2-1981

Fast Sorting of Weyl Sequences Using Comparisons

Martin. H. Ellis

John M. Steele
University of Pennsylvania

Recommended Citation

Ellis, M. H., & Steele, J. M. (1981). Fast Sorting of Weyl Sequences Using Comparisons. *SIAM Journal on Computing, 10* (1), 88-95. http://dx.doi.org/10.1137/0210007

Fast Sorting of Weyl Sequences Using Comparisons

Abstract

An algorithm is given which makes only $O(\log n)$ comparisons, and which will determine the ordering of the uniformly distributed (pseudo random) Weyl sequences given by $\{ (k\alpha)\bmod 1:1 \leqq k \leqq n\} $, where $\alpha $ is an unspecified irrational number. This result is shown to be best possible in the sense that no algorithm can perform the same task with fewer than $ \Omega (\log n)$ comparisons.

Disciplines

Other Mathematics

SIAM J. COMPUT.
Vol. 10, No. 1, February 1981

0097-5397/81/1001-0007 $01.00/0

FAST SORTING OF WEYL SEQUENCES USING COMPARISONS*

MARTIN H. ELLIS† AND J. MICHAEL STEELE‡

Abstract. An algorithm is given which makes only $O(\log n)$ comparisons, and which will determine the ordering of the uniformly distributed (pseudo random) Weyl sequences given by $\{(k\alpha) \bmod 1 : 1 \leqq k \leqq n\}$, where α is an unspecified irrational number. This result is shown to be best possible in the sense that no algorithm can perform the same task with fewer than $\Omega(\log n)$ comparisons.

Key words. sorting, Weyl sequences, information theory lower bound, alpha-sort

1. Introduction[1]. Any algorithm which sorts sets of n real numbers only on the basis of comparisons will always require, in the worst case, at least $\log_2 (n!) = \theta(n \log_2 n)$ comparisons. Similarly, if n reals are chosen at random from any continuous distribution, the expected number of comparisons required for sorting them is also $\theta(n \log_2 n)$. These familiar facts may make it surprising that there are sequences which share many properties with random sequences, but whose order can always be determined with fewer than $4 \log_2 n$ comparisons.

The sequences considered here are the so called Weyl sequences given by $X_k = k\alpha \bmod 1$, where α is an irrational number. These sequences share with the *independent* uniformly distributed random variables the basic property that the number of elements from $X_1, X_2, \cdots, X_n$ in (a, b) is asymptotic to $n(b-a)$, for $0 \leqq a < b \leqq 1$. (For a purely probabilistic proof of this property, see Feller [2, p. 268].) Since the Weyl sequences are "uniformly distributed" in the sense described, Franklin [3] has further examined the pseudo-random virtues of $\{X_k\}$ by a variety of statistical tests. This inherent randomness, together with their rich and well studied mathematical structure, makes it intriguing to see just how efficiently the Weyl sequences can be ordered.

The principal objective of this paper is to provide an algorithm which determines the order of $X_1, X_2, \cdots, X_n$ on the basis of fewer than $4 \log_2 n$ comparisons. We further show that any algorithm for sorting $\{(k\alpha) \bmod 1 : 1 \leqq k \leqq n\}$ by comparisons must make at least $\Omega(\log_2 n)$ comparisons, so the algorithm given here is the best possible.

One key motivation for studying the sorting of Weyl sequences is the general question: "How does one use the fact that a sequence is of a certain structure to provide a sorting algorithm which is information theoretically optimal?" This problem was explicitly posed in M. L. Friedman [4] and is implicit in Berlecamp's problem on sum set sorting (see, e.g., Harper, et al. [51]).

A second motivation for studying the sorting of Weyl sequences by comparisons is provided by recent work of Papadimitriou on efficient search for rationals. Papadimitriou [6] gives an elegant algorithm which establishes that $O(\log M)$ queries of the form "is $x \leqq p/q$", where $p, q \leqq M$, are sufficient to determine any rational $x = a/b$ with $a, b \leqq M$. The present algorithm is quite distinct from Papadimitriou's in method (relative comparisons vs. absolute comparisons) and in purpose (sort vs. search). Still, there is a close connection since (as the following sections implicitly show) the ordering

* Received by the editors December 18, 1978 and in final revised form May 16, 1980. This research was supported in part by the National Science Foundation under grants MCS 77-03659 and MCS 77-16974.

† Professor Martin H. Ellis of the Department of Mathematics, Northeastern University, Boston, Massachusetts, died on February 15, 1980, after a brief illness.

‡ Department of Statistics, Stanford University, Stanford, California 94305.

[1] θ, Ω, O, o denote "order of exactly", "order of at least", "order of at most", and "order of less than", respectively.

of $\{(k\alpha): 1 \leqq k \leqq n\}$ is closely connected to the location of the irrational α in the Farey dissection of the unit interval.

In the next section, we give an algorithm called Alpha-Sort, which is a very simple procedure which sorts any collection of the form $\{(k\alpha) \bmod 1 : 1 \leqq k \leqq n\}$ with fewer than $o(n)$ comparisons. The third section then uses the structures uncovered by Alpha-Sort to provide the required information theoretic lower bound $\Omega(\log_2 n)$. The fourth section applies a binary search speedup of Alpha-Sort which gives an explicit algorithm which performs as well as the theoretical lower bound can permit. The final section makes a brief speculation about the use of sorting as an appropriate measure of complexity of a pseudo random sequence.

2. Alpha-Sort: An $o(n)$ algorithm for sorting $(k\alpha)$ mod 1, $1 \leqq k \leqq n$. For brevity, we will subsequently write $\langle k\alpha \rangle$ for the representative of $(k\alpha)$ mod 1 in $[0, 1)$. The key idea for efficient sorting of $\{\langle ka \rangle : 1 \leqq k \leqq n\}$ is that the order structure can be completely determined from the largest and smallest elements of the set. We define L^* and R^* to be the integers in $\{1, 2, \cdots, n\}$ satisfying $\langle L^*\alpha \rangle = \min_{1 \leqq k \leqq n} \langle k\alpha \rangle$ and $\langle R^*\alpha \rangle = \max_{1 \leqq k \leqq n} \langle k\alpha \rangle$. The Alpha-Sort algorithm shows how one can compute L^* and R^*, and how these integers can be used to determine the ordering of $\langle k\alpha \rangle$, $1 \leqq k \leqq n$.

Alpha-Sort algorithm. Given $X_k = (k\alpha) \bmod 1$, $1 \leqq k \leqq n$, this algorithm returns $i_1, i_2, \cdots, i_n$ such that $X_{i_1} < X_{i_2} < \cdots < X_{i_n}$.

A1. [Initialize] Set $L \leftarrow 1$, $R \leftarrow 1$, $M \leftarrow 1$.

A2. [Compute L^* and R^*] While $L + R \leqq n$, set $R \leftarrow L + R$ if $X_{L+R-1} < X_{L+R}$; otherwise, set $L \leftarrow L + R$.

A3. Print $ML \bmod (L + R)$ if $ML \bmod (L + R) \leqq n$.

A4. Set $M \leftarrow M + 1$. If $M < L + R$ go to A3; otherwise end program.

The fact that Alpha-Sort correctly performs the task of sorting $\{\langle k\alpha \rangle : 1 \leqq k \leqq n\}$ with $O(n)$ comparisons, will follow from the next two lemmas. These elementary results will form the theoretical core for the rest of the analysis.

LEMMA 1. *Suppose* $\min\{\langle\alpha\rangle, \langle 2\alpha\rangle, \cdots, \langle j\alpha\rangle\} = \langle L\alpha \rangle$ *and* $\max\{\langle\alpha\rangle, \langle 2\alpha\rangle, \cdots, \langle j\alpha\rangle\} = \langle R\alpha\rangle$, $L, R \in \{1, 2, \cdots, j\}$. *Then*

(i) $\min\{\langle\alpha\rangle, \langle 2\alpha\rangle, \cdots, \langle (L + R - 1)\alpha\rangle = \langle L\alpha\rangle$,

(ii) $\max\{\langle\alpha\rangle, \langle 2\alpha\rangle, \cdots, \langle (L + R - 1)\alpha\rangle\} = \langle R\alpha\rangle$,

(iii) *either* $\langle (R + L)\alpha\rangle < \langle L\alpha\rangle$ *or* $\langle (R + L)\alpha\rangle > \langle R\alpha\rangle$.

Proof. (i). If $1 \leqq H < R$ and $\langle (L + H)\alpha\rangle < \langle L\alpha\rangle$, then $\langle (L + H)\alpha\rangle \neq \langle L\alpha\rangle + \langle H\alpha\rangle$; hence,

$$\langle L\alpha\rangle + \langle H\alpha\rangle > 1. \tag{1}$$

The definitions of L and R and (1) imply

$$\begin{aligned} \langle R\alpha\rangle &= \langle R\alpha\rangle - \langle H\alpha\rangle + \langle H\alpha\rangle \\ &= \langle (R - H)\alpha\rangle + \langle H\alpha\rangle \\ &\geqq \langle L\alpha\rangle + \langle H\alpha\rangle \\ &> 1. \end{aligned} \tag{2}$$

This contradiction establishes (i).

(ii). If $1 \leqq H < R$ and $\langle (L + H)\alpha\rangle > \langle R\alpha\rangle$, then the definitions of L and R imply

$$\begin{aligned} \langle (L + H - R)\alpha\rangle &= \langle (L + H)\alpha\rangle - \langle R\alpha\rangle \\ &= \langle L\alpha\rangle + \langle H\alpha\rangle - \langle R\alpha\rangle \\ &< \langle L\alpha\rangle. \end{aligned}$$

This is a contradiction to the fact that $\langle (L+H-R)\alpha\rangle > \langle L\alpha\rangle$.

(iii). Either

$$\langle (R+L)\alpha\rangle = \langle R\alpha\rangle + \langle L\alpha\rangle > \langle R\alpha\rangle,$$

or

$$\langle (R+L)\alpha\rangle = \langle R\alpha\rangle + \langle L\alpha\rangle - 1 < \langle L\alpha\rangle,$$

so (iii) is also established. □

LEMMA 2. *For* $\langle L\alpha\rangle = \min\{\langle k\alpha\rangle : 1 \leqq k \leqq n\}$ *and* $\langle R\alpha\rangle = \max\{\langle k\alpha\rangle : 1 \leqq k \leqq n\}$ *we have* $\langle L\alpha\rangle = \langle m_1\alpha\rangle < \langle m_2\alpha\rangle < \cdots < \langle m_S\alpha\rangle = \langle R\alpha\rangle$ *where* $m_k = kL \bmod (L+R)$ *and* $S = L+R-1$. *Furthermore, L and R are relatively prime, and* $\{m_i : 1 \leqq i \leqq S\}$ *is a permutation of the numbers* $1, 2, \cdots, S$.

Proof. First we will show by induction on n that L and R are relatively prime. If $n = 1$ then $L = R = 1$, so L and R are relatively prime. Suppose the assertion is true for $n = l$. If the maximum and minimum remain unchanged when n is raised to $l+1$, they remain relatively prime. If not, then Lemma 1 implies that $l+1 = L+R$ and either L or R (but not both) must be replaced with $L+R$. Since

$$\gcd\{L, L+R\} = \gcd\{L+R, R\} = \gcd\{L, R\} = 1,$$

the assertion is established.

Since L and $R+L$ are relatively prime,

$$\{\langle m_i\alpha\rangle : 1 \leqq i \leqq S\} = \{\langle i\alpha\rangle : 1 \leqq i \leqq S\}.$$

Suppose for some $1 \leqq i \leqq S$

(1) $$\langle m_i\alpha\rangle > \langle m_{i+1}\alpha\rangle.$$

If

(2) $$\langle m_{i+1}\alpha\rangle = \langle (m_i+L)\alpha\rangle,$$

then (1) implies

$$\langle m_{i+1}\alpha\rangle = \langle m_i\alpha\rangle + \langle L\alpha\rangle - 1 < \langle L\alpha\rangle;$$

this is impossible since by Lemma 1(i), $\langle L\alpha\rangle$ must be minimal. Since (2) fails to hold,

(3) $$\langle m_{i+1}\alpha\rangle = \langle (m_i-R)\alpha\rangle.$$

Since $R = m_S$ and $i < S$, Lemma 1(ii) implies $\langle m_i\alpha\rangle < \langle R\alpha\rangle$. But then,

$$\langle m_{i+1}\alpha\rangle = \langle m_i\alpha\rangle - \langle R\alpha\rangle + 1 > \langle m_i\alpha\rangle,$$

contradicting (1). Thus, $\langle m_i\alpha\rangle < \langle m_{i+1}\alpha\rangle$ for $1 \leqq i < S$, and the lemma is proved. □

The first lemma proves that step (A2) of Alpha-Sort correctly determines L^* and R^*. The second lemma shows how these two quantities completely determine the ordering of $\{\langle k\alpha\rangle : 1 \leqq k \leqq n\}$. We actually showed that L^* and R^* determine the ordering of the larger set $\{\langle k\alpha\rangle : 1 \leqq k \leqq L^*+R^*-1\}$; step (A3) of Alpha-Sort deletes the irrelevant members from the ordering of the larger set.

Since each comparison performed by Alpha-Sort increases either L or R, and since these quantities never exceed n, the total number of comparisons performed is at most $O(n)$. One can actually show that for any fixed α, as n increases only $o(n)$ comparisons are required. In fact, one can show that for almost every α (in the sense of Lebesque measure), for every $\varepsilon > 0$ Alpha-Sort will find L^* and R^* with $o((\log n)^{1+\varepsilon})$ comparisons. Still, by taking Liouville irrationals like $\sum_{K=1}^{\infty} 10^{-a_K}$ where $a_1 < a_2 < \cdots$ is a rapidly increasing sequence, one can also show that $o(n)$ is the most precise statement which one can make about the number of comparisons required by Alpha-Sort. We do not need to elaborate on these points since the next section will show that $o(n)$ is far from theoretically optimal, and the final section will sharpen Alpha-Sort to attain that optimal rate.

3. Information theoretic bounds. In the second lemma of the preceding section, we saw that the two quantities L^* and R^* completely determine the ordering of $\{\langle k\alpha\rangle : 1 \leqq k \leqq n\}$. We will now show that this suggests that it may be possible to sort $\{\langle k\alpha\rangle : 1 \leqq k \leqq n\}$ with only $O(\log_2 n)$ comparisons, but no fewer.

Classically, the fact that a binary tree with m leaves must have height at least $\log_2 (m)$, and the fact that there are $n!$ orderings of n real numbers, collectively imply that at least $O(n \log_2 n)$ comparisons are required to determine their order. This information theoretic perspective makes it interesting to determine E_n, the total number of orderings of $\{\langle k\alpha\rangle : 1 \leqq k \leqq n\}$ as α varies through all real values.

Explicitly, we let $|A|$ denote the cardinality of a finite set A and let σ denote any permutation of $\{1, 2, \cdots, n\}$. For

$$E_n = |\{\sigma : \text{for some } \alpha \in (0, 1) \langle \sigma(1)\alpha\rangle < \langle \sigma(2)\alpha\rangle < \cdots < \langle \sigma(n)\alpha\rangle\}|,$$

we have the following fact.

PROPOSITION. $n \leqq E_n \leqq n^2$.

Proof. For any α, we have (by Lemma 2 above) that there are integers $1 \leqq L^* \leqq n$, $1 \leqq R^* \leqq n$ which completely determine the ordering of $\{\langle k\alpha\rangle : 1 \leqq k \leqq n\}$. Since there are only n^2 such pairs L^* and R^*, the upper bound is established.

To see that $E_n \geqq n$, we consider the n irrationals defined by $\alpha_k = 1/k - \varepsilon$ for $k = 1, 2, \cdots, n$ and some very small positive irrational ε. For α_k one can see that $\langle \alpha_k\rangle < \langle 2\alpha_k\rangle < \cdots < \langle k\alpha_k\rangle$ but $\langle (k+1)\alpha\rangle < \langle \alpha_k\rangle$. The $\langle \alpha_k\rangle$ thus each yield a different ordering, so $E_n \geqq n$ as claimed.

Since the conclusions to be drawn from this proposition depend only on $O(\log_2 E)$, we have obtained only the simplest bounds. One can actually show that if ϕ is the Euler phi-function, we have $E_n = \sum_{k \leqq n} \phi(k) = 3/\pi^2 n^2 + O(n \log n)$ (for facts on $\phi(k)$ see [1]). The proposition immediately establishes the following result.

COROLLARY. *At least* $\Omega(\log_2 n)$ *comparisons are required in order to sort* $\{\langle k\alpha\rangle : 1 \leqq k \leqq n\}$.

The upper bound in the preceding proposition also suggests that it might be possible to sort $\{\langle k\alpha\rangle : 1 \leqq k \leqq n\}$ with only $O(\log_2 n)$ comparisons. The main objective of the next section will be to show that this is in fact the case.

4. Fast Alpha-Sort: An $O(\log n)$ algorithm. In Lemma 2, we proved that the ordering of $S = \{\langle \alpha\rangle, \langle 2\alpha\rangle, \cdots, \langle n\alpha\rangle\}$ is completely determined by L^* and R^* where $\langle L^*\alpha\rangle = \min_{1 \leqq k \leqq n} \langle k\alpha\rangle$ and $\langle R^*\alpha\rangle = \max_{1 \leqq k \leqq n} \langle k\alpha\rangle$. Now we will show how Alpha-Sort can be improved to compute these values with only $O(\log_2 n)$ comparisons.

The improvement over Alpha-Sort is made by replacing the linear process for computing L and R by a geometric process. The details are somewhat complicated due to the presence of several cases, but the conceptual essence of the matter is brought out

in the following Lemma 3, which shows essentially that if certain conditions are valid at times k and $2k$ they are valid at all intermediate times.

DEFINITION. A value $\langle j\alpha\rangle$ is called a *left extreme* (respectively *right extreme*) if $\langle j\alpha\rangle$ is the minimum (respectively maximum) of $\{\langle i\alpha\rangle : 1 \leqq i \leqq j\}$.

LEMMA 3. *Let k be a positive integer, and suppose $\langle R\alpha\rangle$ is a right extreme and each $\langle (L+iR)\alpha\rangle$, $0 \leqq i \leqq k$, is a left extreme. If $\langle (L+2kR)\alpha\rangle < \langle (L+kR)\alpha\rangle$, then each $\langle (L+iR)\alpha\rangle$, $k+1 \leqq i \leqq 2k$, is a left extreme.*

Proof. Let k be a positive integer for which each $\langle (L+jR)\alpha\rangle$, $0 \leqq i \leqq k$, is a left extreme. Let j be the smallest positive integer for which $\langle (L+jR)\alpha\rangle$ is not a left extreme, and suppose $j \leqq 2k$. Lemma 1 implies that $\max\{\langle\alpha\rangle, \langle 2\alpha\rangle, \cdots, \langle (L+(j-1)R)\alpha\rangle\} = \langle R\alpha\rangle$, and since $\langle (L+(j-1)R)\alpha\rangle$ is a left extreme, Lemma 1(iii) implies $\langle (L+jR)\alpha\rangle$ is an extreme; by choice of j it is not a left extreme, so it is a right extreme, hence,

$$\langle (L+jR)\alpha\rangle > \langle R\alpha\rangle. \tag{1}$$

Since each $\langle (L+iR)\alpha\rangle$, $0 \leqq i \leqq k$, is a left extreme,

$$\langle (L+iR)\alpha\rangle - \langle (L+(i+1)R)\alpha\rangle = 1 - \langle R\alpha\rangle, \text{ for all } 0 \leqq i < k. \tag{2}$$

Since $\langle (L+jR)\alpha\rangle > \langle L\alpha\rangle$, (2) implies

$$\langle (L+(j+i)R)\alpha\rangle - \langle (L+(j+i+1)R)\alpha\rangle = 1 - \langle R\alpha\rangle, \text{ for all } 0 \leqq i < k. \tag{3}$$

Expressing $\langle (L+2kR)\alpha\rangle$ and $\langle (L+kR)\alpha\rangle$ as telescoping sums and applying (1), (2), (3) and the fact that $j \geqq k+1$, we have

$$\begin{aligned}
\langle (L+2kR)\alpha\rangle &= \langle (L+jR)\alpha\rangle \\
&\quad + \sum_{i=0}^{2k-j-1} (\langle (L+(j+i+1)R)\alpha\rangle - \langle (L+(j+i)R)\alpha\rangle) \\
&> \langle R\alpha\rangle - (2k-j)(1-\langle R\alpha\rangle) \\
&= 1 - (2k+1-j)(1-\langle R\alpha\rangle) \\
&\geqq 1 - k(1-\langle R\alpha\rangle) \\
&> \langle L\alpha\rangle - k(1-\langle R\alpha\rangle) \\
&= \langle L\alpha\rangle + \sum_{i=0}^{k-1} (\langle (L+(i+1)R)\alpha\rangle - \langle (L+iR)\alpha\rangle) \\
&= \langle (L+kR)\alpha\rangle.
\end{aligned} \tag{4}$$

Inequality (4) shows that if there is a $j \in \{i : k+1 \leqq i \leqq 2k\}$ for which $\langle (L+jR)\alpha\rangle$ is not a left extreme, then $\langle (L+2kR)\alpha\rangle > \langle (L+kR)\alpha\rangle$. The lemma follows by contraposition. □

A similar argument establishes the following result.

LEMMA 4. *Let l be a positive integer, and suppose that each $\langle (R+iL)\alpha\rangle$, $0 \leqq i \leqq l$, is a right extreme. If $\langle (R+2lL)\alpha\rangle > \langle (R+lL)\alpha\rangle$, then each $\langle (L+iR)\alpha\rangle$, $l+1 \leqq i \leqq 2l$, is a right extreme.*

Besides serving to prove the validity of the following Fast Alpha-Sort algorithms, the preceding lemmas should also serve to motivate the algorithm. As a tool for use within Fast Alpha-Sort, we will require a binary search procedure which we call SEARCH(L, R, v, z). The parameters L, R, v are integers provided in the course of the Fast Alpha-Sort Algorithm, and z is either 0 or 1, depending on whether the algorithm

is looking for a new candidate for a left extreme or a right extreme. (The duality between the left and right procedures can be immediately seen, but for clarity we will not strain to unify the two.)

SEARCH(L, R, v, z).

S1. If $v = 1$ set SEARCH(L, R, v, z) $\leftarrow 1$ and stop.

S2. If $v = 2$ set SEARCH(L, R, v, z) $\leftarrow 3$ and stop if $z = 0$ and $L + 3R \leqq n$ and $\langle (L+2R)\alpha \rangle > \langle (L+3R)\alpha \rangle$, or if $z = 1$ and $3L + R \leqq n$ and $\langle (2L+R)\alpha \rangle < \langle (3L+R)\alpha \rangle$; otherwise if $v = 2$ set SEARCH(L, R, v, z) $\leftarrow 2$ and stop.

S3. If $v \geqq 3$ set $S \leftarrow 3 \cdot 2^{v-2}$ and set $T \leftarrow 2^{v-3}$.

S4. If $z = 0$ and $L + SR \leqq n$ and $\langle (L + \frac{1}{2}SR)\alpha \rangle > \langle (L+SR)\alpha \rangle$, or if $z = 1$ and $SL + R \leqq n$ and $\langle (\frac{1}{2}SL + R)\alpha \rangle < \langle (SL+R)\alpha \rangle$, set $S \leftarrow S + T$; otherwise set $S \leftarrow S - T$.

S5. Set $T \leftarrow \frac{1}{2}T$. If $T \geqq 1$ go back to S4.

S6. If $z = 0$ and $L + SR \leqq n$ and $\langle (L + (S-1)R)\alpha \rangle > \langle (L+SR)\alpha \rangle$, or if $z = 1$ and $SL + R \leqq n$ and $\langle ((S-1)L + R)\alpha \rangle < \langle (SL+R)\alpha \rangle$, set SEARCH($L, R, v, z$) $\leftarrow S$ and stop; otherwise set SEARCH(L, R, v, z) $\leftarrow S - 1$ and stop.

The SEARCH subroutine is used in the Fast Alpha-Sort algorithm, and the role it plays there is described in the proof of Lemmas 6 and 7.

Fast Alpha-Sort. Given $X_k = \langle k\alpha \rangle$, $1 \leqq k \leqq n$, this algorithm returns L^* and R^* such that $\langle \alpha L^* \rangle = \min_{1 \leqq k \leqq n} \langle k\alpha \rangle$, $\langle \alpha R^* \rangle = \max_{1 \leqq k \leqq n} \langle k\alpha \rangle$.

FA1. Set $L \leftarrow 0$, $R \leftarrow 1$.

FA2. Starting with $k = 1$, increment k until either $L + 2^k R > n$ or $\langle (L + 2^{k-1}R)\alpha \rangle < \langle (L + 2^k R)\alpha \rangle$.

FA3. Set $L \leftarrow L + R \cdot$ SEARCH($L, R, k, 0$). If $L + R > n$ go to FA6.

FA4. Starting with $l = 1$, increment l until either $R + 2^l L > n$ or $\langle (R + 2^{l-1}L)\alpha \rangle > \langle (R + 2^l L)\alpha \rangle$.

FA5. Set $R \leftarrow R + L \cdot$ SEARCH($L, R, l, 1$). If $L + R \leqq n$, go to FA2.

FA6. Let $L^* = L$ and $R^* = R$, then stop.

The main result of this section is the following:

THEOREM. *The Fast Alpha-Sort Algorithm returns L^* and R^* after at most $O(\log n)$ comparisons between pairs in $\{\langle i\alpha \rangle; 1 \leqq i \leqq n\}$.*

Before proving the theorem we will establish three lemmas.

LEMMA 5. *Computing* SEARCH(L, R, p, z) *requires at most $p - 1$ comparisons between pairs in $\{\langle i\alpha \rangle : 1 \leqq i \leqq n\}$.*

Proof. We simply dissect the possibilities. If $p = 1$, no comparisons are made. If $p = 2$, the only comparison that may be required is between $\langle (L+2R)\alpha \rangle$ and $\langle (L+3R)\alpha \rangle$ if $z = 0$, between $\langle (2L+R)\alpha \rangle$ and $\langle (3L+R)\alpha \rangle$ if $z = 1$. If $p \geqq 3$, then T is set to 2^{p-3} and single comparisons or no comparisons alternate with dividing T by 2, until $T < 1$. Thus, at most $p - 2$ comparisons are made before T becomes less than one. A single additional comparison may be made in (S6), for a total of at most $p - 1$ comparisons. □

LEMMA 6. *Suppose Fast Alpha-Sort has just entered step* (FA2), *$L = p$, $R = q$ and $\langle q\alpha \rangle$ is a right extreme. If each $\langle (p + iq)\alpha \rangle$, $1 \leqq i \leqq j$, is a left extreme but $\langle (p + (j+1)q)\alpha \rangle$ is not a left extreme, Fast Alpha-Sort will set L equal to $p + q * \min(j, \underline{(n-p)/q})$ after making at most $1 + 2 \min(\underline{\log_2 j}, \underline{\log_2((n-p)/q)})$ additional comparisons between pairs in $\{\langle i\alpha \rangle : 1 \leqq i \leqq n\}$.*

Proof. Let $b = \underline{\log_2 j}$ and let $c = \underline{\log_2((n-p)/q)}$.

If $p + 2^{b+1}q \leqq n$, Fast Alpha-Sort will sequentially compare $\langle (p + 2^{i-1}q)\alpha \rangle$ with

$\langle (p+2^i q)\alpha\rangle$, $1 \leqq i \leqq b+1$. It will then compute SEARCH$(p, q, b+1, 0)$, which by Lemma 5 requires at most b additional comparisons, after which it will set $L \leftarrow p+qj$. The total number of comparisons made is thus at most $1+2b$.

If $p+j1 \leqq n < p+2^{b+1}q$, Fast Alpha-Sort will follow the same procedure as above, except the comparison of $\langle (p+2^b q)\alpha\rangle$ with $\langle (p+2^{b+1}q)\alpha\rangle$ will be omitted; hence at most $2b$ comparisons will be made.

If $n < p+jq$, Fast Alpha Sort will sequentially compare $\langle (p+2^{i-1}q)\alpha\rangle$ with $\langle (p+2^i q)\alpha\rangle$, $1 \leqq i \leqq c$. Upon learning that $p+2^{c+1}q > n$, it will then compute SEARCH$(p, q, c+1, 0)$, which by Lemma 5 requires at most c additional comparisons, after which it will set $L \leftarrow p+\underline{[(n-p)/q]}q$. The total number of comparisons made is thus at most $2c$.

In any case, the total number of comparisons is at most $1+2\min(b, c)$. □

The following lemmas can be proved analogously to Lemma 6.

LEMMA 7. *Suppose Fast Alpha-Sort has just entered step* (FA4), $L = p$, $R = q$, *and* $\langle p\alpha\rangle$ *is a left extreme. If each* $\langle (ip+q)\alpha\rangle$, $1 \leqq i \leqq j$ *is a right extreme but* $\langle ((j+1)p+q)\alpha\rangle$ *is not a right extreme, Fast Alpha-Sort will set L equal to* $q+p*\min(j, \underline{(n-q)/p})$ *after making at most* $1+2\min(\underline{\log_2 j}, \underline{\log_2((n-1)/p)})$ *additional comparisons between pairs in* $\{\langle i\alpha\rangle : 1 \leqq i \leqq n\}$.

Proof of Theorem. Lemmas 6 and 7 imply that Fast Alpha-Sort correctly computes L^* and R^*. It remains to show that the number of comparisons between pairs in $\{\langle i\alpha\rangle : 1 \leqq i \leqq n\}$ made by Fast Alpha-Sort in computing L^* and R^* is $O(\log n)$.

Assume Fast Alpha-Sort has just computed L^* and R^*. Let

$$1 = q_0 < q_2 < \cdots < q_{2V} = R^*$$

denote the values taken by R during the course of Fast Alpha-Sort, let

$$0 = p_{-1} < p_1 < p_3 < \cdots < p_{2W-1} = L^*$$

denote the values taken by L during the course of Fast Alpha-Sort, and let $m = \max(2V, 2W-1)$. (Note that $m = 2V = 2W$ if step (FA6) was entered from (FA5) and $m = 2W-1 = 2V+1$ if step (FA6) was entered from (FA3).) For $1 \leqq i \leqq m$, let $j_i = \underline{p_i/q_{i-1}}$ if i is odd and let $j_i = \underline{q_i/p_{i-1}}$ if i is even. Note

$$\prod_{i=1}^{m} j_i \leqq \max(L^*, R^*) \leqq n. \tag{1}$$

Lemmas 6 and 7 imply that the number of comparisons between pairs in $\{\langle i\alpha\rangle : 1 \leqq i \leqq n\}$ made by Fast Alpha-Sort in computing L^* and R^* is bounded above by

$$\sum_{i=1}^{m} (1+\underline{2\log_2 j_i}),$$

which by (1) is bounded by

$$m + 2\log_2\left(\prod_{i=1}^{m} j_i\right) \leqq m + 2\log_2 n. \tag{2}$$

The largest value m can have would occur if the p_i and q_i grew as slowly as possible (i.e., $j_i = 1$ for $1 \leqq i \leqq m$), in which case each p_i or q_i would be the $(i+1)$st number in the Fibonacci sequence. In this case,

$$m \leqq 1 + \frac{\log_2 n}{\log_2 \phi}, \tag{3}$$

where $\phi = (\sqrt{5}+1)/2$. Inequalities (2) and (3) show that the number of comparisons

between pairs in $\{\langle i\alpha\rangle : 1 \leqq i \leqq n\}$ made by Fast Alpha-Sort in computing L^* and R^* is bounded above by

$$1 + (2 + (\log_2 \phi)^{-1}) \log_2 n,$$

which establishes the theorem. □

5. A brief speculation. The introduction isolated two motivations for studying the sorting of Weyl sequences, and a third motivation was deferred until now. This comes from the problem of measuring the complexity of a class of sequences and using this measurement to aid one's choice of pseudo-random number generators. The Weyl sequences are not genuine candidates for pseudo-random numbers, and this is reinforced by the speed with which they are sorted. One would especially like to determine the number of comparisons needed to sort sequences generated by the widely used classes of PRN generators. This analysis has many practical and conceptual complications, but the Weyl sequences can be considered a preliminary case in this wider program.

REFERENCES

[1] T. M. APOSTOL, *Introduction to Analytic Number Theory*, Springer-Verlag, New York, 1976.
[2] W. FELLER, *An Introduction to Probability Theory and Its Applications*, Vol. 2, Second Ed., John Wiley, New York, 1971.
[3] J. N. FRANKLIN, *Deterministic simulation of random processes*, Math. Comp., 17 (1963), pp. 28–59.
[4] M. L. FRIEDMAN, *How good is the information theory bound in sorting*, Theoret. Comput. Sci., 1 (1976), pp. 355–361.
[5] L. H. HARPER, T. H. PAYNE, J. E. SAVAGE AND E. STRAUSS, *Sorting X + Y*, Comm. ACM., 18 (1975), pp. 347–349.
[6] C. H. PAPADIMITRIOU, *Efficient Search for Rationals*, Technical Report 01–78, Center for Research in Computing Technology, Harvard University, 1978.