Discrete Mathematics 27 (1979) 47–57.

BOUNDS FOR SORTING BY PREFIX REVERSAL

William H. GATES
Microsoft, Albuquerque, New Mexico

Christos H. PAPADIMITRIOU*†
Department of Electrical Engineering, University of California, Berkeley, CA 94720, U.S.A.

Received 18 January 1978
Revised 28 August 1978

For a permutation σ of the integers from 1 to n, let $f(\sigma)$ be the smallest number of prefix reversals that will transform σ to the identity permutation, and let $f(n)$ be the largest such $f(\sigma)$ for all σ in (the symmetric group) S_n. We show that $f(n) \leq (5n+5)/3$, and that $f(n) \geq 17n/16$ for n a multiple of 16. If, furthermore, each integer is required to participate in an even number of reversed prefixes, the corresponding function $g(n)$ is shown to obey $3n/2-1 \leq g(n) \leq 2n+3$.

1. Introduction

We introduce our problem by the following quotation from [1]

> The chef in our place is sloppy, and when he prepares a stack of pancakes they come out all different sizes. Therefore, when I deliver them to a customer, on the way to the table I rearrange them (so that the smallest winds up on top, and so on, down to the largest at the bottom) by grabbing several from the top and flipping them over, repeating this (varying the number I flip) as many times as necessary. If there are n pancakes, what is the maximum number of flips (as a function $f(n)$ of n) that I will ever have to use to rearrange them?

In this paper we derive upper and lower bounds for $f(n)$. Certain bounds were already known. For example, consider any stack of pancakes. An *adjacency* in this stack is a pair of pancakes that are adjacent in the stack, and such that no other pancake has size intermediate between the two. If the largest pancake is on the bottom, this also counts as one extra adjacency. Now, for $n \geq 4$ there are stacks of n pancakes that have no adjacencies whatsoever. On the other hand, a sorted stack must have all n adjacencies and each move (flip) can create at most one adjacency. Consequently, for $n \geq 4$, $f(n) \geq n$. By elaborating on this argument, M.R. Garey, D.S. Johnson and S. Lin [2] showed that $f(n) \geq n+1$ for $n \geq 6$.

For upper bounds—algorithms, that is—it was known that $f(n) \leq 2n$. This can be seen as follows. Given any stack we may start by bringing the largest pancake on top and then flip the whole stack: the largest pancake is now at the bottom,

* Research supported by NSF Grant MCS 77-01193.

† Current address: Laboratory for Computer Science, Massachusets, Institute of Technology, Cambridge, Ma 02139, USA.

after two moves. Inductively, bring to the top the largest pancake that has not been sorted yet, and then flip it to the bottom of the unsorted stack. By $2n$ moves we will have thus sorted the whole thing. In fact, $2n$ can be improved to $2n-c$, constant c, by sorting the last few pancakes by a more clever method.

The list of obvious upper and lower bounds ends here. We show in Table 1 the known values of f. The seven first values were known to M.R. Garey, D.S. Johnson, and S. Lin in [2]. The last two are taken from [3].

Table 1

n	1	2	3	4	5	6	7	8	9
f	0	1	3	4	5	7	8	9	10

In Section 2, after introducing some notation and terminology, we prove that $f(n) \leq (5n+5)/3$ by designing a sorting algorithm that always has at least as good performance. In Section 3 we show that $f(n) \geq 17n/16$ infinitely often, by constructing, for each $k \geq 1$, a stack of $16k$ pancakes that requires $17k$ moves in order to be sorted. Finally, in Section 4 we derive bounds for $f(n)$ under the additional restriction that the pancakes must come out not only sorted, but also "right-side-up". In other words, each pancake must suffer an even number of flippings. The motivation is, of course, that the two sides of a pancake may not be the same, and the pancakes are required to come out of the sorting procedure "right side up". If $g(n)$ denotes the corresponding function for this modified problem, we can show that $(3n/2)-1 \leq g(n) \leq 2n+3$.

2. An algorithm

We will represent permutations in S_n as strings in Σ_n^*, where $\Sigma_n = \{1, 2, \ldots, n\}$. We will define a binary relation $\rightarrow$ in S_n by writing $\pi \rightarrow \sigma$ whenever $\pi = xy$, $\sigma = x^R y$, where $x, y \in \Sigma_n^*$ and x^R is the string x reversed (read backwards). If π is a permutation, $f(\pi)$ is the smallest k such that there exists a sequence of permutations $\pi \equiv \pi_0 \rightarrow \pi_1 \rightarrow \cdots \rightarrow \pi_k \equiv e_n$, where $e_n = 123 \cdots n$ is the identity permutation. $f(n)$ is the largest $f(\pi)$ over all $\pi \in S_n$.

Let π be a permutation in S_n. $\pi(j)$ is the jth number in π, where $1 \leq j \leq n$. If $|\pi(j) - \pi(j+1)| = 1$, we say that the pair $(j, j+1)$ is an *adjacency* in π. If $\pi = xby$, where x, b, $y \in \Sigma_n^*$ such that $(j, j+1)$ is an adjacency for $j = |x|+1, \ldots, |x|+|b|-1$, and b is maximal with respect to this property (i.e., $(|x|, |x|+1)$ and $(|x|+|b|, |x|+|b|+1)$ are not adjacencies), then b is called a *block*. If $\pi(j)$ is not in a block, i.e., $(j-1, j)$ and $(j, j+1)$ are not adjacencies in π, then $\pi(j)$ is *free*. For the purposes of this section we will consider $(j, j+1)$ to be an adjacency also if $\{\pi(j), \pi(j+1)\} = \{1, n\}$.

Our algorithm will sort the permutation π so as to create a total of $n-1$ adjacencies, that is, a block b of size n such as the ones shown in Figs. 1(a), 1(b). These permutations can then be transformed to e_n via at most four flippings (Figs.

1(c), 1(d), respectively). In the description of the algorithm below we use o to stand for one of $\{1, -1\}$. Addition is understood modulo n.

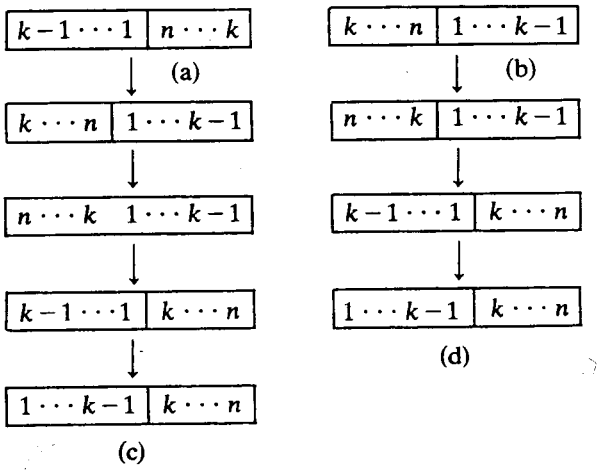

Fig. 1.

Algorithm $\mathcal{A}$

input: a permutation $\pi \in S_n$.
output: a permutation σ with $n-1$ adjacencies.

Set $\sigma = \pi$.
Repeat the following.
Let t be the first element of σ; i.e., $\sigma = t\sigma'$. (At least) one of the following eight cases applies. In each case take the corresponding action.

1. t is free, and $t+$o is also free. Perform the flipping shown in Fig. 2(a).
2. t is free, and $t+$o is the first element of a block. Perform the flipping shown in Fig. 2(b).
3. t is free, but both $t+1$ and $t-1$ are the last elements of blocks. Perform the sequence of flippings shown in Fig. 2(c).
4. t is in a block, and $t+$o is free. Perform the flipping shown in Fig. 2(d).
5. t is in a block, and $t+$o is the first element of a block. Perform the flipping shown in Fig. 2(e).
6. t is in a block with last element $t+k\cdot$o $(k>0)$, $t-$o is the last element of another block and $t+(k+1)\cdot$o is free. Perform the sequence of flippings shown in Fig. 2(f) or 2(g) (depending on the relative position of the two blocks and $t+(k+1)\cdot$o.
7. t is in a block with last element $t+k\cdot$o $(k>0)$, $t-$o is the last element of another block and $t+(k+1)\cdot$o is in a block. Perform the sequence of flippings in Fig. 2(h) or 2(k) (depending on whether $t+(k+1)\cdot 0$ is at the beginning or the end of its block).
8. None of the above. σ has $n-1$ adjacencies; halt.

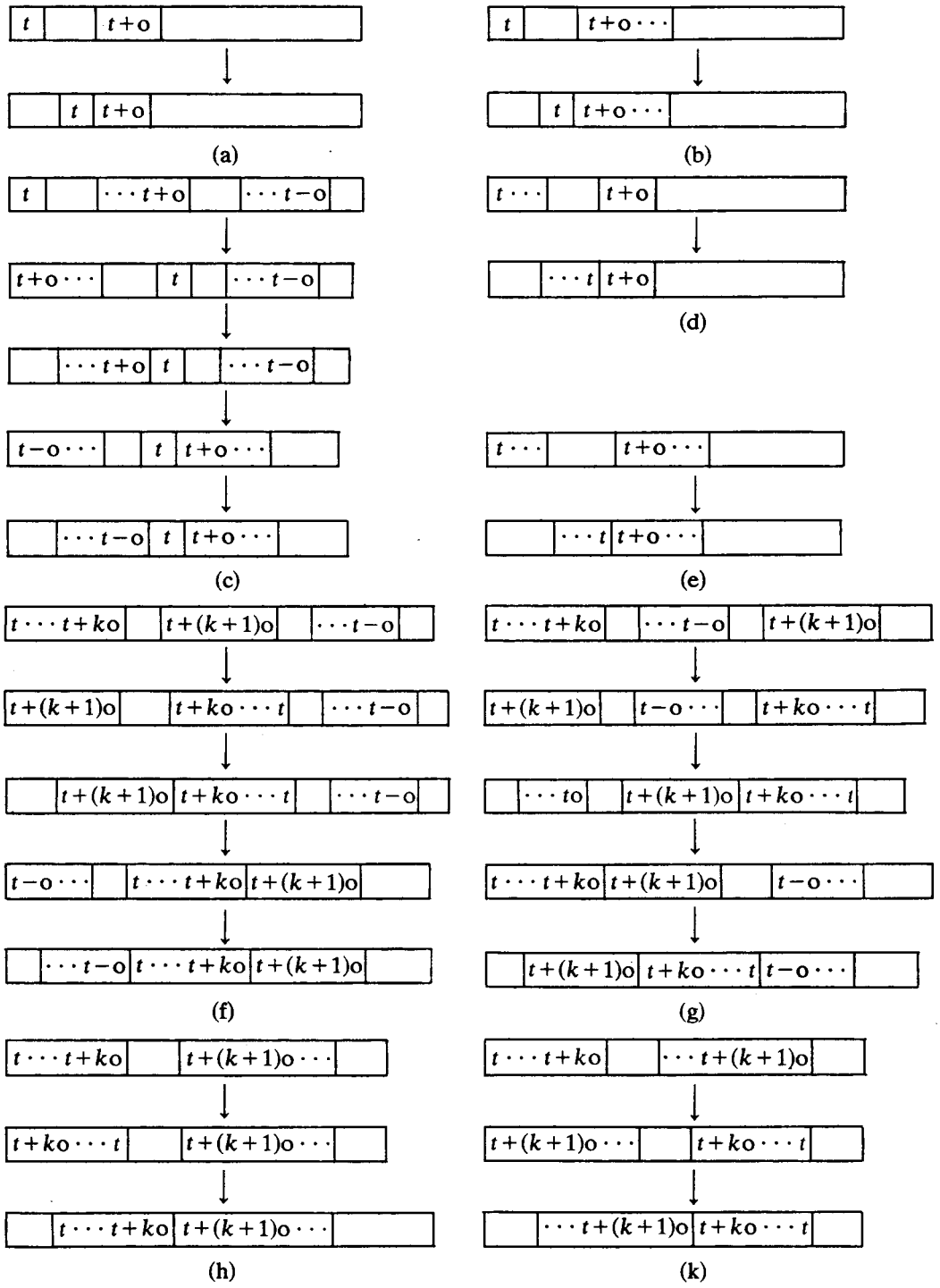

Fig. 2.

Theorem 1. *Algorithm $\mathcal{A}$ creates a permutation with $n-1$ adjacencies by at most $(5n-7)/3$ moves.*

Proof. First, it is clear that if we have a permutation σ with less than $n-1$ adjacencies, one of the cases 1 through 7 is applicable. Hence, the algorithm does not halt unless $n-1$ adjacencies have been created. Obviously the algorithm will eventually halt, since at each execution of the main loop at least one new

adjacency is created and none are destroyed. It remains, however, to prove that it does so in no more than $(5n-7)/3$ moves.

Call the action of case 1 *action 1*, the action in case 2 *action 2*, the action of cases 3 and 6 *action 3*, the action of case 4 *action 4*, and the action of cases 5 and 7 *action 5*, and *action 7*, respectively. Let x_i denote the number of actions of type i performed by an execution of the algorithm. The total number of moves (i.e., flippings) is given by

$$z = x_1 + x_2 + 4x_3 + x_4 + 2x_5 + x_7$$

where x_j is multiplied by the number of flippings involved in the action of type j (see first row of Table 2). Action 3 can be divided into four special cases, according to what happens in the flipping of Fig. 2(c) (or 2(f), or 2(g)) that comes before the last. The top of the stack before the flipping and the element next to $t-o$ may either

1. be non-adjacent,
2. form a new block,
3. merge a block with a singleton,
4. merge two blocks.

Accordingly, we distinguish among these subcases by writing $x_3 = x_{31} + x_{32} + x_{33} + x_{34}$. Now, since each action increases the number of adjacencies as indicated in Table 2, the total number of adjacencies in the conclusion of the algorithm is

$$n-1 = a + x_1 + x_2 + 2x_{31} + 3x_{32} + 3x_{33} + 3x_{34} + x_4 + x_5 + x_7. \tag{1}$$

Finally, if b is the number of blocks in π we have

$$b + x_1 - x_{31} - x_{33} - 2x_{34} - x_5 - x_7 = 1, \tag{2}$$

because each type of action increases or decreases the number of blocks as indicated in Table 2, we start with b blocks and we end up with 1 block. Also, notice that $b \leq a$, whereby (1) becomes

$$x_1 + x_2 + 2x_{31} + 3x_{32} + 3x_{33} + 3x_{34} + x_4 + x_5 + x_7 + b \leq n-1. \tag{3}$$

Table 2

Action	1	2	31	32	33	34	4	5	7
Number of flips	1	1	4	4	4	4	1	2	1
Increase in adjacencies	1	1	2	3	3	3	1	1	1
Increase in number of blocks	1	0	−1	0	−1	−2	0	−1	−1

Thus any possible application of the algorithm would, at worst, maximize

$$z = x_1 + x_2 + 4x_3 + x_4 + 2x_5 + x_7$$

subject to (2) and (3) above. We *claim* that the maximum is achieved for the values

$$x_1 = (n+1)/3, \qquad x_2 = 0, \qquad x_3 = x_{31} = (n-2)/3$$
$$x_4 = x_5 = x_7 = b = 0,$$

yielding a value of z equal to $(5n-7)/3$. To show our claim, recall the duality Theorem of Von Neuman, Kuhn and Tucker, Gale, and Dantzig [4] stating essentially that this maximum value equals the minimum value of the *dual linear program*:

$$\text{minimize} \quad \omega = \xi_2 + (n-1)\xi_3,$$

subject to the inequalities

$$\begin{aligned}
\xi_2 + \xi_3 &\geq 1,\\
\xi_3 &\geq 1,\\
-\xi_2 + 2\xi_3 &\geq 4,\\
3\xi_3 &\geq 4,\\
-\xi_2 + 3\xi_3 &\geq 4,\\
-2\xi_2 + 3\xi_3 &\geq 4,\\
\xi_3 &\geq 1,\\
-\xi_2 + \xi_3 &\geq 1,\\
-\xi_2 + \xi_3 &\geq 2,\\
\xi_2 + \xi_3 &\geq 0.
\end{aligned}$$

Thus, in order to prove our claim, we just have to exhibit a pair (ξ_2, ξ_3) satisfying these inequalities and having $\omega = \xi_2 + (n-1)\xi_3 = (5n-7)/3$. And such a pair is $\xi_2 = -2/3$, $\xi_3 = 5/3$.

The bound $f(n) \leq (5n+5)/3$ now follows directly, since it takes four more moves to transform a permutation with $n-1$ adjacencies to e. In any event, the constant term of the bound can be improved quite easily by stopping the algorithm when $n-k$, for some k, adjacencies have been formed, and then optimally putting together the $k+1$ pieces.

3. A lower bound

Let $\tau = 17536428$. For k a positive integer, τ_k denotes τ with each of the integers increased by $8 \cdot (k-1)$. In other words, τ_k is the sequence $1_k 7_k 5_k 3_k 6_k 4_k 2_k 8_k$, where $m_k = m + (k-1) \cdot 8$. Consider the permutation $\chi = \tau_1 \tau_2^R \tau_3 \tau_4^R \cdots \tau_{m-1} \tau_m^R$, where m is an even integer, and $n = |\chi| = 8 \cdot m$.

Theorem 2. $19n/16 \geqslant f(\chi) \geqslant 17n/16$.

Proof. To show the upper bound, we first do the following sequences of moves

$$\chi \to \tau_2\tau_1^R\tau_3 \cdots \to \tau_2^R\tau_1^R\tau_3 \cdots \to \tau_1\tau_2\tau_3 \cdots$$

and so on, bringing the even-indexed τ's in front and then back with the reversal cancelled in three moves. Thus, in $3n/16$ moves we obtain $\chi' = \tau_1\tau_2\tau_3 \cdots \tau_m$. Then, for each copy of τ in χ' we repeat the following sequence of eight moves (among a number of possibilities)

$$\begin{aligned}\chi' = x17536428y &\to 571x^R36428y \to 63x175428y \\ &\to 1x^R3675428y \to 45763x128y \to 67543x128y \\ &\to 76543x128y \to 21x^R345678y \to x12345678y.\end{aligned}$$

Thus in a total of $19n/16$ moves we can produce e starting from χ, and the upper bound is established.

For the lower bound, let $\chi \equiv \chi_0 \to \chi_1 \to \chi_2 \to \cdots \to \chi_{f(\chi)} \equiv e_n$ be an optimal sequence of moves for χ; each of χ_j, $j = 1, \ldots, f(\chi)$ is called a *move*. Let us call a move *k-stable* if it contains a substring of the form $1_k7_k\sigma 2_k8_k$ (or its reverse), where σ is a permutation of $\{3_k, 4_k, 5_k, 6_k\}$. We say that χ_j is an *event* if χ_{j-1} is k-stable, for some k, but $\chi_j, \chi_{j+1}, \ldots, \chi_{f(\chi)}$ are not.

Claim 1. *There are exactly m events.*

To prove Claim 1, we notice that χ_0 is k-stable for $k = 1, \ldots, m$, and $\chi_{f(\chi)}$ is not k-stable for any k. Furthermore, no permutation can cease being k_1-stable *and* k_2-stable, $k_1 \neq k_2$, in only one move.

Let us call χ_j a *waste* if χ_j has no more adjacencies than χ_{j-1}. (Here, by an *adjacency* in σ we mean any pair $(i, i+1)$ such that either $i < n$ and $|\sigma(i) - \sigma(i+1)| = 1$, or $i = n$ and $\sigma(i) = n$.) Let w denote the total number of wastes among $\{\chi_j : j = 1, \ldots, f(\chi)\}$.

Claim 2. $f(\chi) \geqslant n + w$.

To see why this is true, one just has to notice that χ has no adjacencies, e has n adjacencies, and any move that is not a waste creates just one adjacency.

By our Claim 1 we conclude that in the optimal sequence that we are considering there are m events as shown below ($\overset{*}{\to}$ is the transitive closure of $\to$)

$$\chi_{i_1} \overset{*}{\to} \chi_{i_2} \overset{*}{\to} \chi_{i_3} \overset{*}{\to} \cdots \overset{*}{\to} \chi_{i_m}.$$

Claim 3. *For all* j, $1 \leq j \leq m-1$, *there exists a waste* χ_l *with* $i_j \leq l \leq i_{j+1}$.

To prove Claim 3, suppose that it fails. In other words, suppose that there is an event i_j other than the last one, such that all moves χ_l, $i_j \leq l \leq i_{j+1}$ construct a new adjacency without destroying an existing adjacency. Suppose that k is the appropriate index for which χ_{i_j-1} is the last k-stable permutation in the sequence considered. Then, $\chi_{i_j-1} = x1_k7_k\sigma 2_k8_k y$, where x and y are strings of integers and σ is a permutation of $\{3_k, 4_k, 5_k, 6_k\}$. Notice that since our basic string $\tau =$ 17536428 is symmetric (in that $i+j=9$ if and only if $\tau(i)+\tau(j)=9$) this is not a loss of generality. For simplicity in our notation, we shall omit the subscript k in the rest of this part of our argument; we shall also assume that $\sigma = 5364$, since the argument is identical for any σ. Thus

$$\chi_{i_j-1} = x17536428y.$$

We distinguish among two cases.

Case 1. $x = \varepsilon$, the empty string. Since χ_{i_j} is neither a waste nor k-stable, we must have

$$\chi_{i_j} = 46357128y.$$

Now, we must not, according to our hypothesis, have a waste until *after* the next event. This, however, is impossible, since the first move after χ_{i_j} which flips more than four elements is a waste.

Case 2. $x \neq \varepsilon$. That is $\chi_{i_j-1} = x17536428y$. Since χ_{i_j} is neither a waste nor k-stable, it must be the case that $x = 9z$, and $\chi_{i_j} = 2463571z^R98y$. Again, we must not have a waste until after the next event. This means that the only moves permitted are local rearrangements of the integers $\{1, 3, 4, 5, 6, 7\}$; thus

$$\chi_{i_j} = 2463571z^R98y \xrightarrow{*} 7654321z^R98y.$$

Again, the next move has to be a waste.

The theorem now follows directly from Claims 1, 2, and 3.

$$f(\chi) \geq n + w \geq n + \frac{m}{2} = \frac{17n}{16}.$$

4. Bounds for a restricted version

Let us define a binary relation $\Rightarrow$ on $S_n \times 2^{\{1,\ldots,n\}}$ as follows: $(\sigma, S) \Rightarrow (\sigma', S')$ if and only if $\sigma = xy$, $\sigma' = x^R y$, and $S' = S \oplus X$, where X is the set of integers

involved in x, and $\oplus$ stands for symmetric difference. Let $g(\sigma)$ be the shortest chain in $\Rightarrow$ from $(\sigma, \emptyset)$ to $(e_n, \emptyset)$, and let $g(n)$ be the largest $g(\sigma)$ over all $\sigma \in S_n$.

Theorem 3. $g(n) \leqslant 2n+3$.

Proof. First observe that $g(\sigma)$ is not greater than $f(\sigma')$ where $\sigma' \in S_{2n}$ is defined as follows, for each $\sigma \in S_n : \sigma'(2i-1) = 2\sigma(i)-1$ and $\sigma'(2i) = 2\sigma(i)$ for all $i = 1, \ldots, n$. The complexity of sorting σ' without the restriction can now be bounded from above by the algorithm $\mathcal{A}$ of Section 2. The equations governing the complexity of $\mathcal{A}$ when applied to σ' are (1), (2), and (3) of Section 2 with n replaced by $2n$, $b = a = n$, and also noting that only x_5 can be nonzero, since all other actions are possible only in the presence of free elements. The maximum is therefore $2n-2$. Allowing five more moves to sort the resulting permutation, we get the claimed bound.

We shall now derive a lower bound for $g(n)$. The "hard" permutation in this case is $e_n^R = n, n-1, \ldots, 2, 1$, a permutation which is next to trivial with our restriction removed. Consider an optimal sequence for e_n^R

$$A_0 \equiv (e_n^R, \emptyset) \Rightarrow A_1 \equiv (\chi_1, S_1) \Rightarrow \cdots \Rightarrow A_j \equiv (\chi_j, S_j) \Rightarrow \cdots A_{g(e_n^R)} \equiv (e_n, \emptyset).$$

A pair $(i, i+1)$, $i < n$, is an *adjacency in* (χ, S) if either $\chi(i+1) = \chi(i)+1$ and $\chi(i)$, $\chi(i+1) \notin S$ or $\chi(i+1) = \chi(i)-1$ and $\chi(i)$, $\chi(i+1) \in S$. A pair $(i, i+1)$ is an *anti-adjacency* in (χ, S) if either $\chi(i+1) = \chi(i)-1$ and $\chi(i), \chi(i+1) \notin S$ or $\chi(i+1) = \chi(i)+1$ and $\chi(i), \chi(i+1) \in S$. A move A_j is a *waste* if there are no more adjacencies in A_j than there are in A_{j-1}. A set $\{\chi(i), \chi(i+1), \ldots, \chi(j)\}$, $j > i$, such that $(k, k+1)$, $i \leqslant k < j$, is an anti-adjacency in (χ, S) is called a *clan*. Notice that e_n^R is all non-adjacencies; in other words, it has one big clan, namely $\{1, \ldots, n\}$. At each move we may "break" at most one clan C replacing it by two new clans C_1, C_2 with $C_1 \cup C_2 = C$. Thus, the process of sorting e_n^R can be thought of partly as "breaking up clans", since e_n has no clans. Interestingly enough, it is this aspect of sorting e_n^R whose complexity can be captured quite easily.

A move A_j is called an (a, b)-*cut* if a clan C in A_{j-1} is replaced by two clans C_1, C_2 in A_j such that $C = C_1 \cup C_2$ and $|C_1| = a$, $|C_2| = b$.

Lemma 1. *Let A_j be an (a, b) cut.*

(1) *If $a, b > 1$, then both A_j, A_{j+1} are wastes,*

(2) *If the only one of $a, b > 1$, then either A_j or A_{j+1} is a waste.*

Proof. An easy case-by-case analysis.

Theorem 4. $g(e_n^R) \geqslant \frac{3}{2}n - 1$.

Proof. e_n^R has no adjacencies; so $g(e_n^R) \geq n + w$, where w is the number of wastes in the sequence considered. In order to bound w from below, let x_1 be the number of cuts in (1) above, x_2 (and x_3) are the numbers of moves of case (2) of Lemma 1 which are (resp. are not) wastes, and $x_4(x_5)$ the number of (1, 1)-cuts that are (resp. are not) wastes. Finally, let y be the total number of moves that result each in the creation of a clan C from either another clan C' $|C'| = |C| - 1$ and a singleton, or from two singletons. It is easy to see that such a move is a waste, and cannot be a cut. Obviously we have

$$w \geq x_1 + x_2 + x_4 + y, \tag{4}$$

and

$$x_1 + x_2 + x_3 + x_4 + x_5 \geq y + n - 1, \tag{5}$$

because at least $y + n - 1$ cuts must be eventually produced.

We next observe that

$$n + 2y - 1 \geq x_2 + x_3 + 2x_4 + 2x_5, \tag{6}$$

because we start with no singletons (elements not in a clan), we end up with n, each move counted by y (a y-move for short) absorbs at most two singletons, each x_2-move or x_3-move creates a singleton, and each x_4 or x_5-move creates two singletons. Finally, we claim that

$$w - x_1 \geq x_3 - 1. \tag{7}$$

To prove this, we shall show how each x_3-move, except for the last, can be paired off with a waste that is not an x_1-move. By Lemma 1, each x_3-move is followed by a waste. If this waste is an x_1-move, it must be followed by another waste, by Lemma 1. Thus every x_3-move except for the last is followed by a sequence of x_1-moves (possibly empty) followed by a waste that is not an x_1-move.

How small can w be? To find out, we minimize w subject to inequalities (4), (5), (6), and (7). The minimum is $n/2 - 1$, achieved at

$$w = n/2 - 1 = x_2, \qquad x_3 = n/2, \qquad x_1 = x_4 = x_5 = y = 0.$$

To show this, we just need to exhibit, as in the proof of Theorem 1, the dual variables $\xi_4 = 1$, $\xi_5 = \xi_6 = \xi_7 = 1/2$, with the same value. We therefore conclude that $w \geq n/2 - 1$, and hence $g(e_n^R) \geq 3n/2 - 1$.

5. Discussion

We presented an algorithm sorting any permutation of length n in about $5n/3$ prefix reversals; improving the multiplicative constant seems to be quite challenging. We also described a technique for deriving lower bounds for $f(n)$, and showed how it can be used to establish that $f(n) \geq 17n/16$. Improving on this particular lower bound does not appear too hard; in fact, we conjecture that for our "hard" permutation χ, $f(\chi) = 19n/16$. Also, slightly better lower bounds may be conceivably proved by using different τ's—of length 7, say. However, we do not know how the upper and lower bounds can be narrowed significantly. Naturally, it is not clear at all that $f(n)/n$ converges, and hence it may be that no better bounds are attainable.

Acknowledgment

We wish to thank Mike Garey and Harry Lewis for suggesting this problem to us. We acknowledge the comments and corrections of Jacob E. Goodman (the "Harry Dweighter" of [1]) and of an anonymous referee.

References

[1] Amer. Math. Monthly 82 (1) (1975) 1010.
[2] Amer. Math. Monthly 84 (4) (1977) 296.
[3] David P. Robbins, private communication, November 1977.
[4] G.B. Dantzig, Linear Programming and Extensions (Princeton University Press, Princeton, 1963).

Note added in proof

Ervin Győri of the Hungarian Academy of Sciences and György Turán of J. Atilla University have independently discovered a proof of Theorem 1. Their algorithm and proof are essentially the same as ours.