} = \langle\, \ell_1 \cdots \ell_n \,\rangle$ be a tree path-length sequence of length n. The* **contraction $\boldsymbol{\ell}^-$** *of $\boldsymbol{\ell}$ is the sequence of length $(n-1)$ defined by*

$$\boldsymbol{\ell}^- \quad = \quad \mathrm{sort}{\uparrow}\,(\,\langle\ \ell_1\ \cdots\ \ell_{n-2}\ \ (\ell_{n-1}-1)\ \rangle\,).$$

The **position i expansion of $\boldsymbol{\ell}$** *is the sequence of length $(n+1)$ defined by*

$$\mathrm{sort}{\uparrow}\,(\,\langle\ \ell_1\ \cdots\ \ell_{i-1}\ (\ell_i+1)\ (\ell_i+1)\ \ell_{i+1}\ \cdots\ \ell_n\ \rangle\,).$$

As permitted by Theorem 3.1, if we write

$$\boldsymbol{\ell} \quad = \quad \langle\ \ \cdots\ (q-j)\ \ \overbrace{q\ \ \cdots\ \ q}^{2k}\ \ \rangle$$

with $j,\ k > 0$, then $2k$ is the **suffix length** *of $\boldsymbol{\ell}$, and j is the* **suffix increment** *of $\boldsymbol{\ell}$. The* **contraction $\boldsymbol{\ell}^-$** *is then*

$$\boldsymbol{\ell}^- \quad = \quad \langle\ \ \cdots\ (q-j)\ \ (q-1)\ \ \overbrace{q\ \ \cdots\ \ q}^{2k-2}\ \ \rangle.$$

The **lower expansion $\boldsymbol{\ell}_+$** *is the position $n-2k$ expansion of $\boldsymbol{\ell}$:*

$$\boldsymbol{\ell}_+ \quad = \quad \langle\ \ \cdots\ (q-j+1)\ \ (q-j+1)\ \ \overbrace{q\ \ \cdots\ \ q}^{2k}\ \ \rangle.$$

The **upper expansion $\boldsymbol{\ell}^+$** *is the position n expansion of $\boldsymbol{\ell}$:*

$$\boldsymbol{\ell}^+ \quad = \quad \langle\ \ \cdots\ (q-j)\ \ \overbrace{q\ \ \cdots\ \ q}^{2k-1}\ \ (q+1)\ \ (q+1)\ \ \rangle.$$

Note the definition for $\boldsymbol{\ell}_+$ assumes $2k < n$. When $2k = n$, requiring n to be a power of 2 and $\boldsymbol{\ell} = \langle\ q\ \cdots\ q\ \rangle$ where $q = \log_2(n)$, the preceding formula does not define $\boldsymbol{\ell}_+$. In this very special case we define $\boldsymbol{\ell}_+ = \boldsymbol{\ell}^+$ rather than leave $\boldsymbol{\ell}_+$ undefined.

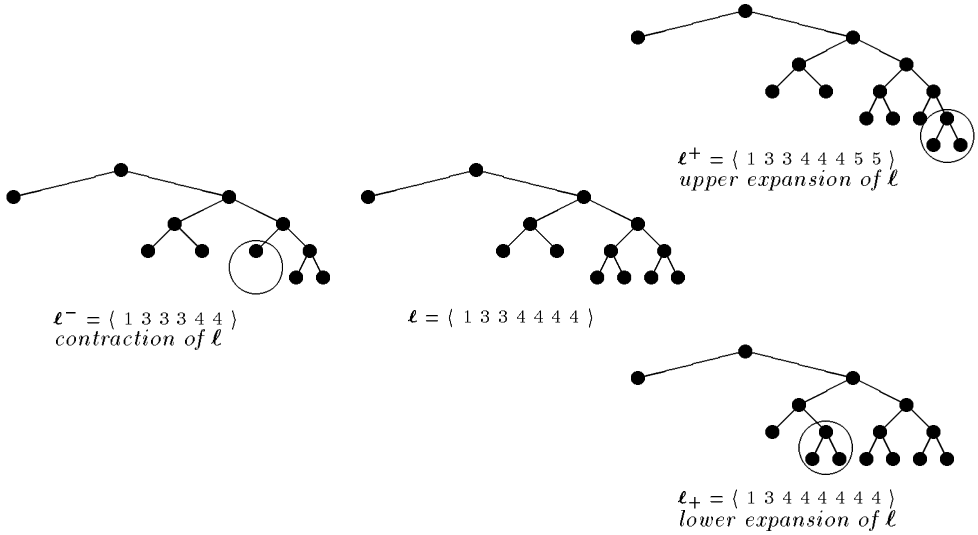

$\boldsymbol{\ell}^+ = \langle\ 1\ 3\ 3\ 4\ 4\ 4\ 5\ 5\ \rangle$
upper expansion of $\boldsymbol{\ell}$
$\boldsymbol{\ell}^- = \langle\ 1\ 3\ 3\ 3\ 4\ 4\ \rangle$
contraction of $\boldsymbol{\ell}$
$\boldsymbol{\ell} = \langle\ 1\ 3\ 3\ 4\ 4\ 4\ 4\ \rangle$
$\boldsymbol{\ell}_+ = \langle\ 1\ 3\ 4\ 4\ 4\ 4\ 4\ 4\ \rangle$
lower expansion of $\boldsymbol{\ell}$

FIG. 3.4. *The contraction and expansions of the path-length sequence* $\boldsymbol{\ell} = \langle\ 1\ 3\ 3\ 4\ 4\ 4\ 4\ \rangle$

path-length sequence	*suffix length*	*suffix incr.*	*contraction*	*lower expansion*	*upper expansion*
$\boldsymbol{\ell}$	$2k$	j	$\boldsymbol{\ell}^-$	$\boldsymbol{\ell}_+$	$\boldsymbol{\ell}^+$
$\langle\ 1\ 2\ 3\ 4\ 5\ 6\ 6\ \rangle$	2	1	$\langle\ 1\ 2\ 3\ 4\ 5\ \mathbf{5}\ \rangle$	$\langle\ 1\ 2\ 3\ 4\ \mathbf{6}\ \mathbf{6}\ 6\ 6\ \rangle$	$\langle\ 1\ 2\ 3\ 4\ 5\ 6\ \mathbf{7}\ \mathbf{7}\ \rangle$
$\langle\ 1\ 2\ 4\ 4\ 4\ 5\ 5\ \rangle$	2	1	$\langle\ 1\ 2\ 4\ 4\ 4\ \mathbf{4}\ \rangle$	$\langle\ 1\ 2\ 4\ 4\ \mathbf{5}\ \mathbf{5}\ 5\ 5\ \rangle$	$\langle\ 1\ 2\ 4\ 4\ 4\ 5\ \mathbf{6}\ \mathbf{6}\ \rangle$
$\langle\ 1\ 2\ 3\ 5\ 5\ 5\ 5\ \rangle$	4	2	$\langle\ 1\ 2\ 3\ \mathbf{4}\ 5\ 5\ \rangle$	$\langle\ 1\ 2\ \mathbf{4}\ \mathbf{4}\ 5\ 5\ 5\ 5\ \rangle$	$\langle\ 1\ 2\ 3\ 5\ 5\ 5\ \mathbf{6}\ \mathbf{6}\ \rangle$
$\langle\ 1\ 3\ 3\ 3\ 4\ 5\ 5\ \rangle$	2	1	$\langle\ 1\ 3\ 3\ 3\ 4\ \mathbf{4}\ \rangle$	$\langle\ 1\ 3\ 3\ 3\ \mathbf{5}\ \mathbf{5}\ 5\ 5\ \rangle$	$\langle\ 1\ 3\ 3\ 3\ 4\ 5\ \mathbf{6}\ \mathbf{6}\ \rangle$
$\boxed{\langle\ 1\ 3\ 3\ 4\ 4\ 4\ 4\ \rangle}$	4	1	$\boxed{\langle\ 1\ 3\ 3\ \mathbf{3}\ 4\ 4\ \rangle}$	$\boxed{\langle\ 1\ 3\ \mathbf{4}\ \mathbf{4}\ 4\ 4\ 4\ 4\ \rangle}$	$\boxed{\langle\ 1\ 3\ 3\ 4\ 4\ 4\ \mathbf{5}\ \mathbf{5}\ \rangle}$
$\langle\ 2\ 2\ 2\ 3\ 4\ 5\ 5\ \rangle$	2	1	$\langle\ 2\ 2\ 2\ 3\ 4\ \mathbf{4}\ \rangle$	$\langle\ 2\ 2\ 2\ 3\ \mathbf{5}\ \mathbf{5}\ 5\ 5\ \rangle$	$\langle\ 2\ 2\ 2\ 3\ 4\ 5\ \mathbf{6}\ \mathbf{6}\ \rangle$
$\langle\ 2\ 2\ 2\ 4\ 4\ 4\ 4\ \rangle$	4	2	$\langle\ 2\ 2\ 2\ \mathbf{3}\ 4\ 4\ \rangle$	$\langle\ 2\ 2\ \mathbf{3}\ \mathbf{3}\ 4\ 4\ 4\ 4\ \rangle$	$\langle\ 2\ 2\ 2\ 4\ 4\ 4\ \mathbf{5}\ \mathbf{5}\ \rangle$
$\langle\ 2\ 2\ 3\ 3\ 3\ 4\ 4\ \rangle$	2	1	$\langle\ 2\ 2\ 3\ 3\ 3\ \mathbf{3}\ \rangle$	$\langle\ 2\ 2\ 3\ 3\ \mathbf{4}\ \mathbf{4}\ 4\ 4\ \rangle$	$\langle\ 2\ 2\ 3\ 3\ 3\ 4\ \mathbf{5}\ \mathbf{5}\ \rangle$
$\langle\ 2\ 3\ 3\ 3\ 3\ 3\ 3\ \rangle$	6	1	$\langle\ 2\ \mathbf{2}\ 3\ 3\ 3\ 3\ \rangle$	$\langle\ \mathbf{3}\ \mathbf{3}\ 3\ 3\ 3\ 3\ 3\ 3\ \rangle$	$\langle\ 2\ 3\ 3\ 3\ 3\ 3\ \mathbf{4}\ \mathbf{4}\ \rangle$

TABLE 3.1
Path-length sequences of length 7, with their contractions and expansions. Note that all contractions have length 6, and expansions length 8. Emboldened digits reflect changes from $\boldsymbol{\ell}$*.*

These definitions will be used heavily throughout the rest of the paper. Figure 3.4 and Table 3.1 give examples for $n = 7$. Figure 3.3 also gives examples illustrating the relationships these definitions produce among the imbalance orderings for successive values of n.

THEOREM 3.9. *If* $\boldsymbol{\ell}$ *is a path-length sequence,* $\boldsymbol{\ell}_+ \trianglelefteq \boldsymbol{\ell}^+$ *and* $(\boldsymbol{\ell}_+)^- \trianglelefteq (\boldsymbol{\ell}^+)^- = \boldsymbol{\ell}$*. Furthermore, either* $\boldsymbol{\ell} = (\boldsymbol{\ell}^-)_+$*, or* $\boldsymbol{\ell} = (\boldsymbol{\ell}^-)^+$*. Thus* $(\boldsymbol{\ell}^-)_+ \trianglelefteq \boldsymbol{\ell} \trianglelefteq (\boldsymbol{\ell}^-)^+$*.*

Proof. $\boldsymbol{\ell}_+$ and $\boldsymbol{\ell}^+$ differ by a ternary exchange, so $\boldsymbol{\ell}_+ \trianglelefteq \boldsymbol{\ell}^+$. From Theorem 3.1 we can assume $\boldsymbol{\ell} = \langle \cdots (q-j)\ \overbrace{q \cdots q}^{2k} \rangle$, and thus $(\boldsymbol{\ell}^+)^- = \boldsymbol{\ell}$. Furthermore $(\boldsymbol{\ell}_+)^- = \boldsymbol{\ell}$ if $j = 1$ and $(\boldsymbol{\ell}_+)^- = \langle \cdots (q-j+1)(q-j+1)(q-1)\ \overbrace{q \cdots q}^{2k-2} \rangle \trianglelefteq \boldsymbol{\ell}$ if $j > 1$. Finally $(\boldsymbol{\ell}^-)^+ = \boldsymbol{\ell}$ if $k = 1$ (necessitating $j = 1$) and $(\boldsymbol{\ell}^-)_+ = \boldsymbol{\ell}$ if $k > 1$. Consequently $\boldsymbol{\ell} \in \{\ (\boldsymbol{\ell}^-)_+,\ (\boldsymbol{\ell}^-)^+\ \}$. □

THEOREM 3.10. *If* $\mathrm{s} \trianglelefteq \mathbf{t}$*, then* $\mathrm{s}^- \trianglelefteq \mathbf{t}^-$*,* $\mathrm{s}_+ \trianglelefteq \mathbf{t}_+$*, and* $\mathrm{s}^+ \trianglelefteq \mathbf{t}^+$*.*

Proof. Recall that if $\mathrm{s} \trianglelefteq \mathbf{t}$, then there are sequences $\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_m$ $(m \geq 1)$ such that $\mathbf{t} = \boldsymbol{\ell}_1$, $\boldsymbol{\ell}_m = \mathrm{s}$ and for each i, $1 \leq i < m$, there is a balancing exchange from $\boldsymbol{\ell}_i$ to $\boldsymbol{\ell}_{i+1}$. Our approach here is very simple: to prove $\mathrm{s}^- \trianglelefteq \mathbf{t}^-$ we convert the derivation $\mathbf{t} = \boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_m = \mathrm{s}$ directly to the derivation $\mathbf{t}^- = \boldsymbol{\ell}_1{}^-, \ldots, \boldsymbol{\ell}_m{}^- = \mathrm{s}^-$. For this it is sufficient to show that either each step from $(\boldsymbol{\ell}_i)^-$ to $(\boldsymbol{\ell}_{i+1})^-$ is a balancing exchange, or $(\boldsymbol{\ell}_i)^- = (\boldsymbol{\ell}_{i+1})^-$. The former must hold if $\boldsymbol{\ell}_i$ and $\boldsymbol{\ell}_{i+1}$ agree in the final two positions. If they disagree,

$$\begin{array}{rcl} \boldsymbol{\ell}_i & = & \langle\ \cdots \quad\quad p \quad\quad\quad a \ \cdots \ b \quad q \quad q \quad \rangle, \\ \boldsymbol{\ell}_{i+1} & = & \langle\ \cdots \ (p{+}1) \ (p{+}1) \quad a \ \cdots \ b \quad (q{-}1) \quad \rangle \end{array}$$

because they define a balancing exchange, and by Theorem 3.1 necessarily $b = (q{-}1)$. Then

$$\begin{array}{rcl} (\boldsymbol{\ell}_i)^- & = & \mathrm{sort}\!\uparrow(\langle\ \cdots \quad\quad p \quad\quad\quad a \ \cdots \ (q{-}1) \ \ (q{-}1) \quad \rangle), \\ (\boldsymbol{\ell}_{i+1})^- & = & \mathrm{sort}\!\uparrow(\langle\ \cdots \ (p{+}1) \ (p{+}1) \quad a \ \cdots \quad (q{-}2) \quad\quad \rangle). \end{array}$$

If $(p{+}1) = b = (q{-}1)$, then $p = (q{-}2)$ and the two contractions are equal. If not, they still differ by a balancing exchange.
Proving $\mathrm{s}_+ \trianglelefteq \mathbf{t}_+$ is similar, where $(\boldsymbol{\ell}_i)_+ = (\boldsymbol{\ell}_{i+1})_+$ iff $\boldsymbol{\ell}_i = \langle\ \cdots\ (q{-}j)\ q\ q\ \cdots\ q\ \rangle$, $\boldsymbol{\ell}_{i+1} = \langle\ \cdots\ (q{-}j{+}1)\ (q{-}j{+}1)\ (q{-}1)\ \cdots\ q\ \rangle$ and $j \geq 2$. Proving $\mathrm{s}^+ \trianglelefteq \mathbf{t}^+$ is also similar, but easier, since then it is never the case that $(\boldsymbol{\ell}_i)^+ = (\boldsymbol{\ell}_{i+1})^+$. □

3.4. The Vector Lattice and Distribution Lattice. Recall [5] that a **lattice** is an algebra $\langle \mathcal{S}, \sqsubseteq, \sqcap, \sqcup \rangle$ in which $\mathcal{S}$ is a set, $\sqsubseteq$ is a partial ordering on $\mathcal{S}$, and for all a, $b \in \mathcal{S}$, there is a unique **greatest lower bound (glb)** $a \sqcap b$ and **least upper bound (lub)** $a \sqcup b$. The lattice is called **distributive** if these operators satisfy the distributive law:

$$\text{for all } a, b, c \text{ in } \mathcal{S}, \quad a \sqcap (b \sqcup c) \quad = \quad (a \sqcap b) \sqcup (a \sqcap c).$$

Optionally the lattice can have a **greatest element** $\top$ and **least element** $\bot$.

DEFINITION 3.11. *Let* $\leq_{vec}$ *be the element-wise ordering on vectors (sequences) in* $\Re^n$. *Then*

$$\mathbf{x} \ \leq_{vec} \ \mathbf{y} \quad \textbf{iff} \quad x_i \ \leq \ y_i \ \ \textit{for } 1 \leq i \leq n.$$

Also define vector element-wise minima and maxima

$$\begin{array}{rcl} \mathbf{x} \ \ \mathbf{min}_{vec} \ \ \mathbf{y} & = & \langle \ \min(x_1, y_1) \ \cdots \ \min(x_n, y_n) \ \rangle \\ \mathbf{x} \ \ \mathbf{max}_{vec} \ \ \mathbf{y} & = & \langle \ \max(x_1, y_1) \ \cdots \ \max(x_n, y_n) \ \rangle. \end{array}$$

THEOREM 3.12. *The nonnegative vectors* $\langle \Re_+{}^n, \leq_{vec}, \mathbf{min}_{vec}, \mathbf{max}_{vec} \rangle$ *form a distributive lattice called the* **vector lattice**.

The set $\mathcal{P}$ *of distribution sequences of length n (ascending nonnegative vectors* $\mathbf{v}$ *with* $v_n = 1$*) also form a distributive lattice,* $\langle \mathcal{P}, \leq_{vec}, \mathbf{min}_{vec}, \mathbf{max}_{vec} \rangle$, *called the* **distribution lattice**, *with least element* $\bot = \langle\, 0\, 0 \cdots 0\, 1 \,\rangle$ *and greatest element* $\top = \langle\, 1\, 1 \cdots 1\, 1 \,\rangle$.

Proof. The one-dimensional algebra $\langle \Re_+, \leq, \min, \max \rangle$ is a distributive lattice. The vector properties required here follow from this. □

3.5. The Majorization Lattice and Density Lattice. We reproduce basic majorization concepts developed in the paper [34]. Majorization as defined here is an extension of the classical majorization of Muirhead and Hardy-Littlewood-Pólya [16], useful in the study of inequalities. Marshall and Olkin [27] provides a very good account of the classical theory and its applications. The classical theory defines a majorization ordering on descendingly-ordered (or sometimes ascendingly-ordered) multisets, and although quite beautiful is also quite complex. We have transplanted the theory to rely only on linear algebra and convexity. Thus the definitions in this section are ours, and the results vary from those in [27].

DEFINITION 3.13. *The* **Zeta matrix** $\int = (\zeta_{ij})$ *is defined by*

$$\zeta_{ij} \ = \ 1 \textit{ if } i \geq j, \quad 0 \textit{ otherwise.}$$

The **Möbius matrix** $\boldsymbol{\partial} = (\mu_{ij})$ *is defined by*

$$\mu_{ij} \ = \ 1 \textit{ if } i = j, \quad -1 \textit{ if } j = i - 1, \quad 0 \textit{ otherwise.}$$

The Möbius matrix is the inverse of the Zeta matrix. For example, when $n = 5$:

$$\int \;=\; \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix} \qquad \boldsymbol{\partial} \;=\; \int^{-1} \;=\; \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 1 \end{pmatrix}.$$

The Möbius matrix is also significant in that it corresponds directly to the concept of pairwise exchange (of adjacent elements in a sequence). The theory of Möbius inversion [36] gives a generalized notion of differential on partially-ordered domains (although here we consider only totally-ordered sequences). We can think of $\int$ as an integral operator (which transforms a sequence to its left-to-right "integral"), with $\boldsymbol{\partial}$ as its inverse differential operator.

THEOREM 3.14. *If* $\mathbf{x}$ *and* $\mathbf{y}$ *are density sequences (so* $\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i = 1$*), then*

$$\mathbf{x} \preceq \mathbf{y} \quad \textbf{iff} \quad (\textstyle\int \mathbf{x}) \leq_{vec} (\textstyle\int \mathbf{y}).$$

If $\mathbf{v}$ *and* $\mathbf{v}'$ *are distribution sequences, then* $\boldsymbol{\partial}\mathbf{v}$ *and* $\boldsymbol{\partial}\mathbf{v}'$ *are density sequences and*

$$\mathbf{v} \leq_{vec} \mathbf{v}' \quad \textbf{iff} \quad (\boldsymbol{\partial}\mathbf{v}) \preceq (\boldsymbol{\partial}\mathbf{v}').$$

Proof. $(\int \mathbf{x}) \leq_{vec} (\int \mathbf{y})$ is equivalent to:

$$x_1 \leq y_1, \qquad x_1 + x_2 \leq y_1 + y_2, \qquad \ldots \qquad x_1 + x_2 + \cdots + x_n \leq y_1 + y_2 + \cdots + y_n.$$

Note $\mathbf{x}$ is a density sequence iff $(\int \mathbf{x})$ is a distribution sequence. The second statement then follows since the Möbius and Zeta transformations are inverses of one another. □

This isomorphism between $\leq_{vec}$ and $\preceq$ implies that majorization defines a lattice.

DEFINITION 3.15. *Majorization lub and glb operators are definable by*

$$\begin{aligned} \mathbf{x} \sqcup \mathbf{y} &= \boldsymbol{\partial}((\textstyle\int \mathbf{x}) \ \mathbf{max}_{vec} \ (\textstyle\int \mathbf{y})) \\ \mathbf{x} \sqcap \mathbf{y} &= \boldsymbol{\partial}((\textstyle\int \mathbf{x}) \ \mathbf{min}_{vec} \ (\textstyle\int \mathbf{y})). \end{aligned}$$

THEOREM 3.16. *The nonnegative reals ordered by majorization forms a distributive lattice* $\langle {\Re_+}^n, \preceq, \sqcap, \sqcup \rangle$ *called the* **majorization lattice***.*

The set $\mathcal{D}$ *of density sequences of length n (nonnegative* $\mathbf{x}$ *with* $\sum_{i=1}^{n} x_i = 1$*) forms a distributive lattice* $\langle \mathcal{D}, \preceq, \sqcap, \sqcup, \bot, \top \rangle$ *called the* **density lattice***, with least element* $\bot = \langle 0\,0 \cdots 0\,1 \rangle$ *and greatest element* $\top = \langle 1\,0 \cdots 0\,0 \rangle$*.*

Proof. The transformation $\mathbf{x} \mapsto \int \mathbf{x}$ defines a lattice isomorphism between the vector and majorization lattices, and between the distribution and density lattices. Here $\mathbf{x} \sqcap \mathbf{y}$ and $\mathbf{x} \sqcup \mathbf{y}$ are defined just so as to be the majorization glb and lub:

$$\begin{array}{ll|ll} & \mathbf{z} \preceq \mathbf{x}, \quad \mathbf{z} \preceq \mathbf{y} & & \mathbf{x} \preceq \mathbf{z}, \quad \mathbf{y} \preceq \mathbf{z} \\ \Leftrightarrow & (\int \mathbf{z}) \leq_{vec} (\int \mathbf{x}), \ (\int \mathbf{z}) \leq_{vec} (\int \mathbf{y}) & \Leftrightarrow & (\int \mathbf{x}) \leq_{vec} (\int \mathbf{z}), \ (\int \mathbf{y}) \leq_{vec} (\int \mathbf{z}) \\ \Leftrightarrow & (\int \mathbf{z}) \leq_{vec} ((\int \mathbf{x}) \ \mathbf{min}_{vec} (\int \mathbf{y})) & \Leftrightarrow & ((\int \mathbf{x}) \ \mathbf{max}_{vec} (\int \mathbf{y})) \leq_{vec} (\int \mathbf{z}) \\ \Leftrightarrow & \mathbf{z} \preceq \boldsymbol{\partial}((\int \mathbf{x}) \ \mathbf{min}_{vec} (\int \mathbf{y})) & \Leftrightarrow & \boldsymbol{\partial}((\int \mathbf{x}) \ \mathbf{max}_{vec} (\int \mathbf{y})) \preceq \mathbf{z} \\ \Leftrightarrow & \mathbf{z} \preceq \mathbf{x} \sqcap \mathbf{y}. & \Leftrightarrow & \mathbf{x} \sqcup \mathbf{y} \preceq \mathbf{z}. \end{array}$$

Thus the majorization algebra also forms a distributive lattice. □

Even when $\mathbf{x}$ and $\mathbf{y}$ are in descending order, the sequences $(\mathbf{x} \sqcap \mathbf{y})$ and $(\mathbf{x} \sqcup \mathbf{y})$ defined here are not necessarily in descending order: $\mathbf{x} = \langle\, 2^{-2}\ 2^{-2}\ 2^{-3}\ 2^{-4}\ 2^{-4}\ 2^{-4}\ 2^{-4}\ 2^{-4}\ 2^{-4} \,\rangle$ and $\mathbf{y} = \langle\, 2^{-2}\ 2^{-3}\ 2^{-3}\ 2^{-3}\ 2^{-3}\ 2^{-3}\ 2^{-4}\ 2^{-5}\ 2^{-5} \,\rangle$ yield the least upper bound $\mathbf{x} \sqcup \mathbf{y} = \boldsymbol{\partial}(\int \mathbf{x} \ \mathbf{max}_{vec} \int \mathbf{y}) = \langle\, 2^{-2}\ 2^{-2}\ 2^{-3}\ \boxed{2^{-4}}\ 2^{-4}\ \boxed{2^{-3}}\ 2^{-4}\ 2^{-5}\ 2^{-5} \,\rangle$. See Figure 3.6.

$$\begin{array}{rcccccccccccc}
\mathbf{s} & = & & (& 1 & 2 & 4 & 4 & 5 & 5 & 5 & 6 & 6 &) \\
2^{-\mathbf{s}} & = & & (& 2^{-1} & 2^{-2} & 2^{-4} & 2^{-4} & 2^{-5} & 2^{-5} & 2^{-5} & 2^{-6} & 2^{-6} &) \\
\int 2^{-\mathbf{s}} & = & 2^{-7} & (& 64 & 96 & 104 & 112 & 116 & 120 & 124 & 126 & 128 &) \\
\mathbf{t} & = & & (& 2 & 2 & 2 & 3 & 4 & 5 & 6 & 7 & 7 &) \\
2^{-\mathbf{t}} & = & & (& 2^{-2} & 2^{-2} & 2^{-2} & 2^{-3} & 2^{-4} & 2^{-5} & 2^{-6} & 2^{-7} & 2^{-7} &) \\
\int 2^{-\mathbf{t}} & = & 2^{-7} & (& 32 & 64 & 96 & 112 & 120 & 124 & 126 & 127 & 128 &) \\
\int 2^{-\mathbf{s}} \ \mathbf{max}_{vec} \int 2^{-\mathbf{t}} & = & 2^{-7} & (& 64 & 96 & 104 & 112 & 120 & 124 & 126 & 127 & 128 &) \\
\partial(\int 2^{-\mathbf{s}} \ \mathbf{max}_{vec} \int 2^{-\mathbf{t}}) & = & 2^{-7} & (& 64 & 32 & 8 & 8 & 8 & 4 & 2 & 1 & 1 &) \\
= \ 2^{-\mathbf{s}} \sqcup 2^{-\mathbf{t}} & = & & (& 2^{-1} & 2^{-2} & 2^{-4} & 2^{-4} & 2^{-4} & 2^{-5} & 2^{-6} & 2^{-7} & 2^{-7} &) \\
\mathbf{s} \vee \mathbf{t} & = & & (& 1 & 2 & 4 & 4 & 4 & 5 & 6 & 7 & 7 &) \\
\int 2^{-\mathbf{s}} \ \mathbf{min}_{vec} \int 2^{-\mathbf{t}} & = & 2^{-7} & (& 32 & 64 & 96 & 112 & 116 & 120 & 124 & 126 & 128 &) \\
\partial(\int 2^{-\mathbf{s}} \ \mathbf{min}_{vec} \int 2^{-\mathbf{t}}) & = & 2^{-7} & (& 32 & 32 & 32 & 16 & 4 & 4 & 4 & 2 & 2 &) \\
= \ 2^{-\mathbf{s}} \sqcap 2^{-\mathbf{t}} & = & & (& 2^{-2} & 2^{-2} & 2^{-2} & 2^{-3} & 2^{-5} & 2^{-5} & 2^{-5} & 2^{-6} & 2^{-6} &) \\
\mathbf{s} \wedge \mathbf{t} & = & & (& 2 & 2 & 2 & 3 & 5 & 5 & 5 & 6 & 6 &)
\end{array}$$

FIG. 3.5. *Related points in the majorization and imbalance lattices, showing their connection*

$$\begin{array}{rcccccccccccc}
\mathbf{s} & = & & (& 2 & 2 & 3 & 4 & 4 & 4 & 4 & 4 & 4 &) \\
2^{-\mathbf{s}} & = & & (& 2^{-2} & 2^{-2} & 2^{-3} & 2^{-4} & 2^{-4} & 2^{-4} & 2^{-4} & 2^{-4} & 2^{-4} &) \\
\int 2^{-\mathbf{s}} & = & 2^{-7} & (& 32 & 64 & 80 & 88 & 96 & 104 & 112 & 120 & 128 &) \\
\mathbf{t} & = & & (& 2 & 3 & 3 & 3 & 3 & 3 & 4 & 5 & 5 &) \\
2^{-\mathbf{t}} & = & & (& 2^{-2} & 2^{-3} & 2^{-3} & 2^{-3} & 2^{-3} & 2^{-3} & 2^{-4} & 2^{-5} & 2^{-5} &) \\
\int 2^{-\mathbf{t}} & = & 2^{-7} & (& 32 & 48 & 64 & 80 & 96 & 112 & 120 & 124 & 128 &) \\
\int 2^{-\mathbf{s}} \ \mathbf{max}_{vec} \int 2^{-\mathbf{t}} & = & 2^{-7} & (& 32 & 64 & 80 & 88 & 96 & 112 & 120 & 124 & 128 &) \\
\partial(\int 2^{-\mathbf{s}} \ \mathbf{max}_{vec} \int 2^{-\mathbf{t}}) & = & 2^{-7} & (& 32 & 32 & 16 & 8 & 8 & 16 & 8 & 4 & 4 &) \\
= \ 2^{-\mathbf{s}} \sqcup 2^{-\mathbf{t}} & = & & (& 2^{-2} & 2^{-2} & 2^{-3} & \boxed{2^{-4}} & 2^{-4} & \boxed{2^{-3}} & 2^{-4} & 2^{-5} & 2^{-5} &) \\
\mathbf{s} \vee \mathbf{t} & = & & (& 2 & 2 & 3 & \boxed{3} & 4 & \boxed{4} & 4 & 5 & 5 &)
\end{array}$$

FIG. 3.6. *Results of* $(2^{-\mathbf{s}} \sqcap 2^{-\mathbf{t}})$ *and* $(2^{-\mathbf{s}} \sqcup 2^{-\mathbf{t}})$ *are not necessarily in descending order*

$$\begin{array}{rcccccccccccc}
\mathbf{s} & = & & (& 1 & 2 & 4 & 5 & 5 & 5 & 5 & 5 & 5 &) \\
2^{-\mathbf{s}} & = & & (& 2^{-1} & 2^{-2} & 2^{-4} & 2^{-5} & 2^{-5} & 2^{-5} & 2^{-5} & 2^{-5} & 2^{-5} &) \\
\int 2^{-\mathbf{s}} & = & 2^{-7} & (& 64 & 96 & 104 & 108 & 112 & 116 & 120 & 124 & 128 &) \\
\mathbf{t} & = & & (& 2 & 2 & 2 & 3 & 4 & 5 & 6 & 7 & 7 &) \\
2^{-\mathbf{t}} & = & & (& 2^{-2} & 2^{-2} & 2^{-2} & 2^{-3} & 2^{-4} & 2^{-5} & 2^{-6} & 2^{-7} & 2^{-7} &) \\
\int 2^{-\mathbf{t}} & = & 2^{-7} & (& 32 & 64 & 96 & 112 & 120 & 124 & 126 & 127 & 128 &) \\
\int 2^{-\mathbf{s}} \ \mathbf{min}_{vec} \int 2^{-\mathbf{t}} & = & 2^{-7} & (& 32 & 64 & 96 & 108 & 112 & 116 & 120 & 124 & 128 &) \\
\partial(\int 2^{-\mathbf{s}} \ \mathbf{min}_{vec} \int 2^{-\mathbf{t}}) & = & 2^{-7} & (& 32 & 32 & 32 & 12 & 4 & 4 & 4 & 4 & 4 &) \\
= \ 2^{-\mathbf{s}} \sqcap 2^{-\mathbf{t}} & = & & (& 2^{-2} & 2^{-2} & 2^{-2} & \boxed{2^{-\alpha}} & \boxed{2^{-5}} & 2^{-5} & 2^{-5} & 2^{-5} & 2^{-5} &) \\
 & & & & & & & \multicolumn{6}{l}{-\alpha \ = \ (-7 + \log_2(12)) \ \approx \ -3.4150375} & \\
\succeq & & & (& 2^{-2} & 2^{-2} & 2^{-2} & \boxed{2^{-4}} & \boxed{2^{-4}} & 2^{-5} & 2^{-5} & 2^{-5} & 2^{-5} &) \\
\mathbf{s} \wedge \mathbf{t} & = & & (& 2 & 2 & 2 & \boxed{4} & \boxed{4} & 5 & 5 & 5 & 5 &)
\end{array}$$

$2^{-(\mathbf{s} \wedge \mathbf{t})} \preceq (2^{-\mathbf{s}} \sqcap 2^{-\mathbf{t}})$; the two differ where indicated. Non-integral exponents occur for $n \geq 9$.

FIG. 3.7. *The imbalance lattice is not simply conjugate to a sublattice of the majorization lattice*

3.6. The Imbalance Lattice: A Discrete Cousin of the Majorization Lattice. Since every pair of sequences in Figures 3.2 and 3.3 has a unique glb and lub, the imbalance ordering is not only a partial order, but also a *lattice*. In this section we prove this by showing that every pair of sequences $\mathbf{s}$, $\mathbf{t}$ has a glb $\mathbf{s} \wedge \mathbf{t}$ and lub $\mathbf{s} \vee \mathbf{t}$. We also relate the **imbalance lattice** directly to the majorization lattice, as illustrated in Figures 3.5—3.7.

THEOREM 3.17. *On tree path-length sequences, the imbalance ordering is isomorphic to the majorization ordering. Specifically, whenever* $\mathbf{s}$ *and* $\mathbf{t}$ *are tree path-length sequences, then*

$$\mathbf{s} \;\trianglelefteq\; \mathbf{t} \quad \textbf{iff} \quad 2^{-\mathbf{s}} \;\preceq\; 2^{-\mathbf{t}}.$$

Proof. We show first that balancing exchanges cause a reduction in the majorization ordering. Let $\mathbf{s}$ be the result of a balancing exchange on $\mathbf{t}$ (so $\mathbf{s} \trianglelefteq \mathbf{t}$). Then:

$$\begin{array}{rclllllllllllll}
\mathbf{t} & = & (& \cdot & \cdot & \cdot & p & u & \cdot & \cdot & v & (q+1) & (q+1) & \cdot & \cdot\,) \\
\mathbf{s} & = & (& \cdot & \cdot & \cdot & (p+1) & (p+1) & u & \cdot & \cdot & v & q & \cdot & \cdot\,) \\
 & & & & & & & & & & & & & & \\
2^{-\mathbf{t}} & = & (& \cdot & \cdot & \cdot & 2^{-p} & 2^{-u} & \cdot & \cdot & 2^{-v} & 2^{-(q+1)} & 2^{-(q+1)} & \cdot & \cdot\,) \\
2^{-\mathbf{s}} & = & (& \cdot & \cdot & \cdot & 2^{-(p+1)} & 2^{-(p+1)} & 2^{-u} & \cdot & \cdot & 2^{-v} & 2^{-q} & \cdot & \cdot\,) \\
 & & & & & & & & & & & & & & \\
\int 2^{-\mathbf{t}} & = & (& \cdot & \cdot & S & S+2^{-p} & S+2^{-p}+2^{-u} & \cdot & \cdot & T & T+2^{-(q+1)} & T+2^{-q} & \cdot & 1\,) \\
\int 2^{-\mathbf{s}} & = & (& \cdot & \cdot & S & S+2^{-(p+1)} & S+2^{-p} & \cdot & \cdot & T-2^{-v} & T & T+2^{-q} & \cdot & 1\,) \\
\hline
 & & (& 0 & \cdot & 0 & +2^{-(p+1)} & +2^{-u} & \cdot & \cdot & +2^{-v} & +2^{-(q+1)} & 0 & \cdot & 0\,)
\end{array}$$

Thus $\int 2^{-\mathbf{s}}$ and $\int 2^{-\mathbf{t}}$ differ only in the values appearing between p and q, and each element in $\int 2^{-\mathbf{t}} - \int 2^{-\mathbf{s}}$ is nonnegative, so $2^{-\mathbf{s}} \preceq 2^{-\mathbf{t}}$.

The proof of the converse, that $2^{-\mathbf{s}} \preceq 2^{-\mathbf{t}}$ implies $\mathbf{s} \trianglelefteq \mathbf{t}$ for tree path-length sequences $\mathbf{s}$, $\mathbf{t}$, can proceed by assuming a counterexample for which the difference in the levels of balance

$$m \quad = \quad (\text{level of balance of } \mathbf{s}) \;-\; (\text{level of balance of } \mathbf{t}).$$

is minimal. Since $2^{-\mathbf{s}} \preceq 2^{-\mathbf{t}}$, let a, b, c, d be the rightmost aligned pairwise-differing values among the two sorted sequences such that $\mathbf{s} = \langle \cdots\ a\ \cdots\ b\ \cdots \rangle$ and $\mathbf{t} = \langle \cdots\ c\ \cdots\ d\ \cdots \rangle$, where $c < a$, $b < d$ because of the majorization inequality, $a \le b$ and $c \le d$ because the sequences are ascending, $c \ne d$ since $c < a \le b < d$, and finally $2^{-a} + \cdots + 2^{-b} = 2^{-c} + \cdots + 2^{-d}$, which is always possible by the Kraft equality. Because $b < d$ necessarily $\mathbf{t} = \langle \cdots\ c\ \cdots\ d\ d\ \cdots \rangle$, since otherwise we reach a contradiction (multiplying both sides of the equality by 2^d makes the left side even but the right side odd). Thus, if we define the result $\mathbf{t}' = \langle \cdots\ (c+1)\ (c+1)\ \cdots\ (d-1)\ \cdots \rangle$ of a balancing exchange on $\mathbf{t} = \langle \cdots\ c\ \cdots\ d\ d\ \cdots \rangle$, then the level difference between $\mathbf{s}$ and $\mathbf{t}'$ is at most $(m-1)$, and $2^{-\mathbf{t}'} \preceq 2^{-\mathbf{t}}$. Furthermore we claim $2^{-\mathbf{s}} \preceq 2^{-\mathbf{t}'}$, using the following schematic:

$$\begin{array}{rclllllllllll}
\mathbf{t} & = & (& \cdot & \cdot & \cdot & c & u & \cdot & d & d & \cdot & \cdot\,) \\
\mathbf{t}' & = & (& \cdot & \cdot & \cdot & (c+1) & (c+1) & \cdot & \cdot & (d-1) & \cdot & \cdot\,) \\
\mathbf{s} & = & (& \cdot & \cdot & \cdot & a & w & \cdot & \cdot & b & \cdot & \cdot\,) \\
 & & & & & & & & & & & & \\
2^{-\mathbf{t}} & = & (& \cdot & \cdot & \cdot & 2^{-c} & 2^{-u} & \cdot & 2^{-d} & 2^{-d} & \cdot & \cdot\,) \\
2^{-\mathbf{t}'} & = & (& \cdot & \cdot & \cdot & 2^{-(c+1)} & 2^{-(c+1)} & \cdot & \cdot & 2^{-(d-1)} & \cdot & \cdot\,) \\
2^{-\mathbf{s}} & = & (& \cdot & \cdot & \cdot & 2^{-a} & 2^{-w} & \cdot & \cdot & 2^{-b} & \cdot & \cdot\,) \\
 & & & & & & & & & & & & \\
\int 2^{-\mathbf{t}} - \int 2^{-\mathbf{s}} & = & (& 0 & \cdot & 0 & +(2^{-c} - 2^{-a}) & +S_1 & \cdot & +S_k & 0 & \cdot & 0\,) \\
\int 2^{-\mathbf{t}} - \int 2^{-\mathbf{t}'} & = & (& 0 & \cdot & 0 & +2^{-(c+1)} & +2^{-u} & \cdot & +2^{-d} & 0 & \cdot & 0\,) \\
\hline
\int 2^{-\mathbf{t}'} - \int 2^{-\mathbf{s}} & = & (& 0 & \cdot & 0 & +(2^{-(c+1)} - 2^{-a}) & +(S_1 - 2^{-u}) & \cdot & +(S_k - 2^{-d}) & 0 & \cdot & 0\,)
\end{array}$$

Because $2^{-\mathbf{s}} \preceq 2^{-\mathbf{t}}$, the running totals $S_1, \ldots, S_k$ are nonnegative. Also, $(2^{-(c+1)} - 2^{-a}) \ge 0$ since $c < a$. Furthermore $c \le (a-1) \le w$, implying $S_1 - 2^{-u} = (2^{-c} - 2^{-a}) - 2^{w} \ge 2^{-(a-1)} - 2^{-w} \ge 0$. Finally $S_k + 2^{-d} - 2^{-b} = 0$, so $b \le (d-1)$ implies $S_k - 2^{-d} = (2^{-b} - 2^{-d}) - 2^{-d} = 2^{-b} - 2^{-(d-1)} \ge 0$. Thus $\int 2^{-\mathbf{s}} \le_{vec} \int 2^{-\mathbf{t}'}$ (i.e., $2^{-\mathbf{s}} \preceq 2^{-\mathbf{t}'}$), contradicting the assumed minimality of m, and existence of a counterexample. □

THEOREM 3.18. *The imbalance ordering on binary trees determines a bona-fide lattice in which, for all* s *and* t, *the glb* $\mathrm{s} \wedge \mathrm{t}$ *and lub* $\mathrm{s} \vee \mathrm{t}$ *are defined with the following recursive algorithms, where the expansion used is chosen from among the lower and upper expansions:*

$$\mathrm{s} \wedge \mathrm{t} \quad = \quad \begin{cases} \mathrm{s} & \textit{if} \quad \mathrm{s} \trianglelefteq \mathrm{t} \\ \mathrm{t} & \textit{if} \quad \mathrm{t} \trianglelefteq \mathrm{s} \\ \textit{the greatest expansion of } \ \mathrm{s}^- \wedge \mathrm{t}^- & \\ \textit{that is also a lower bound for } \mathrm{s} \textit{ and } \mathrm{t} & \textit{otherwise}; \end{cases}$$

$$\mathrm{s} \vee \mathrm{t} \quad = \quad \begin{cases} \mathrm{t} & \textit{if} \quad \mathrm{s} \trianglelefteq \mathrm{t} \\ \mathrm{s} & \textit{if} \quad \mathrm{t} \trianglelefteq \mathrm{s} \\ \textit{the least expansion of } \ \mathrm{s}^- \vee \mathrm{t}^- & \\ \textit{that is also an upper bound for } \mathrm{s} \textit{ and } \mathrm{t} & \textit{otherwise.} \end{cases}$$

Proof. We must show that, whenever s and t are tree path-length sequences of length n, there are unique path-length sequences $\mathrm{s} \wedge \mathrm{t}$ and $\mathrm{s} \vee \mathrm{t}$ such that:

- $\mathrm{s} \wedge \mathrm{t} \trianglelefteq \mathrm{s},\ \mathrm{t}$; also if $\boldsymbol{\ell}$ is any path-length sequence, then $\boldsymbol{\ell} \trianglelefteq \mathrm{s},\ \mathrm{t}$ **iff** $\boldsymbol{\ell} \trianglelefteq \mathrm{s} \wedge \mathrm{t}$.
- $\mathrm{s},\ \mathrm{t} \trianglelefteq \mathrm{s} \vee \mathrm{t}$; also if $\boldsymbol{\ell}$ is any path-length sequence, then $\mathrm{s},\ \mathrm{t} \trianglelefteq \boldsymbol{\ell}$ **iff** $\mathrm{s} \vee \mathrm{t} \trianglelefteq \boldsymbol{\ell}$.

This can be done by induction on n. We consider only the glb here, the proof for the lub being similar. The theorem holds trivially for $n \leq 6$, since then the trees are totally ordered. Assume it holds for sequences of size $n-1$ or less.

First, s and t must have a common lower bound:
The glb $\mathrm{s}^- \wedge \mathrm{t}^-$ exists by induction, and (Theorems 3.9 and 3.10) lower expansion gives a lower bound

$$(\mathrm{s}^- \wedge \mathrm{t}^-)_+ \quad \trianglelefteq \quad (\mathrm{s}^-)_+ \quad \trianglelefteq \quad \mathrm{s}, \qquad\quad (\mathrm{s}^- \wedge \mathrm{t}^-)_+ \quad \trianglelefteq \quad (\mathrm{t}^-)_+ \quad \trianglelefteq \quad \mathrm{t}.$$

Second, if s and t have two greatest lower bounds $\boldsymbol{\ell}$ and $\boldsymbol{\ell}'$, then they must be equal:
From $\boldsymbol{\ell} \trianglelefteq \mathrm{s},\ \mathrm{t}$ and $\boldsymbol{\ell}' \trianglelefteq \mathrm{s},\ \mathrm{t}$ we infer $\boldsymbol{\ell}^- \trianglelefteq \mathrm{s}^- \wedge \mathrm{t}^-$ and $\boldsymbol{\ell}'^- \trianglelefteq \mathrm{s}^- \wedge \mathrm{t}^-$ by Theorem 3.10. Since furthermore $\boldsymbol{\ell}$ and $\boldsymbol{\ell}'$ are greatest lower bounds, $\mathrm{s}^- \wedge \mathrm{t}^- \trianglelefteq \boldsymbol{\ell}^-$ and $\mathrm{s}^- \wedge \mathrm{t}^- \trianglelefteq \boldsymbol{\ell}'^-$. Thus $\boldsymbol{\ell}^- = \boldsymbol{\ell}'^-$. By Theorem 3.9, the only way $\boldsymbol{\ell} \neq \boldsymbol{\ell}'$ can arise is that

$$\boldsymbol{\ell} \;=\; (\boldsymbol{\ell}^-)_+, \quad \boldsymbol{\ell}' \;=\; (\boldsymbol{\ell}^-)^+ \qquad \textbf{or} \qquad \boldsymbol{\ell} \;=\; (\boldsymbol{\ell}^-)^+, \quad \boldsymbol{\ell}' \;=\; (\boldsymbol{\ell}^-)_+$$

so $\boldsymbol{\ell} \trianglelefteq \boldsymbol{\ell}'$ or $\boldsymbol{\ell}' \trianglelefteq \boldsymbol{\ell}$, contradicting their both being greatest lower bounds. Thus $\boldsymbol{\ell} = \boldsymbol{\ell}'$.

Third, the algorithm produces a glb that is as good as any other lower bound:
Assuming this for $(\mathrm{s}^- \wedge \mathrm{t}^-)$ by induction, there can be no lower bound $\boldsymbol{\ell} \neq (\mathrm{s} \wedge \mathrm{t})$ such that $(\mathrm{s}^- \wedge \mathrm{t}^-)^+ \trianglelefteq \boldsymbol{\ell}$, since otherwise $(\mathrm{s}^- \wedge \mathrm{t}^-) \trianglelefteq \boldsymbol{\ell}^- \trianglelefteq \mathrm{s}^-,\ \mathrm{t}^-$ contradicting our assumption. □

The table of nontrivial examples in Figure 3.3 gives an appreciation for glbs and lubs. The first example (which is illustrated in the figure) is expanded in Table 3.2. Note the final pairs of entries in s and t are the same as the final pairs of entries in $\mathrm{s} \wedge \mathrm{t}$ and $\mathrm{s} \vee \mathrm{t}$, and the suffix lengths of s and t are never shorter than those of $\mathrm{s} \wedge \mathrm{t}$ and $\mathrm{s} \vee \mathrm{t}$.

THEOREM 3.19. *If* s *and* t *are path-length sequences of length* n*, then:*

$$\mathrm{s} \vee \mathrm{t} \quad = \quad \begin{cases} (\mathrm{s}^- \vee \mathrm{t}^-)^+ & \textit{if} \quad \mathrm{s} = (\mathrm{s}^-)^+ \quad \textit{and} \quad \mathrm{t} = (\mathrm{t}^-)^+ \\ (\mathrm{s}^- \vee \mathrm{t}^-)_+ & \textit{if} \quad \mathrm{s} = (\mathrm{s}^-)_+ \quad \textit{and} \quad \mathrm{t} = (\mathrm{t}^-)_+ \end{cases}$$

$$\mathrm{s} \wedge \mathrm{t} \quad = \quad \begin{cases} (\mathrm{s}^- \wedge \mathrm{t}^-)^+ & \textit{if} \quad \mathrm{s} = (\mathrm{s}^-)^+ \quad \textit{and} \quad \mathrm{t} = (\mathrm{t}^-)^+ \\ (\mathrm{s}^- \wedge \mathrm{t}^-)_+ & \textit{if} \quad \mathrm{s} = (\mathrm{s}^-)_+ \quad \textit{and} \quad \mathrm{t} = (\mathrm{t}^-)_+ \end{cases}$$

Otherwise, if either $\mathrm{s} = (\mathrm{s}^-)^+$ *and* $\mathrm{t} = (\mathrm{t}^-)_+$*, or* $\mathrm{s} = (\mathrm{s}^-)_+$ *and* $\mathrm{t} = (\mathrm{t}^-)^+$*, then either* $\mathrm{s} \wedge \mathrm{t} = (\mathrm{s}^- \wedge \mathrm{t}^-)^+$ *and* $\mathrm{s} \vee \mathrm{t} = (\mathrm{s}^- \vee \mathrm{t}^-)_+$*, or* $\mathrm{s} \wedge \mathrm{t} = (\mathrm{s}^- \wedge \mathrm{t}^-)_+$ *and* $\mathrm{s} \vee \mathrm{t} = (\mathrm{s}^- \vee \mathrm{t}^-)^+$.

Furthermore: if the final pairs of entries of s *and* t *are* $\langle p\ p \rangle$ *and* $\langle q\ q \rangle$*, where* $p \leq q$*, then the final pairs of entries of* $\mathrm{s} \wedge \mathrm{t}$ *and* $\mathrm{s} \vee \mathrm{t}$ *are respectively* $\langle p\ p \rangle$ *and* $\langle q\ q \rangle$.

Also: the suffix lengths of s *and* t *are at least as long as those of* $(\mathrm{s} \wedge \mathrm{t})$ *and* $(\mathrm{s} \vee \mathrm{t})$.

n	$\mathbf{s}$	$\mathbf{t}$	$\mathbf{s} \wedge \mathbf{t}$	$\mathbf{s} \vee \mathbf{t}$
9	$\langle\, 1\ 4\ 4\ 4\ 4\ 4\ 4\ 4\ 4 \,\rangle$	$\langle\, 2\ 2\ 2\ 3\ 5\ 5\ 5\ 6\ 6 \,\rangle$	$\langle\, 2\ 2\ 3\ 4\ 4\ 4\ 4\ 4\ 4 \,\rangle$	$\langle\, 1\ 3\ 3\ 3\ 5\ 5\ 5\ 6\ 6 \,\rangle$
	$\uparrow$ lower expansion	$\uparrow$ upper expansion	$\uparrow$ lower expansion	$\uparrow$ upper expansion
8	$\langle\, 1\ 3\ 4\ 4\ 4\ 4\ 4\ 4 \,\rangle$	$\langle\, 2\ 2\ 2\ 3\ 5\ 5\ 5\ 5 \,\rangle$	$\langle\, 2\ 2\ 3\ 3\ 4\ 4\ 4\ 4 \,\rangle$	$\langle\, 1\ 3\ 3\ 3\ 5\ 5\ 5\ 5 \,\rangle$
	$\uparrow$ lower expansion	$\uparrow$ lower expansion	$\uparrow$ lower expansion	$\uparrow$ lower expansion
7	$\langle\, 1\ 3\ 3\ 4\ 4\ 4\ 4 \,\rangle$	$\langle\, 2\ 2\ 2\ 3\ 4\ 5\ 5 \,\rangle$	$\langle\, 2\ 2\ 2\ 4\ 4\ 4\ 4 \,\rangle$	$\langle\, 1\ 3\ 3\ 3\ 4\ 5\ 5 \,\rangle$
	$\uparrow$ lower expansion	$\uparrow$ upper expansion	$\uparrow$ lower expansion	$\uparrow$ upper expansion
6	$\langle\, 1\ 3\ 3\ 3\ 4\ 4 \,\rangle$	$\langle\, 2\ 2\ 2\ 3\ 4\ 4 \,\rangle$	$\langle\, 2\ 2\ 2\ 3\ 4\ 4 \,\rangle$	$\langle\, 1\ 3\ 3\ 3\ 4\ 4 \,\rangle$

TABLE 3.2

Elaboration of the first example of representative bounds in Figure 3.3, showing how $\mathbf{s} \wedge \mathbf{t}$ *and* $\mathbf{s} \vee \mathbf{t}$ *can be derived with their recursive algorithms.*

Proof. These properties follow by induction on n. For the basis, they all hold trivially when $n \leq 6$, since then the imbalance lattice is a total order and $\{\, \mathbf{s},\ \mathbf{t} \,\} = \{\, \mathbf{s} \vee \mathbf{t},\ \mathbf{s} \wedge \mathbf{t} \,\}$, and the final two entries of any path-length sequence are a pair by Theorem 3.1. For the induction step, we can write

$$\begin{array}{rclcrcl} \mathbf{s} & = & \langle\ \cdots\ (p{-}i)\ \overbrace{p\ \ \cdot\ \ \cdot\ \ \cdot\ \ p}^{2h}\ \rangle & \qquad & \mathbf{t} & = & \langle\ \cdots\ (q{-}j)\ \overbrace{q\ \ \cdot\ \ \cdot\ \ \cdot\ \ q}^{2k}\ \rangle \\ \mathbf{s}^- & = & \langle\ \cdots\ (p{-}i)\ (p{-}1)\ \overbrace{p\ \cdots\ p}^{2h-2}\ \rangle & & \mathbf{t}^- & = & \langle\ \cdots\ (q{-}j)\ (q{-}1)\ \overbrace{q\ \cdots\ q}^{2k-2}\ \rangle \end{array}$$

where $i, j, h, k > 0$, and we assume with no loss of generality that $p \leq q$. There are four cases to consider, depending on the the suffix lengths $2h$ of $\mathbf{s}$ and $2k$ of $\mathbf{t}$. In the first, $h = 1$ and $k = 1$, i.e., $\mathbf{s} = (\mathbf{s}^-)^+$ and $\mathbf{t} = (\mathbf{t}^-)^+$. Then $i = 1$ and $j = 1$ by Theorem 3.1. By induction $(\mathbf{s}^- \wedge \mathbf{t}^-)$ and $(\mathbf{s}^- \vee \mathbf{t}^-)$ have respective final pairs $\langle (p{-}1)\ (p{-}1) \rangle$ and $\langle (q{-}1)\ (q{-}1) \rangle$, and have suffix lengths not exceeding those of $\mathbf{s}^-$ and $\mathbf{t}^-$. Now, by Theorem 3.10 $(\mathbf{s}^- \wedge \mathbf{t}^-)^+ \trianglelefteq (\mathbf{s}^-)^+$ and $(\mathbf{s}^- \wedge \mathbf{t}^-)^+ \trianglelefteq (\mathbf{t}^-)^+$. Because $(\mathbf{s}^-)^+ = \mathbf{s}$ and $(\mathbf{t}^-)^+ = \mathbf{t}$, the recursive algorithm in Theorem 3.18 will find $(\mathbf{s}^- \wedge \mathbf{t}^-)^+ \ = \ \mathbf{s} \wedge \mathbf{t}$. Thus the final pair of $\mathbf{s} \wedge \mathbf{t}$ will be $\langle p\ p \rangle$, and it will have suffix length 2. Similarly $\mathbf{s} \vee \mathbf{t} \ = \ (\mathbf{s}^- \vee \mathbf{t}^-)^+$ because $\mathbf{s} \vee \mathbf{t} \ \in \ \{\, (\mathbf{s}^- \vee \mathbf{t}^-)_+,\ (\mathbf{s}^- \vee \mathbf{t}^-)^+ \,\}$, and choosing $(\mathbf{s}^- \vee \mathbf{t}^-)_+$ gives a contradiction: if $\mathbf{s} \ = \ (\mathbf{s}^-)^+ \trianglelefteq (\mathbf{s}^- \vee \mathbf{t}^-)_+$ and $\mathbf{t} \ = \ (\mathbf{t}^-)^+ \trianglelefteq (\mathbf{s}^- \vee \mathbf{t}^-)_+$ then (because of Theorem 3.10) $\mathbf{s}^- \ = \ ((\mathbf{s}^-)^+)^- \trianglelefteq ((\mathbf{s}^- \vee \mathbf{t}^-)_+)^- \ \neq \ ((\mathbf{s}^- \vee \mathbf{t}^-)^+)^- \ = \ \mathbf{s}^- \vee \mathbf{t}^-$ and correspondingly $\mathbf{t}^- \ = \ ((\mathbf{t}^-)^+)^- \trianglelefteq ((\mathbf{s}^- \vee \mathbf{t}^-)_+)^- \ \neq \ ((\mathbf{s}^- \vee \mathbf{t}^-)^+)^- \ = \mathbf{s}^- \vee \mathbf{t}^-$ so the lub of $\mathbf{s}^-$ and $\mathbf{t}^-$ is not $\mathbf{s}^- \vee \mathbf{t}^-$, a contradiction. Again the final pair of $\mathbf{s} \vee \mathbf{t}$ will be $\langle q\ q \rangle$, with suffix length 2.

The other three cases, where $h > 1$ and/or $k > 1$, are similar. $\square$

4. Submodularity of Weighted Path-Length over the Lattices.

Huffman codes for a positive descending weight sequence $\mathbf{w} = \langle\, w_1\, w_2\, \cdots\, w_n \,\rangle$ are binary tree path-length sequences $\boldsymbol{\ell} = \langle\, \ell_1\, \ell_2\, \cdots\, \ell_n \,\rangle$ that minimize the weighted path-length

$$g_{\mathbf{w}}(\boldsymbol{\ell}) \quad = \quad \sum_{i=1}^{n} w_i\ \ell_i .$$

In this section we show that $g_{\mathbf{w}}$ is submodular over the lattice of trees. which helps explain why efficient algorithms for finding optimal trees are possible at all.

4.1. Submodularity.

Most work on submodular functions assumes the lattice is the lattice of subsets of a given set, the case originally emphasized by Edmonds [6]. However, the definition applies to any lattice:

DEFINITION 4.1. *A real-valued function* $f : \mathcal{L} \to \Re$ *defined on a lattice* $\langle \mathcal{L}, \sqsubseteq, \sqcap, \sqcup \rangle$ *is* **submodular** *if*

$$f(x\ \sqcap\ y)\ +\ f(x\ \sqcup\ y) \quad \leq \quad f(x)\ +\ f(y)$$

for all $x, y \in \mathcal{L}$. Equivalently, f is submodular if a "differential" inequality holds:

$$\Delta^2 f(x,y) \overset{def}{=} f(x) + f(y) - f(x \sqcap y) - f(x \sqcup y) \geq 0.$$

Section 4.4 discusses the relationship between submodularity and convexity.

4.2. Submodularity of weighted path-length on the Majorization lattice.

In this section we show that weighted path-length on the imbalance lattice of trees (or a logarithmic variant on the majorization lattice of densities) is a submodular function.

Define the function $G_{\mathbf{w}}$ on the majorization lattice of densities by

$$G_{\mathbf{w}}(\mathbf{x}) = g_{\mathbf{w}}(-\log_2(\mathbf{x})) = -\sum_i w_i \log_2(x_i).$$

Note that $G_{\mathbf{w}}$ is convex on $\Re_+{}^n$, as its Hessian $\nabla^2 G_{\mathbf{w}} = (\frac{\partial^2 G_{\mathbf{w}}(\mathbf{x})}{\partial x_i \partial x_j}) = \frac{1}{\ln(2)} \operatorname{diag}(\frac{w_i}{x_i^2})$ is positive semidefinite there [27, p.448]. (recall we are assuming all weights are positive).

$G_{\mathbf{w}}$ is actually also submodular on the majorization lattice. We prove this directly now, and show later how submodularity can be established using only vector calculus.

THEOREM 4.2. *Assuming $\mathbf{w}$ is a descending positive sequence of length n, $G_{\mathbf{w}}$ is submodular on the majorization lattice. That is, for all nonnegative sequences $\mathbf{x}$, $\mathbf{y}$ of length n,*

$$G_{\mathbf{w}}(\mathbf{x} \sqcap \mathbf{y}) + G_{\mathbf{w}}(\mathbf{x} \sqcup \mathbf{y}) \leq G_{\mathbf{w}}(\mathbf{x}) + G_{\mathbf{w}}(\mathbf{y}).$$

Proof. By induction on n. For $n = 1$, the inequality is satisfied with equality. Let a_n and b_n be the n-th entries of $(\mathbf{x} \sqcap \mathbf{y})$ and $(\mathbf{x} \sqcup \mathbf{y})$, respectively. The theorem follows by induction if we can show

$$w_n \cdot (-\log_2(a_n)) + w_n \cdot (-\log_2(b_n)) \leq w_n \cdot (-\log_2(x_n)) + w_n \cdot (-\log_2(y_n)).$$

Recall that $\mathbf{x} \sqcap \mathbf{y} = \boldsymbol{\partial}((\int \mathbf{x}) \, \mathbf{min}_{vec} (\int \mathbf{y}))$ and $\mathbf{x} \sqcup \mathbf{y} = \boldsymbol{\partial}((\int \mathbf{x}) \, \mathbf{max}_{vec} (\int \mathbf{y}))$. There are four cases, depending on $\mathbf{X} = \int \mathbf{x}$ and $\mathbf{Y} = \int \mathbf{y}$, and specifically on the final values

$$X_{n-1} = \sum_{i=1}^{n-1} x_i, \qquad X_n = \sum_{i=1}^{n} x_i, \qquad Y_{n-1} = \sum_{i=1}^{n-1} y_i, \qquad Y_n = \sum_{i=1}^{n} y_i$$

as follows:

1. if $X_{n-1} \leq Y_{n-1}$ and $X_n \leq Y_n$, then $a_n = x_n$, $b_n = y_n$.
2. if $X_{n-1} \geq Y_{n-1}$ and $X_n \geq Y_n$, then $a_n = y_n$, $b_n = x_n$.
3. if $X_{n-1} \leq Y_{n-1}$ and $X_n \geq Y_n$, then $x_n \geq y_n$, $a_n = Y_n - X_{n-1} = y_n + \epsilon$, $b_n = X_n - Y_{n-1} = x_n - \epsilon$, where $\epsilon = (Y_{n-1} - X_{n-1}) \geq 0$ and $\epsilon \leq x_n - y_n = (x_n \max y_n) - (x_n \min y_n)$.
4. if $X_{n-1} \geq Y_{n-1}$ and $X_n \leq Y_n$, then $y_n \geq x_n$, $a_n = X_n - Y_{n-1} = x_n + \epsilon$, $b_n = Y_n - X_{n-1} = y_n - \epsilon$, where $\epsilon = (X_{n-1} - Y_{n-1}) \geq 0$ and $\epsilon \leq y_n - x_n = (x_n \max y_n) - (x_n \min y_n)$.

Each case satisfies $w_n \cdot (-\log_2(a_n)) + w_n \cdot (-\log_2(b_n)) \leq w_n \cdot (-\log_2(x_n)) + w_n \cdot (-\log_2(y_n))$ as needed; the first two cases satisfy it with equality, and in the last two we have

$$a_n = (x_n \min y_n) + \epsilon, \qquad b_n = (x_n \max y_n) - \epsilon$$

but then assuming $x_n, y_n \geq 0$,

$$\log_2(a_n) + \log_2(b_n) = \log_2(a_n \, b_n) = \log_2(x_n \, y_n + \eta) \geq \log_2(x_n) + \log_2(y_n)$$

where $\eta = \epsilon \, ((x_n \max y_n) - (x_n \min y_n) - \epsilon) \geq 0$ and multiplying by $-w_n$ gives the theorem.
□

4.3. Submodularity of weighted path-length on the Imbalance lattice.

THEOREM 4.3. *Assuming* $\mathbf{w}$ *is a descending positive sequence of length* n*,* $g_{\mathbf{w}}$ *is submodular on the imbalance lattice. That is, for all path-length sequences* s*,* $\mathbf{t}$ *of length* n*,*

$$g_{\mathbf{w}}(\mathrm{s} \wedge \mathbf{t}) \;+\; g_{\mathbf{w}}(\mathrm{s} \vee \mathbf{t}) \quad \leq \quad g_{\mathbf{w}}(\mathrm{s}) \;+\; g_{\mathbf{w}}(\mathbf{t}).$$

Proof. Also by induction on n. The theorem holds with equality for $n \leq 6$, since then the lattice of path-length sequences is totally ordered. We sketch the induction step from $n-1$ to n, showing $\Delta^2 g_{\mathbf{w}}(\mathrm{s}, \mathbf{t}) \;=\; (g_{\mathbf{w}}(\mathrm{s}) \;+\; g_{\mathbf{w}}(\mathbf{t})) \;-\; (g_{\mathbf{w}}(\mathrm{s} \wedge \mathbf{t}) \;+\; g_{\mathbf{w}}(\mathrm{s} \vee \mathbf{t})) \;\geq\; 0$ follows from $\Delta^2 g_{\mathbf{w}}(\mathrm{s}^-, \mathbf{t}^-) \geq 0$ — where $g_{\mathbf{w}}$, when applied to sequences of length $(n-1)$, uses only the first $(n-1)$ entries of $\mathbf{w}$.

Recall $2k$ is the suffix length of the path-length sequence $\boldsymbol{\ell} = \langle\ \cdots\ (q-j)\ \overbrace{q\ \cdots\ q}^{2k}\ \rangle$, and j is its suffix increment. The suffix increment is 1 when $(\boldsymbol{\ell}^-)^+ = \boldsymbol{\ell}$, so:

$$g_{\mathbf{w}}(\boldsymbol{\ell}) \quad = \quad \begin{cases} g_{\mathbf{w}}(\boldsymbol{\ell}^-) \;+\; (w_{n-1} \quad\; + \; q \cdot w_n) & \text{if } \boldsymbol{\ell} = (\boldsymbol{\ell}^-)^+ \quad \text{(i.e., } k = 1\text{)} \;+\; 0 \\ g_{\mathbf{w}}(\boldsymbol{\ell}^-) \;+\; (w_{n-2k+1} \;+\; q \cdot w_n) & \text{if } \boldsymbol{\ell} = (\boldsymbol{\ell}^-)_+ \quad \text{(i.e., } k > 1\text{).} \end{cases}$$

Thus $g_{\mathbf{w}}(\mathrm{s}) > g_{\mathbf{w}}(\mathrm{s}^-)$ and $g_{\mathbf{w}}(\mathbf{t}) > g_{\mathbf{w}}(\mathbf{t}^-)$ in all cases.

However, it can happen that $g_{\mathbf{w}}(\mathrm{s} \wedge \mathbf{t}) < g_{\mathbf{w}}(\mathrm{s}^- \wedge \mathbf{t}^-)$ or $g_{\mathbf{w}}(\mathrm{s} \vee \mathbf{t}) < g_{\mathbf{w}}(\mathrm{s}^- \vee \mathbf{t}^-)$ because it is possible either that $\mathrm{s}^- \wedge \mathbf{t}^- \;\neq\; (\mathrm{s} \wedge \mathbf{t})^-$ or that $\mathrm{s}^- \vee \mathbf{t}^- \;\neq\; (\mathrm{s} \vee \mathbf{t})^-$. Specifically, it is possible $\mathrm{s} \wedge \mathbf{t} \;=\; \langle\ \cdots\ (q-j)\ (q-j)\ \overbrace{q\ \cdots\ q}^{2k}\ \rangle$ and $\mathrm{s}^- \wedge \mathbf{t}^- \;=\; \langle\ \cdots\ (q-j-1)\ \overbrace{q\ \cdots\ q}^{2k}\ \rangle$, i.e., $\mathrm{s} \wedge \mathbf{t} = (\mathrm{s}^- \wedge \mathbf{t}^-)_+$ and $\mathrm{s} \wedge \mathbf{t}$ has suffix increment $j > 1$, in which case

$$g_{\mathbf{w}}(\mathrm{s} \wedge \mathbf{t}) \quad = \quad g_{\mathbf{w}}(\mathrm{s}^- \wedge \mathbf{t}^-) \;+\; (w_{n-2k+1} \;-\; j \cdot w_{n-2k} \;+\; q \cdot w_n)$$

and the parenthesized expression can be negative.

From Theorem 3.19, the final pairs of entries of s and $\mathbf{t}$ are always the same as the final pairs of entries of $\mathrm{s} \wedge \mathbf{t}$ and $\mathrm{s} \vee \mathbf{t}$, and the suffix lengths for each of s and $\mathbf{t}$ cannot be less than those for each of $(\mathrm{s} \wedge \mathbf{t})$ and $(\mathrm{s} \vee \mathbf{t})$. We now consider the same four cases addressed in the proof of Theorem 3.19.

In the case where both s is the upper expansion of s^-, and $\mathbf{t}$ is the upper expansion of $\mathbf{t}^-$, then by Theorem 3.19 $\mathrm{s} \wedge \mathbf{t} = (\mathrm{s}^- \wedge \mathbf{t}^-)^+$ and $\mathrm{s} \vee \mathbf{t} = (\mathrm{s}^- \vee \mathbf{t}^-)^+$, so

$$\begin{aligned} \Delta^2 g_{\mathbf{w}}(\mathrm{s}, \mathbf{t}) \quad &= \quad (g_{\mathbf{w}}(\mathrm{s}) \;+\; g_{\mathbf{w}}(\mathbf{t})) \;-\; (g_{\mathbf{w}}(\mathrm{s} \wedge \mathbf{t}) \;+\; g_{\mathbf{w}}(\mathrm{s} \vee \mathbf{t})) \\ &= \quad (g_{\mathbf{w}}(\mathrm{s}^-) + g_{\mathbf{w}}(\mathbf{t}^-)) \;-\; (g_{\mathbf{w}}(\mathrm{s}^- \wedge \mathbf{t}^-) + g_{\mathbf{w}}(\mathrm{s}^- \vee \mathbf{t}^-)) \quad = \quad \Delta^2 g_{\mathbf{w}}(\mathrm{s}^-, \mathbf{t}^-) \end{aligned}$$

with the preceding analysis for $g_{\mathbf{w}}(\boldsymbol{\ell})$ with $k = 1$. In this situation only the final pairs of entries of s, $\mathbf{t}$ and of $\mathrm{s} \wedge \mathbf{t}$, $\mathrm{s} \vee \mathbf{t}$ can cause the two differences to be unequal, but we now know them to give the same two pairs. So in this case the theorem follows by induction.

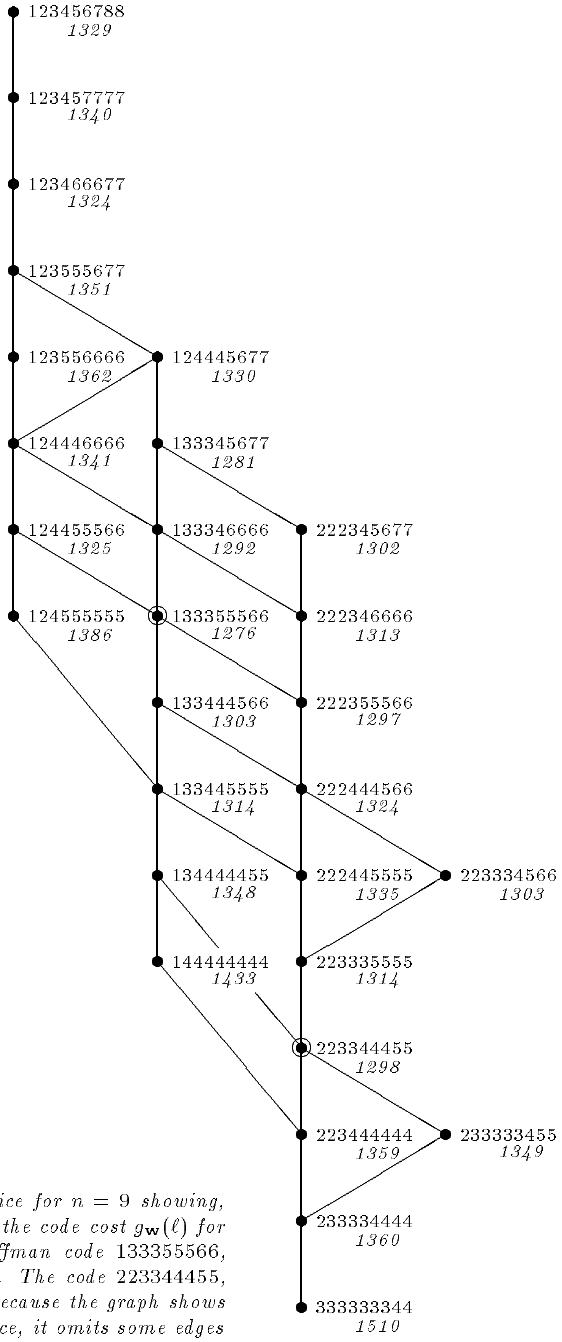
It remains to treat the cases where s is the lower expansion of s^- or $\mathbf{t}$ is the lower expansion of $\mathbf{t}^-$. In these cases it can happen that $g_{\mathbf{w}}(\mathrm{s} \wedge \mathbf{t}) < g_{\mathbf{w}}(\mathrm{s}^- \wedge \mathbf{t}^-)$ or $g_{\mathbf{w}}(\mathrm{s} \vee \mathbf{t}) < g_{\mathbf{w}}(\mathrm{s}^- \vee \mathbf{t}^-)$ as previously noted.

In the case where either s is the lower expansion of s^- or $\mathbf{t}$ is the lower expansion of $\mathbf{t}^-$, but not both, then by Theorem 3.19, either $\mathrm{s} \wedge \mathbf{t} \;=\; (\mathrm{s}^- \wedge \mathbf{t}^-)_+$ and $\mathrm{s} \vee \mathbf{t} \;=\; (\mathrm{s}^- \vee \mathbf{t}^-)^+$, or $\mathrm{s} \wedge \mathbf{t} \;=\; (\mathrm{s}^- \wedge \mathbf{t}^-)^+$ and $\mathrm{s} \vee \mathbf{t} \;=\; (\mathrm{s}^- \vee \mathbf{t}^-)_+$. The lower expansions among these two cannot yield as large a $g_{\mathbf{w}}$ increase as the lower expansions giving s and $\mathbf{t}$, because they expand higher-indexed positions (their suffix lengths are never longer), and the suffix increment of $\mathrm{s}^- \wedge \mathbf{t}^-$ or $\mathrm{s}^- \vee \mathbf{t}^-$ can be greater than 1. Therefore $\Delta^2 g_{\mathbf{w}}(\mathrm{s}, \mathbf{t}) \;\geq\; \Delta^2 g_{\mathbf{w}}(\mathrm{s}^-, \mathbf{t}^-)$.

In the final case where both s is the lower expansion of s^- and $\mathbf{t}$ is the lower expansion of $\mathbf{t}^-$, then $\mathrm{s} \wedge \mathbf{t} \;=\; (\mathrm{s}^- \wedge \mathbf{t}^-)_+$ and $\mathrm{s} \vee \mathbf{t} \;=\; (\mathrm{s}^- \vee \mathbf{t}^-)_+$ (Theorem 3.19), and because the lower expansions giving $\mathrm{s} \wedge \mathbf{t}$ and $\mathrm{s} \vee \mathbf{t}$ cannot yield as large a $g_{\mathbf{w}}$ increase as the lower expansions giving s and $\mathbf{t}$, again $\Delta^2 g_{\mathbf{w}}(\mathrm{s}, \mathbf{t}) \;\geq\; \Delta^2 g_{\mathbf{w}}(\mathrm{s}^-, \mathbf{t}^-)$. □

To see an example, the submodularity of $g_{\mathbf{w}}$ can be verified on the lattice for $n = 9$ and the weight sequence shown in Figure 4.1.

Code	*Cost*
Path-length Sequence ℓ	*Weighted Path-length* $g_{\mathbf{w}}(\ell)$
123456788	*1329*
123457777	*1340*
123466677	*1324*
123555677	*1351*
123556666	*1362*
124445677	*1330*
124446666	*1341*
124455566	*1325*
124555555	*1386*
133345677	*1281*
133346666	*1292*
133355566	$\boxed{\mathit{1276}}$
133444566	*1303*
133445555	*1314*
134444455	*1348*
144444444	*1433*
222345677	*1302*
222346666	*1313*
222355566	*1297*
222444566	*1324*
222445555	*1335*
223334566	*1303*
223335555	*1314*
223344455	*1298*
223444444	*1359*
233333455	*1349*
233334444	*1360*
333333344	*1510*



The transitively reduced imbalance lattice for $n = 9$ *showing, for* $\mathbf{w} = \langle$ 189 95 73 71 23 21 18 9 1 $\rangle$*, the code cost* $g_{\mathbf{w}}(\ell)$ *for each path-length sequence* ℓ*. The Huffman code* 133355566*, with cost 1276, is the global minimum. The code* 223344455*, with cost 1298, is a local minimum. (Because the graph shows only the transitive reduction of the lattice, it omits some edges corresponding to exchanges, but the minimum is localized.)*

FIG. 4.1. *Costs of all possible codes for the weights* $\mathbf{w} = \langle$ 189 95 73 71 23 21 18 9 1 $\rangle$.

4.4. Submodularity as a discrete analogue of Convexity. Although it is very simply defined, submodularity is difficult to appreciate. Using only standard vector calculus, we now clarify some basic relationships between submodularity and notions of convexity. We have not seen this done elsewhere.

There are several reasons why submodularity plays an important role here, at the crossroads between information and coding theory. First, submodularity is directly related to the Fortuin-Kasteleyn-Ginibre (FKG) "correlation" inequalities, which generalize a basic inequality of Tchebycheff on mean values of functions (hence expected values of random variables). A fine survey of results with FKG-like inequalities is [15].

Second, submodularity is closely related to convexity. Book-length surveys by Fujishige [9] and Narayanan [28] review connections between submodularity and optimization (and even electrical network theory). The relationship between convexity and submodularity was neatly summarized by Lovász with the following memorable definition and result.

Definition 4.4. *Given a finite set S of cardinality n, we can identify a* **$\{0,1\}$-vector** $\mathbf{t} \in \Re_+{}^n$ *with any subset $T \subseteq S$ specifying the incidence in T of the elements in S (indexed in some fixed order).*

Any nonnegative vector $\mathbf{x} \in \Re_+{}^n$ can be decomposed uniquely into a sum of positive real values multiplied by 'decreasing' $\{0,1\}$-vectors. Specifically, $\mathbf{x} \in \Re_+{}^n$ determines an integer k $(0 \le k \le n)$ such that $\mathbf{x}$ has a unique **greedy decomposition**

$$\mathbf{x} \quad = \quad \sum_{i=1}^{k} \lambda_i \; \mathbf{s}_i$$

where $\lambda_i > 0$, $S_1 \supset \cdots \supset S_k$ are distinct subsets of S, and $\mathbf{s}_i$ is the $\{0,1\}$-vector identified with S_i. For any function $f : S \rightarrow \Re_+$, its **greedy extension** $\widehat{f} : \Re_+{}^n \rightarrow \Re_+$ *to nonnegative vectors is then defined by*

$$\widehat{f}(\mathbf{x}) \quad = \quad \widehat{f}(\sum_{i=1}^{k} \lambda_i \; \mathbf{s}_i \;) \quad = \quad \sum_{i=1}^{k} \lambda_i \; f(S_i).$$

In fact, $\boldsymbol{\lambda} = \boldsymbol{\partial}\,(\mathrm{sort}\!\uparrow(\,\mathbf{x}\,))$, using our notation.

Theorem 4.5. *(Lovász [25, p.249])*
$f : S \rightarrow \Re_+$ is submodular iff its greedy extension $\widehat{f} : \Re_+{}^n \rightarrow \Re_+$ is convex.

Proof. The essence is that for positive constants $\lambda \le \kappa$ and sets $T \ne U$,

$$\widehat{f}(\lambda\,\mathbf{t} + \kappa\,\mathbf{u}) \;\; = \;\; \lambda\,f(T \cup U) \; + \; (\kappa - \lambda)\,f(U) \;\; \le \;\; \lambda\,f(T) \; + \; \kappa\,f(U) \;\; = \;\; \widehat{f}(\lambda\,\mathbf{t}) \; + \; \widehat{f}(\kappa\,\mathbf{u})$$

where $\mathbf{t}$ and $\mathbf{u}$ are the $\{0,1\}$-vectors corresponding to T and U. The central inequality is due to submodularity. Restricting $0 < \lambda < 1$ and $\kappa = (1-\lambda)$ shows $\widehat{f}$ is convex. □

Lovász goes on [25, p.250–251] to point out that

$$\min \; \{ \; f(X) \mid X \;\subseteq\; S \; \} \quad = \quad \min \; \{ \; \widehat{f}(\mathbf{x}) \mid \mathbf{x} \;\in\; [0,1]^n \; \}$$

and that as a consequence there is a polynomial-time algorithm to minimize f.

The vector lattice $\langle \Re_+{}^n, \le_{vec}, \mathbf{min}_{vec}, \mathbf{max}_{vec} \rangle$, is exactly the extension of the set lattice to nonnegative vectors. Vector lattices, also called **Riesz spaces**, can be more 'natural' than set lattices in some ways. For example, submodularity has a natural characterization:

Theorem 4.6. *(Lorentz [27, p.150])*
When twice differentiable, f is submodular on the vector lattice $\langle \Re_+{}^n, \le_{vec}, \mathbf{min}_{vec}, \mathbf{max}_{vec} \rangle$ iff

$$\frac{\partial^2 f}{\partial x_i \partial x_j} \quad \le \quad 0 \qquad\qquad (i \ne j, \;\; 1 \le i, \, j \le n).$$

Proof. Essentially definition. Using the shorthand $f\langle\ u\ v\ \rangle$ to denote the expression $f(\langle\ x_1\ \ldots\ x_{i-1}\ \ u\ \ x_{i+1}\ \ldots\ x_{j-1}\ \ v\ \ x_{j+1}\ \ldots\ x_n\ \rangle)$ gives the result

$$\frac{\partial^2 f}{\partial x_i \partial x_j} \;=\; \lim_{\epsilon_i,\epsilon_j\to 0} \frac{f\langle\,(x_i{+}\epsilon_i)\ (x_j{+}\epsilon_j)\,\rangle - f\langle\,(x_i{+}\epsilon_i)\ x_j\,\rangle - f\langle\,x_i\ (x_j{+}\epsilon_j)\,\rangle + f\langle\,x_i\ x_j\,\rangle}{\epsilon_i\ \ \epsilon_j} \;\le\; 0$$

where the inequality comes from the fact that f is submodular, since with respect to $\le_{vec}$, the points $\mathbf{x} = \langle\ (x_i{+}\epsilon_i)\ x_j\ \rangle$ and $\mathbf{y} = \langle\ x_i\ (x_j{+}\epsilon_j)\ \rangle$ have the upper and lower bounds $\mathbf{x}\ \mathbf{max}_{vec}\ \mathbf{y} = \langle\ (x_i{+}\epsilon_i)\ (x_j{+}\epsilon_j)\ \rangle$ and $\mathbf{x}\ \mathbf{min}_{vec}\ \mathbf{y} = \langle\ x_i\ x_j\ \rangle$. For the converse, if f is not submodular on a rectangle defined by $\mathbf{x} = \langle\ (x_i{+}a)\ x_j\ \rangle$ and $\mathbf{y} = \langle\ x_i\ (x_j{+}b)\ \rangle$ Lorentz pointed out we can find a subrectangle on which $\partial^2 f/\partial x_i \partial x_j > 0$. □

Note: the derivatives $\frac{\partial^2 f}{\partial x_i^2}$ can still be positive. In fact, the Hessian $\nabla^2 f = (\frac{\partial^2 f}{\partial x_i \partial x_j})$ still can even be positive semidefinite (hence f can be convex), or be an **M-matrix** [3, ch.6].

THEOREM 4.7. *When twice differentiable, F is submodular on the majorization lattice $\langle \Re_+{}^n, \preceq, \sqcap, \sqcup\rangle$ iff for all $i \ne j$ between 1 and $n-1$,*

$$\frac{\partial^2 F(\mathbf{z})}{\partial z_i \partial z_j} \;-\; \frac{\partial^2 F(\mathbf{z})}{\partial z_{i+1} \partial z_j} \;-\; \frac{\partial^2 F(\mathbf{z})}{\partial z_i \partial z_{j+1}} \;+\; \frac{\partial^2 F(\mathbf{z})}{\partial z_{i+1} \partial z_{j+1}} \;\;\le\;\; 0.$$

Proof. Theorem 3.16 shows that the Möbius transformation $\boldsymbol{\partial}$ gives a bijection between the majorization lattice $\langle \Re_+{}^n, \preceq, \sqcap, \sqcup\rangle$ and the vector lattice $\langle \Re_+{}^n, \le_{vec}, \mathbf{min}_{vec}, \mathbf{max}_{vec}\rangle$. Thus $f(\mathbf{x}) = F(\boldsymbol{\partial}\mathbf{x})$ is submodular on the vector lattice when F is submodular on the majorization lattice. Expanding the inequality

$$\frac{\partial^2}{\partial x_i \partial x_j}(F(\boldsymbol{\partial}\mathbf{x})) \quad=\quad \frac{\partial^2}{\partial x_i \partial x_j}(f(\mathbf{x})) \quad\le\quad 0$$

(which follows from the previous theorem) with the chain rule gives the stated result, because $\mathbf{z} = \boldsymbol{\partial}\mathbf{x} = \langle\ x_1\ (x_2{-}x_1)\ (x_3{-}x_2)\ \cdots\ (x_n{-}x_{n-1})\ \rangle$. □

Revisiting Theorem 4.2, $G_{\mathbf{w}}(\mathbf{z}) \;=\; -\sum_{i=1}^n\ w_i\ \log_2(z_i)$ satisfies

$$\frac{\partial^2 G_{\mathbf{w}}(\mathbf{z})}{\partial z_i \partial z_j} - \frac{\partial^2 G_{\mathbf{w}}(\mathbf{z})}{\partial z_{i+1} \partial z_j} - \frac{\partial^2 G_{\mathbf{w}}(\mathbf{z})}{\partial z_i \partial z_{j+1}} + \frac{\partial^2 G_{\mathbf{w}}(\mathbf{z})}{\partial z_{i+1} \partial z_{j+1}} \;=\; \frac{1}{\ln(2)} \begin{cases} 0 & |i-j|>1 \\ -w_i/z_i^2 & i\ =\ j+1 \\ -w_{i+1}/z_{i+1}^2 & j\ =\ i+1 \\ w_{i+1}/z_{i+1}^2 + w_i/z_i^2 & i\ =\ j \end{cases}$$

and, for example, $G_{\mathbf{w}}(\boldsymbol{\partial}\mathbf{x}) \;=\; -\,w_1\ \log_2(x_1) \;-\; \sum_{i=1}^{n-1}\ w_i\ \log_2(x_{i+1}-x_i)$ satisfies

$$\frac{\partial^2}{\partial x_i \partial x_j}(G_{\mathbf{w}}(\boldsymbol{\partial}\mathbf{x})) \quad=\quad \frac{1}{\ln(2)}\ \frac{-w_{i+1}}{(x_{i+1}-x_i)^2} \quad\le\quad 0 \qquad\quad \text{when } j = i+1.$$

This gives two alternative proofs of Theorem 4.2, showing how such results can be derived more easily.

DEFINITION 4.8. *A function $F : \Re_+{}^n \to \Re$ is* **Schur convex** *if it preserves the majorization ordering, i.e., $\mathbf{x} \preceq \mathbf{y}$ implies $F(\mathbf{x}) \le F(\mathbf{y})$.*

THEOREM 4.9. *F is Schur convex on the majorization lattice iff $f(\mathbf{x}) = F(\boldsymbol{\partial}\mathbf{x})$ is monotone on the vector lattice.*

Proof. Again a direct result of the bijection between the two lattices. If f is differentiable, f is monotone on the vector lattice iff $\nabla f(\mathbf{x}) = \langle\ \partial f/\partial x_1 \cdots \partial f/\partial x_n\ \rangle \ge_{vec} \mathbf{0}$, which implies

$$\frac{\partial F}{\partial x_i}(\boldsymbol{\partial}\mathbf{x}) \;\;-\;\; \frac{\partial F}{\partial x_{i+1}}(\boldsymbol{\partial}\mathbf{x}) \quad=\quad \frac{\partial}{\partial x_i}(F(\boldsymbol{\partial}\mathbf{x})) \quad=\quad \frac{\partial}{\partial x_i} f(\mathbf{x}) \quad\ge\quad 0 \qquad (1 \le i \le n-1).$$

This rederives the result that $\partial F/\partial z_i \ge \partial F/\partial z_{i+1}$ when F is Schur convex [34]. □

Since monotonicity and convexity are related, Theorems 4.5–4.9 connect convexity, submodularity, Schur convexity, and majorization. There are actually many connections. See the survey [27, ch.6], in which submodular functions are called **$\mathcal{L}$-subadditive functions**. Just as Lovász showed for submodular functions [25], Schur convex functions [37, 29] are closed under various operations: min, max, convolution, composition with convex functions, etc. [27, ch.3]. Theorem 4.5 is also reminiscent of **symmetric gauge** functions, which are Schur convex; see [27, p.96].

5. Huffman Coding as Submodular Dynamic Programming.

The results of the previous sections can now be applied to Huffman coding.

5.1. Non-monotonicity of Weighted Path-Length over the Lattices.

It is important to realize that weighted path-length is not monotone on the imbalance lattice, so greedy search may not always find its way to an optimal solution. This is illustrated by the example in Figure 4.1. For this problem the sequence $\langle\, 2\,2\,3\,3\,4\,4\,4\,5\,5\,\rangle$ with cost 1298 is a local minimum: each of the 7 sequences reachable from it by imbalancing exchanges, and each of the 3 sequences reachable from it by balancing exchanges, have greater weighted path length. The diagram shows only the transitive reduction of the imbalance lattice, omitting many balancing exchanges (because they would clutter the picture), but it conveys the general situation for larger Huffman coding problems. It shows that, even though it may do very well in practice, simple hill-climbing along ternary exchanges is not guaranteed to find the optimum sequence.

Although weighted path length $g_{\mathbf{w}}$ is not monotone on the imbalance lattice of trees, a monotone summary of weighted path length ${g_{\mathbf{w}}}^{mon}$ has the properties we need.

In [25, p.241], Lovász stated the following definition and theorem for set lattices (easily proved for general lattices) about the 'monotonization' of a function f:

Definition 5.1. *If $f : \mathcal{L} \to \Re_+$ is a real-valued function on a lattice $\mathcal{L}$ with ordering relation $\sqsubseteq$, define*

$$f^{mon}(\mathbf{x}) \quad = \quad \min\ \{\ f(\mathbf{x}') \mid \mathbf{x}' \sqsubseteq \mathbf{x}\ \}.$$

Theorem 5.2. *If f is submodular, then f^{mon} is also submodular.*

Proof. From the definition of f^{mon}, for all $\mathbf{x}$, $\mathbf{y}$ in $\mathcal{L}$, there exists a $\mathbf{x}' \sqsubseteq \mathbf{x}$ such that $f^{mon}(\mathbf{x}) = f(\mathbf{x}')$ and a $\mathbf{y}' \sqsubseteq \mathbf{y}$ such that $f^{mon}(\mathbf{y}) = f(\mathbf{y}')$. But then $(\mathbf{x}' \sqcap \mathbf{y}') \sqsubseteq (\mathbf{x} \sqcap \mathbf{y})$ and $(\mathbf{x}' \sqcup \mathbf{y}') \sqsubseteq (\mathbf{x} \sqcup \mathbf{y})$, so

$$\begin{array}{rcccl} f^{mon}(\mathbf{x} \sqcap \mathbf{y}) & \leq & f^{mon}(\mathbf{x}' \sqcap \mathbf{y}') & \leq & f(\mathbf{x}' \sqcap \mathbf{y}'), \\ f^{mon}(\mathbf{x} \sqcup \mathbf{y}) & \leq & f^{mon}(\mathbf{x}' \sqcup \mathbf{y}') & \leq & f(\mathbf{x}' \sqcup \mathbf{y}'). \end{array}$$

$$\begin{array}{rcll} f^{mon}(\mathbf{x} \sqcap \mathbf{y}) \ + \ f^{mon}(\mathbf{x} \sqcup \mathbf{y}) & \leq & f(\mathbf{x}' \sqcap \mathbf{y}') \ + \ f(\mathbf{x}' \sqcup \mathbf{y}') & \\ & \leq & f(\mathbf{x}') \ + \ f(\mathbf{y}') & \text{(as } f \text{ is submodular)} \\ & = & f^{mon}(\mathbf{x}) \ + \ f^{mon}(\mathbf{y}). & \end{array}$$

□

Thus ${g_{\mathbf{w}}}^{mon}$ is both submodular and monotone on the tree imbalance lattice.

5.2. Dynamic Programming reconstruction of the Huffman Algorithm.

Based the analysis above, we can derive Huffman codes using a simple recursion.

Definition 5.3. *The* **Huffman contraction $\mathbf{w}^{\frown}$** *of a descending weight sequence $\mathbf{w} = \langle\, w_1 \cdots w_n \,\rangle$ is*

$$\mathbf{w}^{\frown} \quad = \quad \mathrm{sort}{\downarrow}(\,\langle\ w_1\ \cdots\ w_{n-2}\ (w_{n-1}{+}w_n)\ \rangle\,).$$

Parenthetically, note that $\boldsymbol{\ell}^{-} \ = \ -\log_2(\,(2^{-\boldsymbol{\ell}})^{\frown}\,)$ for path-length sequences $\boldsymbol{\ell}$.

If $n \geq 1$ is the length of $\mathbf{w}$*, then the (most balanced)* **Huffman code** *for* $\mathbf{w}$ *is defined by*

$$\mathit{Huffman}(\mathbf{w}) \quad = \quad \begin{cases} \langle\ 0\ \rangle & \textit{if}\ \ n = 1 \\ \mathit{better\_expansion}(\,\mathit{Huffman}(\,\mathbf{w}^{\frown}\,)\,,\mathbf{w}) & \textit{if}\ \ n > 1 \end{cases}$$

$$\mathit{better\_expansion}(\,\boldsymbol{\ell}\,,\mathbf{w}) \quad = \quad \begin{cases} \boldsymbol{\ell}_+ & \textit{if}\ \ g_{\mathbf{w}}(\,\boldsymbol{\ell}_+\,) \ \leq\ g_{\mathbf{w}}(\,\boldsymbol{\ell}^+\,) \\ \boldsymbol{\ell}^+ & \textit{otherwise.} \end{cases}$$

For example, the example in Figure 4.1 can be traced through Figure 3.3 and Table 2.2:

$$\begin{array}{lcl}
\mathit{Huffman}(\ \langle\ 189\ 95\ 73\ 71\ 23\ 21\ 18\ 9\ 1\ \rangle\ ) & = & \langle\ 1\ 3\ 3\ 3\ 5\ 5\ 5\ 6\ 6\ \rangle \\
\mathit{Huffman}(\ \langle\ 189\ 95\ 73\ 71\ 23\ 21\ 18\ 10\ \rangle\ ) & = & \langle\ 1\ 3\ 3\ 3\ 5\ 5\ 5\ 5\ \rangle \\
\mathit{Huffman}(\ \langle\ 189\ 95\ 73\ 71\ 28\ 23\ 21\ \rangle\ ) & = & \langle\ 1\ 3\ 3\ 3\ 4\ 5\ 5\ \rangle \\
\mathit{Huffman}(\ \langle\ 189\ 95\ 73\ 71\ 44\ 28\ \rangle\ ) & = & \langle\ 1\ 3\ 3\ 3\ 4\ 4\ \rangle \\
\mathit{Huffman}(\ \langle\ 189\ 95\ 73\ 72\ 71\ \rangle\ ) & = & \langle\ 1\ 3\ 3\ 3\ 3\ \rangle \\
\mathit{Huffman}(\ \langle\ 189\ 143\ 95\ 73\ \rangle\ ) & = & \langle\ 1\ 2\ 3\ 3\ \rangle \\
\mathit{Huffman}(\ \langle\ 189\ 168\ 143\ \rangle\ ) & = & \langle\ 1\ 2\ 2\ \rangle \\
\mathit{Huffman}(\ \langle\ 311\ 189\ \rangle\ ) & = & \langle\ 1\ 1\ \rangle \\
\mathit{Huffman}(\ \langle\ 500\ \rangle\ ) & = & \langle\ 0\ \rangle.
\end{array}$$

This dynamic programming definition is similar to the standard Huffman algorithm, but differs in a few ways. First, it considers only upper and lower expansions of *Huffman*$(\,\mathbf{w}^{\frown}\,)$; we prove momentarily that this is sufficient. Second, it produces a *unique* Huffman code, which is the most balanced possible because it uses lower expansions whenever possible. (This avoids the issue of multiplicity of solutions that arises in implementation of the standard Huffman algorithm. For example, if $\mathbf{w} = \langle\ 3\ 2\ 2\ 1\ \rangle$ then $\langle\ 1\ 2\ 3\ 3\ \rangle$ is a Huffman code, but the more balanced sequence $\langle\ 2\ 2\ 2\ 2\ \rangle$ is also and will be produced by the algorithm above.)

Actually, the definition of *Huffman*$(\mathbf{w})$ can be "simplified". Note that when

$$\boldsymbol{\ell} \quad = \quad \mathit{Huffman}(\,\mathbf{w}^{\frown}\,) \quad = \quad \langle\ \cdots\ (q{-}j)\ \overbrace{q\ \cdots\ q}^{2k}\ \rangle$$

with suffix length $2k$ and suffix increment j, the condition on *better_expansion*$(\boldsymbol{\ell}, \mathbf{w})$ is:

$$\begin{array}{lrcl}
 & g_{\mathbf{w}}(\boldsymbol{\ell}_+) & \leq & g_{\mathbf{w}}(\boldsymbol{\ell}^+) \\
\Leftrightarrow & g_{\mathbf{w}}(\boldsymbol{\ell})\ +\ w_{n-2k-1}\ -\ (j{-}1)\,w_{n-2k}\ +\ q\,w_n & \leq & g_{\mathbf{w}}(\boldsymbol{\ell})\ +\ w_{n-1}\ +\ (q+1)\,w_n \\
\Leftrightarrow & w_{n-2k-1}\ -\ (j{-}1)\,w_{n-2k} & \leq & w_{n-1}\ +\ w_n.
\end{array}$$

Going further, the proof of Theorem 5.5 below implies that this condition actually can be simplified to take $j = 1$, so that *better_expansion*$(\boldsymbol{\ell}, \mathbf{w}) = \boldsymbol{\ell}_+$ if $w_{n-2k-1} \leq w_{n-1} + w_n$.

THEOREM 5.4. *Huffman*$(\mathbf{w})$ *is the most balanced optimal code for* $\mathbf{w}$*.*

Proof. Let $\mathrm{s} = \mathit{Huffman}(\mathbf{w})$, so s is the cheaper of $\mathit{Huffman}(\mathbf{w}^{\frown})_+$ and $\mathit{Huffman}(\mathbf{w}^{\frown})^+$, or is the former (which is more balanced) if they have equal cost.

Only these two expansions need be considered. Like the usual Huffman algorithm, this algorithm assigns w_{n-1} and w_n maximal path length. Therefore $(w_{n-1} + w_n)$ must appear in the "suffix" of $\mathbf{w}^{\frown}$, i.e., among the $2k + 1$ final entries, where $2k$ is the suffix length of $\boldsymbol{\ell} = \mathit{Huffman}(\mathbf{w}^{\frown})$; and so it has path length either $(q - 1)$ or q, corresponding to the two possible expansions. Thus only Huffman codes are derived with the algorithm above.

Submodularity of ${g_{\mathbf{w}}}^{mon}$ now proves that there is a unique most balanced Huffman code (and thus greedy search will find this code). Suppose that s and $\mathbf{t}$ are maximally balanced Huffman codes that are not comparable in the balance ordering. Then ${g_{\mathbf{w}}}^{mon}(\mathbf{t}) = g_{\mathbf{w}}(\mathbf{t}) = g_{\mathbf{w}}(\mathrm{s})$. Because $\mathbf{t}$ is optimal ${g_{\mathbf{w}}}^{mon}(\mathrm{s} \vee \mathbf{t}) = {g_{\mathbf{w}}}^{mon}(\mathbf{t})$. Submodularity of ${g_{\mathbf{w}}}^{mon}$ then implies that ${g_{\mathbf{w}}}^{mon}(\mathrm{s} \wedge \mathbf{t}) \leq {g_{\mathbf{w}}}^{mon}(\mathrm{s})$. By the definition of ${g_{\mathbf{w}}}^{mon}$, ${g_{\mathbf{w}}}^{mon}(\mathrm{s} \wedge \mathbf{t}) = {g_{\mathbf{w}}}^{mon}(\mathrm{s})$. But then $\mathrm{s} \wedge \mathbf{t}$ is optimal, hence a Huffman code, and it is more balanced than both s and $\mathbf{t}$. This gives a contradiction. □

THEOREM 5.5. $\mathit{Huffman}(\mathbf{w}^{\frown}) \;=\; \mathit{Huffman}(\mathbf{w})^{-}$.

Proof. We prove this by induction on the length n of $\mathbf{w}$. The base case $n = 2$ is trivial.

For the induction step, let $\mathrm{s} = \mathit{Huffman}(\mathbf{w})$, so by the previous theorem s is the most balanced optimal code for $\mathbf{w}$. Let $\boldsymbol{\ell} = \mathit{Huffman}(\mathbf{w}^{\frown})$. As $\mathbf{w}^{\frown}$ has length $(n-1)$, $\boldsymbol{\ell}$ is the most balanced optimal code for $\mathbf{w}^{\frown}$ by the induction hypothesis. By definition s is the better (cheaper or more balanced if equally cheap) of $\boldsymbol{\ell}^{+}$ or $\boldsymbol{\ell}_{+}$. We consider these two possibilities.

First, if $\mathrm{s} = \boldsymbol{\ell}^{+}$, then $\boldsymbol{\ell} = \mathrm{s}^{-}$ as required, because $\boldsymbol{\ell} = (\boldsymbol{\ell}^{+})^{-}$ by Theorem 3.9.

Second, if $\mathrm{s} = \boldsymbol{\ell}_{+}$, then $w_{n-2k-1} - (j-1)\,w_{n-2k} \;\le\; w_{n-1} + w_n$, where $2k$ is the suffix length of $\boldsymbol{\ell}$, j is its suffix increment, and by Theorem 3.1, $j \le \log_2(2k)$ or equivalently $2^{j-1} \le k$. If $j = 1$ we find again $\boldsymbol{\ell} = \mathrm{s}^{-}$ as required.

We now claim that $j > 1$ cannot arise in this second possibility where $\mathrm{s} = \boldsymbol{\ell}_{+}$. Let us first understand intuitively why this is so. When the suffix length $j > 1$, $\boldsymbol{\ell} = \langle\ \cdots\ (q-j)\ q\ \cdots\ q\ \rangle$ describes a tree that is perfectly balanced over its suffix, but the rest is at least j levels shorter. The Huffman algorithm will construct such a tree only when the final $2k$ weights of $\mathbf{w}^{\frown}$ are all of similar size, but w_{n-2k-1} is much larger. Specifically, $w_{n-1} + w_n < w_{n-2k-1}$, and w_{n-2k-1} is constrained to be larger than the sum of the 2^{j-1} subsequent weights, or the Huffman algorithm would construct a different tree. But w_{n-2k-1} is also constrained to be small by the inequality in the definition of the Huffman algorithm; if it becomes too large, we get $\mathrm{s} = \boldsymbol{\ell}^{+}$ instead of $\mathrm{s} = \boldsymbol{\ell}_{+}$. These two constraints turn out to not be simultaneously satisfiable when $j > 1$.

Supposing that $j > 1$ and $\mathrm{s} = \boldsymbol{\ell}_{+}$, then, since $w_{n-1} + w_n < w_{n-2k-1}$ we have two cases:

1. $\underline{w_{n-3} \ge w_{n-1} + w_n.}$
   Let $W = w_{n-3}$. Then $w_{n-2k-1} > w_{n-2k} + (2^{j-1} - 1)\,W$ and $2W \ge w_{n-2k} \ge W$, since $j > 1$ implies $2k \ge 4$ by Theorem 3.1, $\ell_{n-2k-1} = (q-j)$, and $\boldsymbol{\ell} = \mathit{Huffman}(\mathbf{w}^{\frown})$ is constructed by the Huffman algorithm. These give the first bound
   $$\frac{w_{n-2k-1}}{w_{n-2k}} \quad > \quad 1 \;+\; (2^{j-1} - 1)/2 \quad = \quad 2^{j-2} + 1/2.$$
   However, from $w_{n-2k-1} - (j-1)\,w_{n-2k} \;\le\; w_{n-1} + w_n$ it follows that
   $$\begin{array}{rcl} w_{n-2k-1} & > & w_{n-2k} \;+\; (2^{j-1} - 1)\,W \\ & \ge & w_{n-2k} \;+\; (2^{j-1} - 1)\,(w_{n-1} + w_n) \\ & \ge & w_{n-2k} \;+\; (2^{j-1} - 1)\,(w_{n-2k-1} - (j-1)\,w_{n-2k}). \end{array}$$
   When $j = 2$, this simplifies to $w_{n-2k-1} > w_{n-2k-1}$, a contradiction. When $j > 2$, it gives the second bound
   $$\frac{w_{n-2k-1}}{w_{n-2k}} \quad < \quad \frac{(j-1) \;-\; 1/(2^{j-1} - 1)}{1 \;-\; 1/(2^{j-1} - 1)}.$$
   However this contradicts the first bound for all $j > 2$.
2. $\underline{w_{n-2k-1} > w_{n-1} + w_n > w_{n-3}.}$
   Like the previous case, but this time when $W = w_{n-3}$ we can derive only the weaker condition $W > (w_{n-1} + w_n)/2$, because $w_{n-3} \ge w_{n-1} \ge w_n$. Still, the same first bound, essentially the second bound (with $(2^{j-1} - 1)$ divided by 2), and the same contradictions, are derivable.

Thus all cases reach a contradiction, implying as required that, in the second possibility, $j = 1$ and $\boldsymbol{\ell} = \mathrm{s}^{-}$. □

5.3. The importance of submodularity in Dynamic Programming. Together, Theorems 5.4 and 5.5 show that Huffman coding (finding the most balanced code that minimizes $g_{\mathbf{w}}{}^{mon}$) is a dynamic programming problem that can be solved in various ways, because the problem enjoys elegant recursive properties.

Huffman coding gives another example of a dynamic programming problem that can be sped up considerably because the objective function is submodular over the solution space. Lawler [24] remarked:

> *If a discrete optimization problem can be solved efficiently, it is quite likely that submodularity is responsible. In recent years there has been a growing appreciation of the fact that submodularity plays a pivotal role in discrete optimization, not unlike that of convexity in continuous optimization.*

Submodularity has a long history in dynamic programming. By 1781, Monge had found a form of submodularity to be important in simplifying the transportation problem [17]. In 1970, Edmonds [6] related submodularity to matroids and greedily-solvable optimization problems. In 1980, Yao [42] generalized upon Knuth's famous $O(n^2)$ algorithm for optimum binary search trees [21] by giving an $O(n^2)$ algorithm for the dynamic programming problem

$$\begin{array}{rcl} c(i,i) & = & 0 \\ c(i,j) & = & w(i,j) \;+\; \min_{\,i<k\leq j}\,(\,c(i,k-1) + c(k,j)\,) \quad (i<j) \end{array}$$

$$\begin{array}{rcccccl} & & w(i,j) & \leq & w(i',j') & & (i' \leq i \leq j \leq j') \\ w(i,j) & + & w(i',j') & \leq & w(i',j) & + \; w(i,j') & (i \leq i' \leq j \leq j'). \end{array}$$

Yao called the final constraint the **quadrangle inequality**, noting that it implies the (inverse) triangle inequality. Writing $I = [i,j]$ and $J = [i',j']$, defining a lattice of intervals of indices in the dynamic programming array, these two constraints require the function $W([a,b]) = -w(a,b)$ to be *monotone decreasing* ($W(I) \geq W(J)$ if $I \subseteq J$) and *submodular* ($W(I) + W(J) \geq W(I \cap J) + W(I \cup J)$). Results from exploiting the quadrangle inequality in dynamic programming appear in [2, 7, 35] for problems ranging from DNA sequencing to minimum cost matching.

Mirroring Theorem 4.7, the **Monge condition** $w_{i,j} + w_{i+1,j+1} \leq w_{i+1,j} + w_{i,j+1}$ on an $n \times n$ weight matrix W is also equivalent to the requirement that, ignoring its first column and row, the matrix $\boldsymbol{\partial}\ W\ \boldsymbol{\partial}^{\top}$ is nonpositive. Burkard et al. have also compiled a comprehensive survey of many incarnations of the Monge condition in [4].

Recently Klein explored the connection between dynamic programming and submodularity [20]. Also Golin and Rote [14] developed dynamic programming algorithms for prefix codes when the codeword letters have differing costs, a useful case not handled by Huffman's algorithm; and they recently extended this work to exploit the Monge property.

6. Other Applications.

The results here also can be used to gain further insight about submodular dynamic programming, the Huffman coding problem, and perhaps also about the applications of lattice concepts in coding. Almost all of the theorems proved here admit interesting extensions and/or special cases. For example, a direct corollary of Theorem 4.3 (using $\mathbf{w} = \langle\, 1 \cdots 1 \,\rangle$) is that the function mapping a path-length sequence to its level of balance is submodular on the imbalance lattice. It would be interesting to extend the work here for the t-ary codes discussed in [19].

Majorization, we believe, can be exploited further in characterizing optimal codes. We have established that the imbalance ordering on tree path-length sequences $\boldsymbol{\ell}$ is isomorphic to the majorization ordering on exponentiated tree path-length sequences $\mathbf{x} = 2^{-\boldsymbol{\ell}}$. Thus any function that is Schur convex (i.e., 'majorization-preserving': monotone with respect to the majorization ordering) on exponentiated path-length sequences and hence monotone on the (continuous) majorization lattice, will also be monotone on the (discrete) imbalance lattice. Negative entropy is an important example of such a function; related functions are discussed in [32].

Furthermore, the methods developed above hold out hope for entirely new approaches to Huffman coding. We sketch two possibilities.

6.1. Continuous Approximation of Huffman Codes.

One possibility is that we can attack the combinatorial problem of Huffman coding with a continuous, real-valued optimization problem. Recall Huffman coding can be expressed as an optimization problem:

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{n} \; w_i \; \ell_i \\ \text{subject to} & \sum_{i=1}^{n} \; 2^{-\ell_i} \;=\; 1, \qquad \ell_i > 0, \;\; \text{integer} \;\; (1 \leq i \leq n). \end{array}$$

Dropping the integrality constraint gives an interesting continuous relaxation of Huffman coding that can be attacked numerically. For example, by treating the constraint as a penalty function, the problem above can be solved numerically with something like the system of equations $\partial/\partial\ell_j \left(\sum_{i=1}^{n} w_i\, \ell_i \;+\; 10^{10}\,(1 \;-\; \sum_{i=1}^{n} 2^{-\ell_i}\,)^2 \right) \;=\; 0 \;\; (1 \leq j \leq n)$. Using the example weight sequence $\mathbf{w} \;=\; \langle\ 189\ 95\ 73\ 71\ 23\ 21\ 18\ 9\ 1\ \rangle$ studied earlier, a simple program found a unique real solution $\boldsymbol{\ell} \approx \langle\ 1.4\ 2.4\ 2.8\ 2.8\ 4.4\ 4.8\ 4.8\ 5.8\ 9.0\ \rangle$ for these equations, with objective ≈ 1241. As expected, this solution is near the optimal Huffman code $\langle\, 1\,3\,3\,3\,5\,5\,5\,6\,6\, \rangle$, with cost 1276.

When the relaxation is faithful to the original, it will be possible to find optimal solutions quickly. The relaxed solution can be used to jump to the right neighborhood in the imbalance lattice, from which balancing exchanges will walk to the optimal code. The penalty function could clearly be varied, and perhaps could be changed to encourage near-integral solutions.

Interior point methods on the majorization lattice may also be possible. Among other things, it may be possible to define $\mathbf{s} \wedge \mathbf{t}$ in terms of $-\!\log_2 \left(2^{-\mathbf{s}} \sqcap 2^{-\mathbf{t}} \right)$ and define $\mathbf{s} \vee \mathbf{t}$ in terms of $-\!\log_2 \left(2^{-\mathbf{s}} \sqcup 2^{-\mathbf{t}} \right)$: they are often identical, and always satisfy

$$2^{-\mathbf{s}\,\wedge\,\mathbf{t}} \;\; \preceq \;\; 2^{-\mathbf{s}} \sqcap 2^{-\mathbf{t}}, \qquad\qquad 2^{-\mathbf{s}} \sqcup 2^{-\mathbf{t}} \;\; \preceq \;\; 2^{-\mathbf{s}\,\vee\,\mathbf{t}}$$

(because $2^{-\mathbf{s}} \sqcap 2^{-\mathbf{t}}$ and $2^{-\mathbf{s}} \sqcup 2^{-\mathbf{t}}$ are the glb and lub with respect to majorization).

For perspective, if $\alpha \;=\; 7 - \log_2(12) \;\approx\; 3.4150375$ and $\beta = (\alpha - 1)$, the following set of examples represent the unusual cases with $n = 9$ where $-\!\log_2 \left(2^{-\mathbf{s}} \;\sqcap\; 2^{-\mathbf{t}} \right) \neq (\mathbf{s} \wedge \mathbf{t})$ or $-\!\log_2 \left(2^{-\mathbf{s}} \;\sqcup\; 2^{-\mathbf{t}} \right) \neq (\mathbf{s} \vee \mathbf{t})$:

$\mathbf{s}$	$\mathbf{t}$	$-\!\log_2 \left(2^{-\mathbf{s}} \;\sqcap\; 2^{-\mathbf{t}} \right)$	$\mathbf{s} \wedge \mathbf{t}$
$\langle\ 1\ 2\ 4\ 5\ 5\ 5\ 5\ 5\ 5\ \rangle$	$\langle\ 1\ 3\ 3\ 3\ 4\ 5\ 6\ 7\ 7\ \rangle$	$\langle\ 1\ 3\ 3\ \alpha\ 5\ 5\ 5\ 5\ 5\ \rangle$	$\langle\ 1\ 3\ 3\ 4\ 4\ 5\ 5\ 5\ 5\ \rangle$
$\langle\ 1\ 2\ 4\ 5\ 5\ 5\ 5\ 5\ 5\ \rangle$	$\langle\ 2\ 2\ 2\ 3\ 4\ 5\ 6\ 7\ 7\ \rangle$	$\langle\ 2\ 2\ 2\ \alpha\ 5\ 5\ 5\ 5\ 5\ \rangle$	$\langle\ 2\ 2\ 2\ 4\ 4\ 5\ 5\ 5\ 5\ \rangle$
$\langle\ 1\ 3\ 4\ 4\ 4\ 4\ 4\ 5\ 5\ \rangle$	$\langle\ 2\ 2\ 2\ 3\ 4\ 5\ 6\ 7\ 7\ \rangle$	$\langle\ 2\ 2\ \beta\ 4\ 4\ 4\ 4\ 5\ 5\ \rangle$	$\langle\ 2\ 2\ 3\ 3\ 4\ 4\ 4\ 5\ 5\ \rangle$

$\mathbf{s}$	$\mathbf{t}$	$-\!\log_2 \left(2^{-\mathbf{s}} \;\sqcup\; 2^{-\mathbf{t}} \right)$	$\mathbf{s} \vee \mathbf{t}$
$\langle\ 1\ 4\ 4\ 4\ 4\ 4\ 4\ 4\ 4\ \rangle$	$\langle\ 2\ 2\ 2\ 3\ 4\ 5\ 6\ 7\ 7\ \rangle$	$\langle\ 1\ 4\ \beta\ 3\ 4\ 5\ 6\ 7\ 7\ \rangle$	$\langle\ 1\ 3\ 3\ 3\ 4\ 5\ 6\ 7\ 7\ \rangle$
$\langle\ 1\ 4\ 4\ 4\ 4\ 4\ 4\ 4\ 4\ \rangle$	$\langle\ 2\ 2\ 2\ 3\ 4\ 6\ 6\ 6\ 6\ \rangle$	$\langle\ 1\ 4\ \beta\ 3\ 4\ 6\ 6\ 6\ 6\ \rangle$	$\langle\ 1\ 3\ 3\ 3\ 4\ 6\ 6\ 6\ 6\ \rangle$
$\langle\ 1\ 4\ 4\ 4\ 4\ 4\ 4\ 4\ 4\ \rangle$	$\langle\ 2\ 2\ 2\ 3\ 5\ 5\ 5\ 6\ 6\ \rangle$	$\langle\ 1\ 4\ \beta\ 3\ 5\ 5\ 5\ 6\ 6\ \rangle$	$\langle\ 1\ 3\ 3\ 3\ 5\ 5\ 5\ 6\ 6\ \rangle$
$\langle\ 1\ 4\ 4\ 4\ 4\ 4\ 4\ 4\ 4\ \rangle$	$\langle\ 2\ 2\ 2\ 4\ 4\ 4\ 5\ 6\ 6\ \rangle$	$\langle\ 1\ 4\ \beta\ 4\ 4\ 4\ 5\ 6\ 6\ \rangle$	$\langle\ 1\ 3\ 3\ 4\ 4\ 4\ 5\ 6\ 6\ \rangle$
$\langle\ 1\ 4\ 4\ 4\ 4\ 4\ 4\ 4\ 4\ \rangle$	$\langle\ 2\ 2\ 2\ 4\ 4\ 5\ 5\ 5\ 5\ \rangle$	$\langle\ 1\ 4\ \beta\ 4\ 4\ 5\ 5\ 5\ 5\ \rangle$	$\langle\ 1\ 3\ 3\ 4\ 4\ 5\ 5\ 5\ 5\ \rangle$

These examples suggest there may be algorithms that 'round up' $-\!\log_2 \left(2^{-\mathbf{s}} \;\sqcap\; 2^{-\mathbf{t}} \right)$ to give $\mathbf{s} \wedge \mathbf{t}$, and 'round down' $-\!\log_2 \left(2^{-\mathbf{s}} \;\sqcup\; 2^{-\mathbf{t}} \right)$ to give $\mathbf{s} \vee \mathbf{t}$.

6.2. Practical Applications in Adaptive Coding. In many practical situations it is difficult or impossible to know *a priori* the weights $\mathbf{w}$ used in Huffman coding. A natural idea, which occurred independently to Faller [8] and Gallager [11], is to allow the weights to be determined dynamically, and have the Huffman code 'evolve' over time. **Dynamic Huffman coding** is the strategy of repeatedly constructing the Huffman code for the input so far, and using it in transmitting the next input symbol. Knuth presented an efficient algorithm for dynamic Huffman coding in [22], and his performance results for the algorithm show it consistently producing compression very near (though not surpassing) the compression attained with static Huffman code for the entire input.

Vitter [40, 41] then developed a dynamic Huffman algorithm that improves on Knuth's in the following way: rather than simply revise the Huffman tree after each input symbol, Vitter also finds a new Huffman tree of minimal external path length $\sum_i \ell_i$ and height $\max_i \ell_i$. With this modification Vitter was actually able to surpass the performance of static Huffman coding on several benchmarks.

A small contribution we can make is to clarify the improvement of Vitter. Basically, Vitter's algorithm differs from Knuth's in *constructing the optimal path-length sequence that is also as balanced as possible.* Note that minimizing the external path length $\sum_i \ell_i$ is identical to maximizing the level of balance. Since there can be more than one optimal code, and unnecessary imbalance tends to penalize the symbol currently being encoded, insisting on maximally balanced codes improves performance.

Another contribution of the lattice perspective here is to encourage development of new adaptive coding schemes. As suggested in Section 5.1, a move between adjacent points in the lattice corresponds to minor alteration of codes, and by moving through the lattice we incrementally modify the cost of a code. Hill-climbing then gives greedy coding algorithms, and on-line hill-climbing gives adaptive coding algorithms. Although we have shown that the codes produced by hill-climbing are not guaranteed to be optimal, lattice-oriented adaptive coding algorithms may still have a role to play in some coding situations, since the Huffman notion of optimality is not really what is needed in the (currently popular and enormously important) adaptive context.

For example, adaptive coding algorithms can start at any point in the lattice, as long as both ends of the communication know which one. Rather than rely on the dynamic Huffman algorithm to derive reasonable operating points for the code, or rely on Knuth's 'windowed' algorithm [22], one can immediately begin with a mutually-agreed-upon, 'reasonable' initial code (depending on the type of information being transmitted), and then adapt this code using some mutually-agreed-upon greedy algorithm for moving in the imbalance lattice.

Acknowledgements. We are very grateful to Pierre Hasenfratz for insightful comments that improved this paper. A conversation with Mordecai Golin, who provided us with an expanded version of [14], inspired us to discuss dynamic programming explicitly in this paper. He also pointed out the survey [4] to us. Also, we are indebted to two anonymous referees for clarifications of the exposition, especially of the significance of submodularity and of Shannon's work [38].

REFERENCES

[1] J. Abrahams, *Code and Parse Trees for Lossless Source Encoding*, Proc. Compression & Complexity of Sequences (SEQUENCES'97), Positano, Italy, 1997, IEEE Press, to appear.

[2] A. Aggarwal, A. Bar-Noy, S. Khuller, D. Kravets, B. Schieber, *Efficient Minimum Cost Matching and Transportation using the Quadrangle Inequality*, J. Algorithms, 19:1 (July 1995), pp. 116–143.

[3] A. Berman, R.J. Plemmons, *Nonnegative matrices in the mathematical sciences*, SIAM, 1994.

[4] R.E. Burkard, B. Klinz, R. Rudolf, *Perspectives of Monge properties in optimization*, Discrete Applied Math., 70 (1996), pp. 95–161.

[5] B.A. Davey, H.A. Priestley, *Introduction to Lattices and Order*, Cambridge U. Press, 1990.

[6] J. Edmonds, *Submodular Functions, Matroids and Certain Polyhedra*, in Combinatorial Structures and their Applications, R. Guy et al., eds., Gordon & Breach, 1970, pp. 69–87.

[7] D. Eppstein, Z. Galil, R. Giancarlo, G.F. Italiano, *Sparse dynamic Programming. II. Convex and concave cost functions*, Journal of the ACM, 39:3 (July 1992), pp. 546–567.

[8] N. Faller, *An adaptive system for data compression*, Pacific Grove, CA: Record of the 7th Asilomar Conference on Circuits, Systems, and Computers, 1973, pp. 593–597.

[9] S. Fujishige, *Submodular functions and optimization*, North-Holland Elsevier, 1991.

[10] R.G. Gallager, *Information Theory and Reliable Communications*, J. Wiley, 1968.

[11] R.G. Gallager, *Variations on a theme by Huffman*, IEEE Trans. Information Theory, IT-24:6 (November 1978), pp. 668–674.

[12] E.N. Gilbert, *Codes Based on Inaccurate Source Probabilities*, IEEE Trans. Information Theory, IT-17:3 (May 1971), pp. 304–314. $g(N)$ is analyzed on p. 309.

[13] C.R. Glassey and R.M. Karp, *On the optimality of Huffman trees*, SIAM J. Appl. Math., 31 (1976), pp. 368–378.

[14] M.J. Golin, G. Rote, *A dynamic programming algorithm for constructing optimal prefix-free codes for unequal letter costs,* Proc. ICALP 95, Z. Fulop, F. Gecseg, eds., Springer-Verlag, 1995, pp. 256–267.

[15] R.L. GRAHAM, *Applications of the FKG Inequality and its Relatives*, in Mathematical Programming: The State of the Art, B. Korte, A. Bachem, M. Grötschel, eds., Springer-Verlag, 1983, pp. 115–131.

[16] G.H. HARDY, J.E. LITTLEWOOD, G. POLYA, *Inequalities*, Cambridge University Press, 1934.

[17] A.J. HOFFMAN. *On Simple Linear Programming Problems*, in Convexity, Proc. Seventh Symposium in Pure Mathematics, Vol. VII, V. Klee, ed., AMS, 1961, pp. 317–327.

[18] D.A. HUFFMAN, *A method for the construction of minimum redundancy codes*, Proc. IRE, 40 (1951), pp. 1098–1101.

[19] F.K. HWANG, *Generalized Huffman Trees*, SIAM J. Appl. Math., 37 (1979), pp. 124–127.

[20] C.M. KLEIN, *A submodular approach to discrete dynamic programming*, European J. Operational Research, 80:1 (Jan. 1995), pp. 145–155.

[21] D.E. KNUTH, *Optimum Binary Search Trees*, Acta Informatica, 1 (1971), pp. 14–25.

[22] D.E. KNUTH, *Dynamic Huffman Coding*, J. Algorithms, 6 (1985), pp. 163–180.

[23] E. LAWLER, *Combinatorial Optimization: Networks & Matroids*, Holt-Rinehart-Winston, 1976.

[24] E.L. LAWLER, *Submodular Functions & Polymatroid Optimization*, in Combinatorial Optimization: Annotated Bibliographies, A.H.G. Rinnooy Kan, M. O'hEigeartaigh, J.K. Lenstra, eds., J. Wiley & Sons, 1985, pp. 32–38.

[25] L. LOVÁSZ, *Submodular functions and convexity*, in Mathematical Programming: The State of the Art, B. Korte, A. Bachem, M. Grötschel, eds., Springer-Verlag, 1983, pp. 235–257.

[26] U. MANBER, *Introduction to Algorithms*, Addison-Wesley, 1989.

[27] A.W. MARSHALL, I. OLKIN, *Inequalities: Theory of Majorization and Its Applications*, Academic Press, 1979.

[28] H. NARAYANAN, *Submodular functions and electrical networks*, North-Holland Elsevier, 1997.

[29] A. OSTROWSKI, *Sur quelques applications des fonctions convexes et concaves au sens de I. Schur (offert en homage à P. Montel)*, J. Math. Pures Appl., 31 (1952), pp. 253–292.

[30] J.M. PALLO, *Enumerating, ranking and unranking binary trees*, Computer Journal, 29:2 (April 1986), pp. 171–175.

[31] J.M. PALLO, *Some properties of the rotation lattice of binary trees,* Computer Journal, 31:6 (Dec. 1988), pp. 564–565.

[32] D.S. PARKER, *Conditions for Optimality of the Huffman Algorithm*, SIAM J. Comput., 9:3 (August 1980), pp. 470–489.

[33] D.S. PARKER, P. RAM, *Greed and Majorization*, November 1994. Issued as Technical Report CSD-960003, UCLA Computer Science Dept., March 1996.

[34] D.S. PARKER, P. RAM, *A Linear Algebraic Reconstruction of Majorization*, Technical Report CSD-970036, UCLA Computer Science Dept., September 1997.

[35] U. PFERSCHY, R. RUDOLF, G.J. WOEGINGER, *Monge matrices make maximization manageable*, Operations Research Letters, 16 (1994), pp. 245–254.

[36] G.-C. ROTA, *On the Foundations of Combinatory Theory I. Theory of Möbius Functions,* Z. Wahrscheinlichkeitstheorie, 2 (1964), pp. 340–368.

[37] I. SCHUR, *Über eine Klasse von Mittelbildungen mit Anwendungen auf die Determinantentheorie*, Sitzungsber. Berl. Math. Ges., 22 (1923), pp. 9–20.

[38] C.E. SHANNON, *The Lattice Theory of Information*, Proc. IRE Trans. Information Theory, 1 (1950). Reprinted in *Claude Elwood Shannon: Collected Papers*, IEEE Press, 1993.

[39] N.J.A. SLOANE, S. PLOUFFE, *The Encyclopedia of Integer Sequences*, Academic Press, 1995.

[40] J.S. VITTER, *Design and Analysis of Dynamic Huffman Codes*, J. ACM, 34:4 (October 1987), pp. 825–845.

[41] J.S. VITTER, *Algorithm 673: Dynamic Huffman Coding*, ACM TOMS, 15:2 (June 1989), pp. 158–167.

[42] F.F. YAO, *Efficient dynamic programming using quadrangle inequalities*, Los Angeles, CA: Proc. 12th Annual ACM Symp. on Theory of Computing, 1980, pp. 429–435.